HEC MONTRÉAL

Précarité énergétique au Québec : Revue de littérature et construction d'indicateurs

par

Coralie Gilbert

Pierre-Olivier Pineau
HEC Montréal
Codirecteur de recherche

Denis Larocque
HEC Montréal
Codirecteur de recherche

Sciences de la gestion
(Spécialisation Intelligence d'affaires)

Mémoire présenté en vue de l'obtention
du grade de maîtrise ès sciences
(M. Sc.)

Octobre 2024
© Coralie Gilbert, 2024

Les membres du jury qui ont évalué ce mémoire ont demandé des corrections mineures.

Résumé

Ce mémoire analyse la précarité et la vulnérabilité énergétiques au Québec, où un habitant sur dix éprouve des difficultés à payer ses factures d'électricité, malgré des tarifs abordables.
Nous adoptons une approche multidimensionnelle pour définir ces concepts et identifier des indicateurs pertinents.
À travers une revue de littérature et une analyse des données d'Hydro-Québec et de Statistique Canada, nous développons un nouvel indice composite de vulnérabilité énergétique qui s'appuie sur les travaux d'Okushima.
Ce travail permet d'identifier les municipalités les plus vulnérables en considérant divers facteurs d'exposition, de sensibilité et de capacité d'adaptation. En comparant notre indice au célèbre indicateur 2M, nous discutons des implications de nos résultats pour l'élaboration de politiques énergétiques plus efficaces et inclusives au Québec.

Sommaire

Sommaire.....4

Liste des tableaux et des graphiques..... 6

Glossaire.....9

Introduction..... 11

Revue de littérature.....13

1. Qu'est-ce que la précarité énergétique ?..... 13

2. La précarité énergétique au Canada et au Québec..... 15

3. La pauvreté énergétique dans le monde au travers de différents indicateurs..... 17

3.1. Les types de mesure de la pauvreté et de la précarité énergétique..... 17

3.1.1. Indicateurs individuels: accès et quantités minimales.....17

3.1.2. Indicateurs économiques..... 18

3.1.3. Indices composites : pourquoi parler de vulnérabilité énergétique ?..... 21

3.2. Quels indicateurs utiliser pour la précarité et la vulnérabilité énergétiques dans un contexte québécois ?..... 24

3.2.1. L'indicateur 2M comme point de départ..... 24

3.2.2. Méthode pour notre indice composite..... 25

Collecte et Analyse des Données : Modélisation des Indicateurs de Précarité Énergétique..... 27

4. Méthodologie.....27

4.1. Municipalités concernées.....28

4.2. Présentation des bases de données.....28

4.2.1. Base de données Hydro-Québec.....28

4.2.2. Base de données de Statistique Canada.....29

4.2.3. Données sur le climat - Bibliothèque Meteostat.....29

4.2.4. Préparation des codes géographiques des municipalités.....30

4.2.5. Données géographiques sur les municipalités.....30

4.3. Nettoyage de la base de données d'Hydro Québec.....30

4.3.1. Imputation.....30

4.3.2. Données obtenues..... 32

4.4. Nettoyage des bases de données de Statistique Canada et association avec les

données d'Hydro-Québec..... 33
4.5. Obtention des données climatiques..... 33
4.6. Sélection des variables..... 34
4.6.1. Variables qui évaluent le risque d'exposition.....34
4.6.2. Variables qui évaluent la capacité d'adaptation..... 35
4.6.3. Variables qui évaluent la sensibilité.....35
4.7. Détermination des k-voisins..... 38
4.7.1. Chargement des librairies..... 38
4.7.2. Sélection des Variables et définition du nombre de voisins (k)..... 38
4.7.3. Détermination des seuils de précarité énergétique..... 38
5. Résultats..... 39
5.1. Municipalités sans valeur manquante..... 39
5.1.1. Méthode des k-voisins.....39
5.1.2. Indice 2M.....43
5.1.3. Statistiques descriptives comparatives..... 48
5.2. Municipalités avec valeurs imputées.....50
5.2.1. Méthode des k-voisins.....50
5.2.2. Indice 2M.....54
5.2.3. Statistiques descriptives comparatives..... 58
5.3. Discussion des résultats.....61
Conclusion.....64
Bibliographie..... 66
Annexe..... 76

Liste des tableaux et des graphiques

Tableau 1 : Récapitulatif des indicateurs économiques les plus populaires..... 17

Schéma : Les trois composantes de la vulnérabilité..... 18

Tableau 2 : Récapitulatif des régressions linéaires robustes pour les variables sélectionnées dans notre modèle..... 32

Tableau 3 : Statistiques descriptives pour les municipalités sans valeur manquante et avec valeurs imputées..... 33

Carte 1 : Municipalités en situation de précarité énergétique identifiées grâce à la méthode des k-voisins pour les municipalités sans valeurs manquantes..... 35

Nuage de points 1 : Municipalités selon la consommation d'électricité par ménage mensuelle moyenne en kWh en fonction du revenu($) (méthode des k-voisins pour les municipalités sans valeurs manquantes)..... 36

Nuage de points 2 : Municipalités selon la consommation d'électricité par ménage mensuelle moyenne en kWh en fonction du pourcentage de personnes âgées (méthode des k-voisins pour les municipalités sans valeurs manquantes)..... 36

Nuage de points 3 : Municipalités selon la consommation d'électricité par ménage mensuelle moyenne en kWh en fonction du pourcentage de réparations majeures requises (méthode des k-voisins pour les municipalités sans valeurs manquantes)..... 37

Nuage de points 4 : Municipalités selon la consommation d'électricité par ménage mensuelle moyenne en kWh en fonction du pourcentage de logements construits avec 1981 (méthode des k-voisins pour les municipalités sans valeurs manquantes)..... 38

Nuage de points 5 : Municipalités selon la consommation d'électricité par ménage mensuelle moyenne en kWh en fonction de la température moyenne en hiver(°C) (méthode des k-voisins pour les municipalités sans valeurs manquantes)..... 39

Cartes 2.a et 2.b : Municipalités en situation de précarité énergétique identifiées avec l'indicateur 2M pour les municipalités sans valeurs manquantes..... 40

Nuage de points 6 : Municipalités selon la consommation d'électricité par ménage mensuelle moyenne en kWh en fonction du revenu($) (indicateur 2M pour les municipalités sans valeurs manquantes)..... 41

Nuage de points 7 : Municipalités selon la consommation d'électricité par ménage mensuelle moyenne en kWh en fonction du revenu($) (indicateur 2M pour les municipalités

sans valeurs manquantes)..... 41

Nuage de points 8 : Municipalités selon la consommation d'électricité par ménage mensuelle moyenne en kWh en fonction du pourcentage de personnes âgées (indicateur 2M pour les municipalités sans valeurs manquantes)..... 42

Nuage de points 9 : Municipalités selon la consommation d'électricité par ménage mensuelle moyenne en kWh en fonction du pourcentage de réparations majeures requises (indicateur 2M pour les municipalités sans valeurs manquantes)..... 43

Nuage de points 10 : Municipalités selon la consommation d'électricité par ménage mensuelle moyenne en kWh en fonction du pourcentage de logements construits avant 1981 (indicateur 2M pour les municipalités sans valeurs manquantes)..... 43

Nuage de points 11 : Municipalités selon la consommation d'électricité par ménage mensuelle moyenne en kWh en fonction de la température moyenne en hiver (indicateur 2M pour les municipalités sans valeurs manquantes)..... 44

Tableau 4 : Statistiques descriptives pour les municipalités sans valeur manquante en situation de précarité énergétique selon la méthode k-NN ou l'indicateur 2M..... 45

Carte 3 : Municipalités en situation de précarité énergétique identifiées grâce à la méthode des k-voisins pour les municipalités avec valeurs imputées..... 47

Nuage de points 12 : Municipalités selon la consommation d'électricité par ménage mensuelle moyenne en kWh en fonction du revenu($) (méthode des k-voisins pour les municipalités avec valeurs imputées)..... 48

Nuage de points 13 : Municipalités selon la consommation d'électricité par ménage mensuelle moyenne en kWh en fonction du pourcentage de personnes âgées (méthode des k-voisins pour les municipalités avec valeurs imputées)..... 48

Nuage de points 14 : Municipalités selon la consommation d'électricité par ménage mensuelle moyenne en kWh en fonction du pourcentage de réparations majeures requises (méthode des k-voisins pour les municipalités avec valeurs imputées)..... 49

Nuage de points 15 : Municipalités selon la consommation d'électricité par ménage mensuelle moyenne en kWh en fonction du pourcentage de logement construit avant 1981 (méthode des k-voisins pour les municipalités avec valeurs imputées)..... 49

Cartes 4.a et 4.b : Municipalités en situation de précarité énergétique selon l'indice 2M pour les municipalités avec valeurs imputées..... 50

Nuage de points 16.a et 16.b : Municipalités selon la consommation d'électricité par ménage mensuelle moyenne en kWh en fonction du revenu($) (indicateur 2M pour les municipalités avec valeurs imputées)..... 51

Nuage de points 17 : Municipalités selon la consommation d'électricité par ménage mensuelle moyenne en kWh en fonction du pourcentage de personnes âgées (indicateur 2M pour les municipalités avec valeurs imputées).....52

Nuage de points 18 : Municipalités selon la consommation d'électricité par ménage mensuelle moyenne en kWh en fonction du pourcentage de réparations majeures (indicateur 2M pour les municipalités avec valeurs imputées).....52

Nuage de points 19 : Municipalités selon la consommation d'électricité par ménage mensuelle moyenne en kWh en fonction du pourcentage de logements construits avant 1981 (indicateur 2M pour les municipalités avec valeurs imputées).....53

Nuage de points 20 : Municipalités selon la consommation d'électricité par ménage mensuelle moyenne en kWh en fonction de la température moyenne en hiver (indicateur 2M pour les municipalités avec valeurs imputées).....54

Tableau 5 : Statistiques descriptives pour les municipalités avec valeurs imputées en situation de précarité énergétique selon la méthode k-NN ou l'indicateur 2M.....55

Tableau 6 : Récapitulatif de la liste des municipalités en situation de précarité énergétique selon la méthode k-NN et l'indicateur 2M.....57

Glossaire

2M

Le terme "2M" fait référence à un indicateur de précarité énergétique qui évalue la situation énergétique d'un individu, ménage ou dans notre cas, municipalité en observant la part de revenu consacré aux dépenses énergétiques. Si elle dépasse deux fois la médiane de l'échantillon, on déclare que cette entité est en situation de précarité énergétique.

Méthode des k-voisins

La méthode des k-voisins, ou k-NN (k-Nearest Neighbors), est une technique d'analyse utilisée pour classer ou prédire des résultats en fonction de données similaires dans un ensemble de données. Dans le cadre de cette étude, cette méthode a été utilisée pour identifier les municipalités en situation de précarité énergétique en se basant sur des indicateurs de vulnérabilité énergétique. Elle évalue la situation d'une municipalité en la comparant à celles de ses voisines immédiates, prenant en compte des facteurs comme la consommation d'électricité, le revenu et d'autres caractéristiques démographiques.

Pauvreté énergétique

La pauvreté énergétique est un état dans lequel un ménage est incapable de s'assurer un accès adéquat aux services énergétiques nécessaires pour répondre à ses besoins fondamentaux, tels que le chauffage, l'éclairage et la cuisine, sans devoir consacrer une part disproportionnée de son revenu à ces dépenses. Cette notion est souvent associée à la notion de précarité énergétique, mais elle se concentre davantage sur l'accès et la disponibilité des services énergétiques et aura tendance à plutôt être utilisée pour les ménages des pays en voie de développement.

Précarité énergétique

La précarité énergétique désigne une situation dans laquelle un ménage fait face à des difficultés pour répondre à ses besoins énergétiques de manière stable et abordable. Cela peut inclure des défis liés aux dépenses énergétiques excessives par rapport au revenu du ménage, des habitudes de consommation qui ne sont pas optimales ou des logements inefficaces sur le plan de l'énergie. La précarité énergétique est souvent mesurée par des indicateurs comme le seuil des 10 % (où un ménage consacre plus de 10 % de son revenu à ses factures d'énergie) ou d'autres seuils relatifs à la consommation d'énergie.

Vulnérabilité énergétique

La vulnérabilité énergétique est une notion plus large qui englobe la précarité énergétique ainsi que d'autres dimensions de la susceptibilité d'un ménage à faire face à des défis

énergétiques. Elle prend en compte non seulement les risques d'accès limité aux services énergétiques, mais aussi la capacité d'adaptation des ménages et leur sensibilité face à des événements défavorables, tels que des augmentations de prix ou des conditions climatiques extrêmes. Cette notion considère les facteurs contextuels, les conditions socio-économiques et la qualité des logements pour comprendre les différentes formes de vulnérabilité que peuvent rencontrer les ménages en matière d'énergie.

Avant-propos et remerciements

Depuis plusieurs années, les questions de justice sociale occupent une place prépondérante dans mes préoccupations académiques et personnelles. C'est dans ce cadre que mes recherches sur la précarité énergétique au Québec ont pris forme. Il est déconcertant de constater le manque d'informations disponibles sur ce sujet crucial qui affecte tant de ménages dans notre province. Ce constat a été le moteur de mon désir de développer un indice mesurant cette précarité, dans le but de contribuer au débat et d'éclairer une problématique encore largement méconnue.

Je tiens à souligner que ce travail représente seulement un point de départ. Comprendre pleinement la précarité énergétique au Québec nécessitera l'engagement de nombreux chercheurs et la collaboration interdisciplinaire pour approfondir les analyses et proposer des solutions concrètes. Il est également crucial de garder à l'esprit que, si la précarité énergétique est problématique, les questions de sobriété énergétique posent également des défis, notamment pour de nombreux ménages québécois qui consomment bien plus d'énergie que dans d'autres provinces du Canada. Je suis reconnaissante d'avoir eu l'opportunité de participer à cette initiative et d'apporter ma contribution à cette réflexion collective.

Je voudrais également exprimer ma profonde gratitude à mes co-directeurs de mémoire, Pierre-Olivier Pineau et Denis Larocque, qui ont été bien plus que de simples accompagnateurs dans mes recherches. Leur incroyable disponibilité et leur engagement constant à offrir des conseils judicieux ont été des atouts inestimables tout au long de ce projet. Ils ont su créer un environnement de travail stimulant et bienveillant, toujours prêts à m'accompagner, à m'encourager et à partager leur vaste expertise. Grâce à leur soutien, mon travail a été considérablement enrichi, tant au niveau intellectuel qu'humain.
Enfin, un immense merci à Mahault, qui m'a été d'un soutien inestimable tout au long de ce parcours.

Sans leur appui ainsi que celui de toutes les personnes qui m'ont entourée durant la rédaction de ce mémoire, ce projet n'aurait pas été possible.

Introduction

Selon les données de Statistique Canada, près d'un ménage sur huit au Québec était en situation de pauvreté en 2017, soit presque 500 000 ménages. A l'échelle du pays, c'est un Canadien sur dix qui déclare avoir du mal à payer ses factures d'électricité (CBC 2023).

Bien que le Québec ait l'électricité la moins chère du Canada pour tous ses résidents (Energy Hub 2023), certains ménages québécois en situation de précarité peuvent rencontrer des difficultés à payer leur facture d'énergie.

La précarité énergétique se manifeste lorsque les coûts d'énergie représentent une part trop importante du revenu d'un consommateur, ou lorsque ceux-ci doivent réduire leur consommation d'énergie à un niveau qui compromet leur santé et leur bien-être (définition de la Commission européenne).

Ce phénomène, bien que propre à chaque foyer, résulte d'une intersection complexe entre plusieurs dimensions, telles que le revenu des ménages, les dépenses énergétiques et les conditions de logement. C'est pourquoi la précarité énergétique demeure un enjeu critique à aborder au Québec d'autant plus que la situation économique de nombreux ménages s'est aggravée, avec une augmentation préoccupante du taux de personnes en situation de faible revenu au Québec (Statistique Canada 2024), ce qui accentue la problématique de la précarité énergétique.

Ce mémoire analyse la précarité et la vulnérabilité énergétiques au Québec en adoptant une approche multidimensionnelle qui se veut complémentaire des indicateurs traditionnels.

Notre recherche a pour objectif de réfléchir à leurs définitions et à l'identification d'indicateurs pertinents pour le Québec, en s'appuyant sur une revue de littérature nationale et internationale. Dans ce contexte, nous nous interrogeons : dans quelle mesure les méthodes d'analyse, telles que la méthode des k-voisins et l'indicateur 2M, révèlent-elles les disparités en matière de précarité énergétique entre différentes municipalités du Québec ?

Dans un premier temps, nous procédons à une revue de littérature approfondie qui nous permet de revisiter les définitions de la précarité énergétique. Nous mettons ainsi en lumière les limites des approches existantes tout en présentant une définition opérationnelle adaptée aux réalités du Québec.

Cette revue s'articule autour de plusieurs sections. En premier lieu, nous répondons à la question "Qu'est-ce que la précarité énergétique ?", en établissant un cadre conceptuel qui permet de mieux comprendre ce phénomène (Section 1). Ensuite, nous examinons la précarité énergétique spécifiquement au Canada et au Québec, ce qui nous permet de contextualiser le problème à l'échelle nationale et locale (Section 2).

Nous élargissons ensuite notre analyse à la pauvreté énergétique dans le monde, en explorant divers indicateurs qui permettent de mesurer cet enjeu sur la scène internationale (Section 3). Nous caractérisons les types de mesures de la pauvreté et de la précarité

énergétique, en distinguant notamment les indicateurs individuels relatifs à l'accès à l'énergie et aux quantités minimales nécessaires (Section 3.1.1), ainsi que les indicateurs économiques qui évaluent l'impact des coûts énergétiques sur le budget des ménages (Section 3.1.2).

Nous soulignons également l'importance des indices composites, justifiant ainsi l'intérêt de parler de vulnérabilité énergétique et de sa pertinence pour une analyse plus complète de la situation (Section 3.1.3). Par la suite, nous cherchons à identifier les indicateurs les plus adaptés pour mesurer la précarité et la vulnérabilité énergétiques dans un contexte québécois, en prenant comme point de départ l'indicateur 2M (Section 3.2.1) et en décrivant la méthode retenue pour élaborer notre indice composite (Section 3.2.2).

Dans un second temps, nous présentons une méthodologie quantitative s'appuyant sur les données d'Hydro-Québec, de Statistique Canada, et sur les données climatiques de Meteostat, une API de données climatiques, afin de développer un nouvel indice composite de vulnérabilité énergétique.

L'application de cet indice permet par la suite d'identifier les municipalités québécoises les plus vulnérables, en considérant à la fois les facteurs d'exposition, les facteurs de sensibilité et la capacité d'adaptation des ménages. Enfin, nous analysons les résultats, en comparant les performances de notre indice à celles du 2M, un indicateur de précarité énergétique populaire qui considère comme en situation de précarité énergétique les ménages – et dans notre cas les municipalités – qui dépensent plus de deux fois la médiane de revenu consacré à l'électricité au sein de l'échantillon. Finalement, nous discutons des implications de nos résultats pour l'élaboration de politiques énergétiques plus efficaces et inclusives au Québec.

Nous espérons que ce travail contribuera à l'ensemble des recherches visant à améliorer les politiques énergétiques au Québec.

Revue de littérature

1. Qu'est-ce que la précarité énergétique ?

La précarité énergétique est peu étudiée au Canada en raison d'un manque de définition claire et de politiques spécifiques. (Das et al. 2022)

Historiquement, le terme "pauvreté énergétique" (“energy poverty”) était utilisé en référence à l'accès à l'énergie dans les pays moins développés ou industrialisés (Nussbaumer et al. 2012) et la recherche dans ce domaine se concentre surtout sur les questions d'accessibilité à l'énergie dans les pays en développement, avec un accent sur le développement institutionnel et infrastructurel (Primc et al. 2021).

À l'autre extrémité, les études sur la “fuel poverty”, la précarité énergétique, dans les pays développés se concentrent principalement sur l'abordabilité de l'énergie, avec une attention particulière portée aux besoins humains (Primc et al. 2021).

Les deux termes sont le fruit d’un long processus de recherches et d’analyses de chercheurs à travers le monde. Dans leur papier *30 years of energy and fuel poverty research: A retrospective analysis and future trends,* Kaja Primc et Al.(2021) nous résument la recherche et les avancées sur ce sujet au travers de plusieurs jalons.

Les concepts de précarité et de pauvreté énergétique ont émergé dans les années 1970, notamment avec la crise pétrolière. En 1979, Isherwood et Hancock ont introduit la pauvreté énergétique, et en 1991, Boardman l'a définie comme une situation où un ménage dépense plus de 10 % de son revenu pour atteindre une température intérieure adéquate.

Par souci de clarté, nous utiliserons le terme "précarité énergétique" pour caractériser le phénomène au Québec mais aussi car le concept de précarité explore historiquement les causes structurelles de la précarité énergétique dans les pays développés (Primc et al. 2021) et, bien que ce mémoire soit conçu pour attirer l’attention sur la précarité énergétique au Québec plutôt qu’une analyse exhaustive du sujet, il a été élaboré en tenant compte d'une série d'études qui ont exploré les processus politiques, socio-économiques, institutionnels et culturels contribuant à la vulnérabilité et à l'appauvrissement des individus.

Depuis les travaux de Boardman, les chercheurs qui se sont penchés sur le sujet ont été de plus en plus nombreux à reconnaître le caractère complexe de la précarité énergétique et les définitions. Il convient de noter que ce phénomène va au-delà de simplement maintenir un logement chaud et désigne plus largement les défis auxquels font face les ménages pour avoir accès aux services énergétiques nécessaires à leurs besoins sociaux et matériels (Bouzarovski et Petrova 2015). De plus, le concept de précarité énergétique se distingue du concept de précarité en général car il n’est pas seulement lié à des difficultés financières mais aussi à des questions d’infrastructures énergétiques interconnectées et/ou domestiques comme l’efficacité énergétique des logements (Boardman 1991).

Nous définirons donc la précarité énergétique comme "une incapacité à accomplir des activités indispensables en raison, directe ou indirecte, d'un accès insuffisant à des services énergétiques abordables, fiables et sûrs, en tenant compte des alternatives raisonnables disponibles pour accomplir ces activités." selon la définition de Day, Walker and Simcock (2016). En effet, face à l'historique des définitions proposées pour ce concept, celle-ci se veut sensible aux circonstances locales et adaptable au contexte. Cette approche nuancée est essentielle, car la précarité énergétique n'est pas seulement un problème théorique; elle se traduit par des réalités concrètes et douloureuses dans la vie quotidienne des individus concernés.

En effet, la précarité énergétique a de nombreuses conséquences sur le quotidien des individus qui en souffrent.

Tout d'abord, elle entraîne des impacts physiologiques importants, notamment pour les personnes les plus vulnérables physiquement, comme les personnes âgées, handicapées, ou celles souffrant de maladies chroniques, ainsi que les bébés et les nourrissons. Ces groupes sont particulièrement sensibles aux variations de température, ce qui peut aggraver leur état de santé.

Ensuite, la précarité énergétique affecte également la santé mentale, en augmentant les risques de dépression, d'anxiété et de stress. Ce lien entre les conditions de vie énergétique et la santé mentale est de plus en plus documenté (Bentley et al. 2023; Xu et al. 2022).

Par ailleurs, elle est associée à divers problèmes de santé physique. En effet, la recherche a également établi des liens entre la précarité énergétique et des maladies telles que les maladies cardiovasculaires, les affections respiratoires, ainsi qu'un risque accru de grippe, de pneumonie, d'asthme et même d'accidents domestiques (Bentley et al. 2023).

Ces conditions peuvent également conduire à des comportements à risque, les personnes en situation de précarité énergétique étant plus susceptibles d'adopter des comportements tels que le tabagisme, la suralimentation ou la consommation excessive d'alcool, exacerbant ainsi leur mauvais état de santé (Middlemiss 2022).

Enfin, les répercussions sociales sont également considérables. La précarité énergétique peut freiner la participation sociale, avoir un impact négatif sur l'emploi, et même affecter la scolarité des jeunes. Par exemple, certaines interventions visant à réduire la précarité énergétique ont montré qu'une amélioration des conditions de vie énergétique pouvait entraîner une baisse des absences au travail et à l'école, notamment grâce à une meilleure santé globale (Middlemiss 2022).

Si les conséquences de la précarité énergétique sur le quotidien des individus sont significatives, cette problématique s'inscrit également dans un cadre plus global. En effet, la pauvreté énergétique plus largement est le reflet d'inégalités socio-économiques profondes, qui varient considérablement d'une région à l'autre du monde. Pour comprendre pleinement ses impacts, il est nécessaire de considérer non seulement les individus, mais aussi les dynamiques géopolitiques et économiques mondiales qui façonnent l'accès à l'énergie.

2. La précarité énergétique au Canada et au Québec

Au Québec, le taux de faible revenu des personnes âgées de 16 ans et plus s'est établi à 9,9 % en 2021, marquant une augmentation notable par rapport à 2020, où il était de 8,2 % (Institut de la Statistique du Québec 2023). Cette hausse reflète une détérioration des conditions économiques pour une portion de la population et peut inquiéter sur l'augmentation de la précarité énergétique dans la province. Bien que cette problématique varie considérablement d'une région à l'autre au Canada, il est important de souligner que la recherche sur la précarité énergétique dans le contexte québécois reste encore limitée (Fonds de Recherche du Québec 2023).

Cela nous pousse à examiner les chiffres relatifs à la précarité en général au Québec ou à la précarité énergétique au Canada ainsi qu'à faire l'état de la situation énergétique au Québec pour établir un portrait plus complet de la situation dans la province.

D'après Das, Martiskainen et Li (2022), en utilisant la mesure des 10 %, les taux de précarité énergétique fluctuent d'une province à l'autre, allant de 3 % en Alberta à 20 % dans les provinces de l'Atlantique.
Une autre étude de Riva et al. (2021) souligne que, selon les mesures des 10 % et des 2M, entre 6 % et 19 % des ménages canadiens vivent en situation de précarité énergétique. Ce chiffre est particulièrement alarmant dans les provinces atlantiques, où plus de 30 % des ménages sont touchés selon la mesure des 2M.
En somme, un ménage canadien sur cinq fait face à la précarité énergétique (Riva et al. 2021). Cette proportion est plus élevée que dans de nombreux pays européens, pour lesquels la qualité de vie est comparable. Contrairement au Canada, ces pays ont considéré la précarité énergétique comme une priorité politique et de recherche depuis plusieurs années, comme en témoignent les nombreux États européens qui ont adopté une définition officielle de ce phénomène (EPAH 2022, Riva et al. 2021).

En effet, le Royaume-Uni a réalisé un travail pionnier sur le sujet de la précarité énergétique et a appliqué ses méthodes à l'échelle du pays. L'implication du Royaume-Uni sur ce sujet a inspiré de nombreuses avancées en Europe. La Commission européenne a financé l'Observatoire de la précarité énergétique de l'UE (EPOV) de 2016 à 2020, qui est ensuite devenu le Centre de conseil sur la précarité énergétique (EPAH) en 2021 avec pour mission de fournir aux autorités locales des recommandations et des outils pour s'attaquer à un large éventail de problèmes, tels que la précarité énergétique. Ce projet a abouti à différentes législations favorisant une transition équitable vers la neutralité climatique et la rénovation des bâtiments les moins performants pour réduire la précarité énergétique à travers l'Europe. En revanche, les pays d'Amérique du Nord n'ont pas formellement reconnu la précarité énergétique comme un problème distinct (Riva 2021, Briot-Arthur et al. 2024) et on ne retrouve aucune définition officielle de la précarité énergétique en ce qui concerne le gouvernement québécois ou canadien.

Cependant, la précarité énergétique est un enjeu crucial dans la poursuite de la transition énergétique du secteur résidentiel canadien et des stratégies d'adaptation au climat,

puisque les ménages en situation de précarité énergétique entrent désavantagés dans cette transition (Middlemiss, 2022).

Il est important de prendre compte du contexte canadien lorsqu'on veut aborder la précarité énergétique au Québec : la distribution de l'énergie y est gérée au niveau provincial, ce qui signifie que le profil de la précarité énergétique de chaque province doit être étudié individuellement (Tardy et Lee 2019). De plus, au sein des provinces, les zones rurales et urbaines sont soumises à la précarité énergétique de manière différente, la prévalence de la précarité énergétique étant plus élevée dans les zones rurales par rapport aux zones urbaines (Das, Martiskainen et Li 2022; Riva et al. 2021).
En ce qui concerne le Québec, 77% de la consommation en énergie du secteur résidentiel concerne l'électricité (Pineau et Whitmore 2024) majoritairement fournie par Hydro-Québec, créée en 1944 pour assurer un accès à une énergie abondante et à faible coût. Hydro-Québec a été mandatée pour offrir l'énergie à des tarifs bas tout en préservant une saine gestion financière.
Pourtant, bien que les Québécois bénéficient de tarifs avantageux — l'électricité étant facturée à 6,5 cents par kilowattheure pour le bloc patrimonial, contre 8,7 à 18,2 cents en Ontario (Perreault 2024) —, la précarité énergétique reste un défi à relever.

Le Plan directeur en transition, innovation et efficacité énergétiques élaboré par le Québec vise à stimuler la rénovation et la construction éconergétiques dans le secteur résidentiel à travers divers programmes (Gouvernement du Québec, 2024) mais il convient de souligner qu'il ne fait aucune mention non plus de la précarité ou de la pauvreté énergétique.
On trouve bien au Québec quelques initiatives pour améliorer la situation comme le programme *Rénoclimat* qui offre une aide financière pour des travaux tels que l'amélioration de l'isolation, le remplacement de portes et fenêtres, et l'installation de systèmes pour la ventilation et le chauffage ou encore le programme *Chauffez vert* qui permet de bénéficier d'un soutien financier pour la transition vers le chauffage électrique. De plus, le programme *Novoclimat* soutient la construction de maisons ou d'immeubles dépassant les standards énergétiques en vigueur.
Enfin, le programme *Éconologis* de la TEQ (Transition Energétique du Québec) livre un programme centralisé d'efficacité énergétique financé par la province et destiné à tous les ménages à faible revenu du Québec depuis 2008. Néanmoins, la priorité du dispositif Éconologis est de parer aux urgences : Beslay et al. en 2010 rapportent dans leur article que des agents du projet ont « l'impression de faire des pansements, mais pas de traiter le problème ».

Dès lors, en ne tenant pas compte de la complexité et de la nature multidimensionnelle de la précarité énergétique, ces programmes risquent d'être inefficaces, voire d'aggraver les injustices existantes.
En effet, les connaissances actuelles soulignent l'importance d'aborder la précarité et l'exclusion sociale au Québec pour cibler les ménages les plus en difficulté. Par exemple, la Loi de 2002 (Gouvernement du Québec 2002) visant à lutter contre la précarité et l'exclusion sociale et le Plan pour une économie verte 2030 (Gouvernement du Québec 2024), reconnaissent que la lutte contre la précarité nécessite des efforts prolongés, englobant l'intégration sociale, la santé et l'éducation.

3. La pauvreté énergétique dans le monde au travers de différents indicateurs

La pauvreté énergétique d'un point de vue global fait écho à des inégalités économiques et géopolitiques marquées.

En effet, la consommation d'énergie et le développement économique sont étroitement liés au sein d'un pays ou d'une région : un pays en développement observe souvent une augmentation de sa consommation d'énergie provenant de sa croissance économique. Par exemple, la consommation d'énergie en Chine a explosé, passant de 150 kWh par habitant en 1970 à 3 000 kWh en 2010. En comparaison, la consommation d'électricité par habitant en Éthiopie a très peu évolué sur la période et en 2010, elle était 250 fois inférieure à celle des États-Unis (González-Eguino 2015).
Les inégalités économiques existant dans le monde se reflètent ainsi dans des inégalités similaires en matière de consommation d'énergie.

Si l'on se réfère à la définition initiale de la précarité énergétique de Day, Walker et Simcock (2016), le manque d'"alternatives raisonnables" est aussi un facteur important de la pauvreté énergétique dans les pays en développement.
Les sources d'énergie principales les plus couramment utilisées dans les pays en développement restent souvent basiques, comme le bois ou le charbon, avec peu d'alternatives disponibles, en particulier dans les zones rurales. En revanche, les pays plus riches accèdent à une diversité de sources d'énergie, ce qui reflète des inégalités structurelles dans la capacité à répondre aux besoins énergétiques (González-Eguino 2015).

Pour ces raisons, la précarité énergétique ne peut être définie que relativement à la réalité économique, technologique et sociale d'un pays ou d'une région et ses facteurs ne seront pas les mêmes d'une zone géographique à une autre. De la même manière, les indicateurs utilisés pour la mesurer ne seront pas les mêmes d'une zone géographique à une autre et sont révélateurs des défis auxquels font face les régions dans lesquelles ils sont utilisés.

3.1. Les types de mesure de la pauvreté et de la précarité énergétique

La précarité énergétique peut être mesurée par plusieurs approches dont certaines seront plus adaptées selon le niveau de richesse et de développement d'une région mais aussi le niveau d'expertise de l'audience visée.

3.1.1. Indicateurs individuels: accès et quantités minimales

Dans un premier temps, les indicateurs individuels sont faciles à utiliser. Ils véhiculent un message puissant et impartial, facile à interpréter en ce qui concerne une dimension spécifique. Cependant, de telles mesures offrent une vision limitée du problème mesuré.

Pour les régions où l'accès même à des services d'énergie énergétique est limité, ces indicateurs peuvent suffire pour dresser un tableau de la situation et pousser à des passages à l'action rapide (Siksnelyte-Butkiene et al. 2021).

Pour évaluer la précarité énergétique, il est courant d'utiliser différents seuils qui varient en général selon le niveau de développement. Ainsi, pour les pays en voie de développement, les indicateurs typiques utilisés seront soit des seuils technologiques, soit des seuils physiques.

Le seuil technologique évalue la précarité énergétique comme un problème d'accès aux services énergétiques "modernes", tels que l'électricité et les sources autres que la biomasse pour la cuisson.

Selon l'Agence Internationale de l'énergie en 2022, il y avait 760 millions de personnes dans le monde vivant sans accès à l'électricité, tandis que 2,3 milliards de personnes continuaient de dépendre de combustibles de cuisson nocifs et polluants tels que le charbon, le bois, les déchets agricoles et le fumier animal . Cependant, cet indicateur ne renseigne pas sur les niveaux de consommation qui sont eux aussi cruciaux pour déterminer une situation énergétique.

Dès lors, cette approche technologique est souvent complétée par l'utilisation de seuils physiques qui seront des indicateurs eux aussi particulièrement intéressants pour les pays en voie de développement.

Cette approche estime la consommation d'énergie minimale nécessaire pour répondre aux besoins de base. Par exemple, si le seuil est fixé à 100 kWh d'électricité et 150 litres d'essence (équivalent à 5 GJ), en 2009, 1,8 milliard de personnes ne pouvaient pas couvrir leurs besoins énergétiques de base (27 % de la population mondiale) (González-Eguino 2015).

Si l'on souhaite explorer plus en détails la précarité au sein des pays développés, les études privilégient une approche prenant en compte les revenus des ménages ou des individus. Elle met en évidence des enjeux liés au pouvoir d'achat et à l'accès à des niveaux de confort énergétique.

3.1.2. Indicateurs économiques

Parmi les indicateurs officiellement appliqués dans des politiques nationales, nombreux sont ceux qui choisissent une approche économique pour caractériser l'obstacle financier qui est bien souvent source de précarité énergétique.

Au Royaume-Uni, la mesure des 10 % a été largement utilisée pour évaluer la précarité énergétique, et elle demeure la méthode officiellement adoptée en Irlande du Nord aujourd'hui (Northern Ireland Department for Communities) . Cette approche définit les ménages en situation de précarité énergétique comme ceux qui dépensent plus de 10 % de leur revenu sur les factures d'énergie. L'étude qui a théorisé cette mesure est celle de E. S. Boardman, publiée en 1991 dans *Fuel Poverty: From Cold Homes to Affordable Warmth*.

Dans son ouvrage, Boardman obtient cet indicateur en sélectionnant deux fois la proportion du revenu médian consacré à l'énergie au Royaume-Uni à l'époque.
Bien que cette mesure soit souvent critiquée aujourd'hui pour être jugée trop simpliste et régulièrement appliquée sans discernement à des contextes nationaux très différents, elle reste un indicateur largement utilisé notamment au Canada (Dionne, Heisel, and Situ 2024) car elle est très facile à communiquer et ne change pas selon les évolutions de la population. Cependant, afin de mieux cerner les nuances de la précarité énergétique, d'autres indicateurs plus ciblés ont été développés.

La mesure 2M, deux fois la médiane du revenu consacré à l'énergie, est la conceptualisation de l'indicateur 10%.
C'est un indicateur de précarité énergétique qui évalue les ménages selon le pourcentage de leur revenu consacré aux dépenses énergétiques (Boardman, 1991).
Plus précisément, un ménage est considéré en situation de précarité énergétique si ses dépenses énergétiques dépassent un seuil déterminé de son revenu, généralement fixé à deux fois la médiane du pourcentage de revenu consacré aux dépenses énergétiques des ménages de l'échantillon.
Pour clarifier son fonctionnement, prenons un exemple concret : supposons qu'après analyse, on détermine que la médiane des dépenses énergétiques pour les ménages dans une municipalité est de 7 % du revenu après impôts. Dans ce cas, le seuil de précarité énergétique selon l'indicateur 2M serait de 14 % (deux fois cette médiane). Cela implique qu'un ménage qui consomme plus de 14 % de son revenu après impôts pour couvrir ses factures énergétiques serait considéré comme étant en situation de précarité énergétique.
Cet indicateur est prisé pour sa capacité à offrir une vision plus adaptée au contexte local de la précarité énergétique que l'indicateur 10%, en tenant compte des spécificités de l'échantillon d'intérêt. De plus, l'indicateur 2M permet un recalcul annuel, ce qui aide à prendre en compte les fluctuations économiques, telles que les variations des prix de l'énergie et les changements dans le revenu.
Cependant, cette mesure, tout comme le seuil des 10 %, fait l'objet de critiques, car elle ne prend pas en compte les ménages à hauts revenus qui, pour des raisons d'inefficacité énergétique, pourraient dépasser ces seuils sans être réellement en situation de précarité. Par exemple, un ménage aisé vivant dans un logement mal isolé pourrait voir son pourcentage de dépenses énergétiques dépasser le seuil sans que cela ne reflète une réelle précarité, car il est financièrement capable de faire face à ces coûts.
Ajoutons aussi qu'en 1988, l'année de référence pour Boardman (1991), le 2M (soit 10% du revenu consacré à l'électricité pour les ménages dans son cas) représentait la part moyenne réelle des dépenses énergétiques parmi les 30 % des ménages les plus pauvres de Grande-Bretagne et il s'est avéré que cette part représentait aussi grossièrement le double de la part médiane des dépenses énergétiques réelles pour tous les ménages.
Or, si de nos jours, les chercheurs acceptent largement la double part médiane comme une considération guide, tandis que la proximité avec les dépenses moyennes réelles des ménages pauvres est considérée comme une simple coïncidence, la justification du 2M devrait à l'inverse s'appuyer sur les dépenses énergétiques des strates de revenu les plus faibles et non sur la double part médiane de l'ensemble des dépenses énergétiques d'un échantillon (Croon et al. 2023).
De plus, il est crucial de noter que des ménages à faible revenu peuvent se retrouver dans des situations où ils doivent choisir entre se chauffer ou se nourrir, un phénomène connu

sous le nom d'effet "heat or eat" (Burns 2023; "FINAL: Energy Poverty Factsheet 2024"; Government of Canada 2023). Cette réalité souligne une limitation notable des indicateurs comme le 2M et le seuil des 10 %, qui se concentrent principalement sur les dépenses énergétiques par rapport aux revenus, sans tenir compte de la tension que certains ménages éprouvent dans leurs choix quotidiens.

En réponse à ces préoccupations, l'indicateur LIHC a été officiellement adopté comme indicateur de précarité énergétique au Royaume-Uni de 2015 à 2020 (Gouvernement du Royaume-Uni 2020). Comme son nom l'indique, l'indicateur LIHC définit la précarité énergétique comme la combinaison de la confrontation à des coûts élevés et d'un faible revenu. Cette approche implique de fixer deux seuils : un pour le revenu et un pour les coûts.
Au Royaume Uni, il était calculé en sélectionnant un ménage à bas revenu comme ayant un revenu inférieur à 60% du revenu médian et ayant des factures d'électricité supérieures à la médiane nationale (Gouvernement du Royaume-Uni 2012).
Cette mesure permet d'identifier les ménages qui tombent dans la précarité énergétique en raison des coûts élevés des combustibles et des bas revenus, aidant ainsi les décideurs politiques à prioriser et à cibler les améliorations d'efficacité énergétique vers les ménages qui nécessitent des réductions significatives de leurs factures de combustibles pour sortir de la précarité énergétique (Kantamneni and Haley 2021).

Dans le même esprit d'évaluation des conditions économiques des ménages, l''indicateur de revenu minimum standard (MIS) considère un ménage comme étant en dessous du seuil de précarité énergétique si son revenu, après dépenses énergétiques, est inférieur au revenu minimum standard pour se loger et répondre à ses besoins de base dans le pays ou dans la région (Moore 2012).
Cet indicateur vise à mesurer la précarité énergétique en fonction du revenu, offrant ainsi une adaptabilité à divers contextes socio-économiques.
Cependant, la capacité de l'indicateur à identifier les personnes vulnérables est limitée, car définir le MIS approprié implique la tâche difficile de déterminer le revenu minimum pour couvrir les besoins d'un ménage donné.

Néanmoins, bien qu'utiles, ces indicateurs économiques ne font souvent appel qu'à deux voire trois dimensions de la précarité énergétique (les revenus, la consommation en énergie et les dépenses en énergie des ménages).
D'autres indices similaires peuvent être retrouvés dans ce document de la commission européenne : Trinomics 2016

Afin de prendre en compte toute la complexité de la précarité énergétique, les chercheurs et les décideurs politiques se tournent de plus en plus vers des indices composites visant à capter plus de dimensions de la précarité énergétique.

Tableau 1 : Récapitulatif des indicateurs économiques les plus populaires

Indicateur économique	Définition	Pays Utilisateurs	Avantages	Inconvénients	Source
10%	Un ménage est en situation de précarité énergétique si ses dépenses énergétiques dépassent 10 % de son revenu.	Italie, Irlande du Nord	Simple à appliquer et facile à communiquer.	N'exclut pas les ménages avec de hauts revenus.	Department for Communities
2M (Double Médiane)	Un ménage est en situation de précarité énergétique si la part de son revenu utilisé pour ses dépenses énergétiques dépassent 2x la médiane de l'échantillon.		Norme largement reconnue.	N'exclut pas les ménages avec de hauts revenus.	Kantamneni and Haley 2021
LIHC (Low Income High Costs)	Définit la précarité énergétique comme la combinaison de coûts énergétiques élevés et d'un faible revenu.	Royaume-Uni	Prend à la fois en compte un seuil de revenu et de dépenses d'électricité	Ne prend pas en compte le "heat or eat" effect	Gouvernement du Royaume-Uni 2020
MIS (Minimum Income Standard)	Un ménage est considéré comme en dessous du seuil de précarité énergétique si son revenu après dépenses énergétiques est inférieur au revenu minimum standard.		Prend en compte d'autres aspects des dépenses d'un ménage (loyer, nécessités)	Peut être difficile à calculer (nécessite des études qualitatives autant que quantitatives)	Moore 2012

3.1.3. Indices composites : pourquoi parler de vulnérabilité énergétique ?

Une critique de plus en plus répandue à propos du terme “précarité énergétique” concerne son absence de flexibilité. L’expression, malgré ses nombreuses définitions, manque d’insister sur le caractère changeant de la précarité énergétique et le terme de “vulnérabilité énergétique” permet de penser la précarité énergétique sous un autre angle.

Le concept de vulnérabilité au sens large désigne pour un individu ou un ménage le degré de susceptibilité à un préjudice qui ne serait pas suffisamment contrebalancé par ses capacités à résister aux impacts négatifs de ce préjudice sur le moyen/long terme tout en maintenant un niveau de bien-être général satisfaisant (Allen 2003).

Ainsi, la précarité énergétique est un état de préjudice et la vulnérabilité énergétique est une situation de risque élevé de précarité énergétique d'un ménage et/ou sa fragilité potentielle aux conséquences d'une situation de précarité énergétique (Allen 2003).

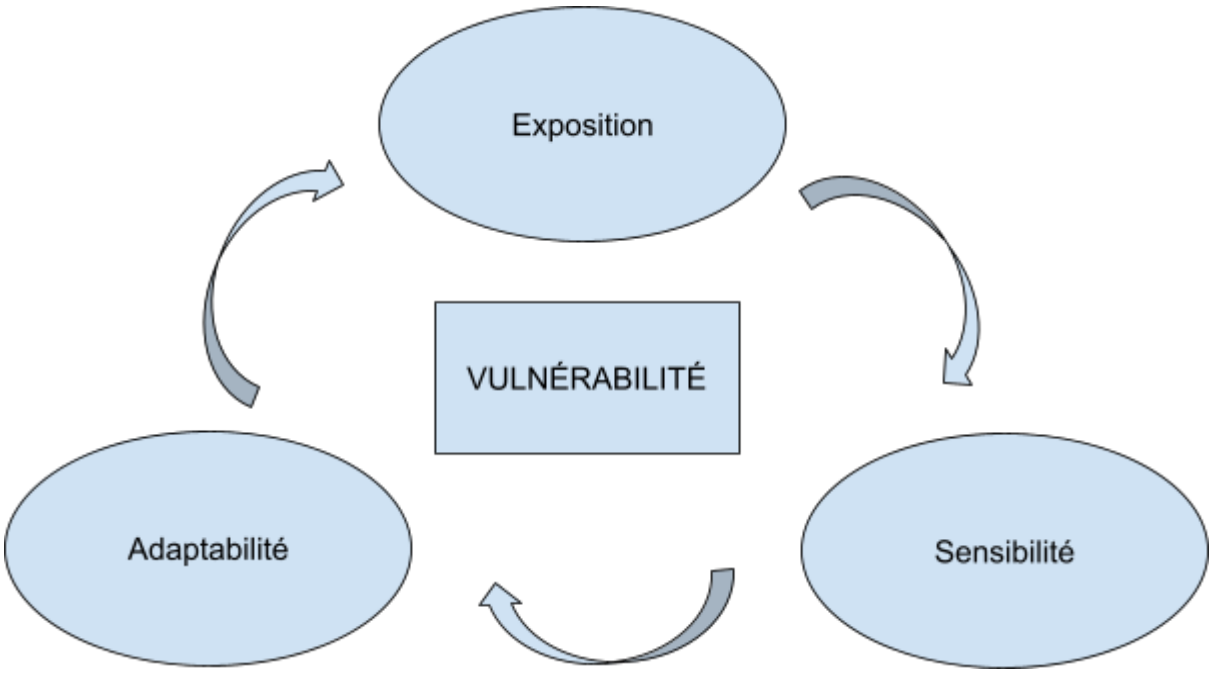
En s'intéressant au concept de vulnérabilité, on accède à des dimensions supplémentaires pour penser et évaluer la précarité énergétique.

En effet, un ménage peut en être affecté en fonction de trois critères :

- le risque qu'il soit soumis à la précarité énergétique
- sa capacité d'adaptation dans un cas de précarité énergétique
- sa sensibilité à une situation de précarité énergétique

Dans notre cas, en s'appuyant sur les définitions admises de ces termes dans les études sur la vulnérabilité (Buse et al. 2023, Mein et al. 2020), nous définissons le risque comme tous les facteurs qui pourraient favoriser une situation de précarité énergétique pour le ménage.
La capacité d'adaptation fait ainsi référence pour nous aux moyens, outils et recours qu'ont les ménages pour s'adapter à un mode de vie affecté par les facteurs à risque de précarité énergétique.
La sensibilité est le degré différentiel selon lequel les ménages au Québec peuvent être affectés par la précarité énergétique.Elle est influencée par des facteurs tels que la physiologie, la biologie, le patrimoine génétique, l'accès aux ressources et services de santé, le genre, l'âge et les pratiques de santé personnelles.

Schéma : Les trois composantes de la vulnérabilité



Étant donnée la nature dynamique de ces concepts, on comprend que la vulnérabilité énergétique d'un ménage sur une période donnée est sujette au changement et multidimensionnelle : elle comprend le risque d'arriver à une situation de précarité énergétique pour un ménage, la capacité de ce ménage à s'adapter aux changements liés à des facteurs de risque de la précarité énergétique et la sensibilité de ce ménage à une situation de précarité énergétique.
Une telle analyse de la vulnérabilité énergétique suggère que différents ménages présentent des degrés de vulnérabilité variés, en fonction de leur exposition, de leur sensibilité et de leur capacité d'adaptation.

Pour notre étude, nous veillerons à prendre en compte ces dimensions de la vulnérabilité afin de proposer un indicateur multidimensionnel qui cherche à capturer la complexité de la précarité énergétique au Québec en s'appuyant sur les recherches effectuées ailleurs dans le monde. Pour ce faire, nous explorerons plusieurs types d'indicateurs qui intègrent ces diverses dimensions.

Parmi les indicateurs composites que nous avons rencontré durant nos lectures, on retrouve des indicateurs dits "consensuels" ou "auto-déclarés" qui visent à prendre en compte la dimension sociale de la vulnérabilité énergétique en faisant appel chez les répondants à ce qu'ils considèrent "standard" (Riva 2024).

Par exemple, l'enquête EU-SILC ("EU statistics on income and living conditions - Eurostat") est un outil majeur pour mesurer la précarité énergétique dans les pays européens. Ce sondage à l'échelle européenne fournit des données autodéclarées sur les conditions de vie, y compris l'accès à l'énergie des ménages européens.
Les indicateurs principaux de la précarité énergétique utilisés par l'EPAH à partir de ce sondage comprennent d'une part, le pourcentage de la population incapable de chauffer adéquatement son logement et ayant des factures de services publics impayées et de l'autre, les indicateurs secondaires qui ne mesurent pas directement le problème, mais fournissent des informations sur divers aspects de la précarité énergétique comme le pourcentage de la population vivant avec des fuites, de l'humidité ou de la moisissure dans leur résidence (Al Kez et al. 2024).
Bien que la méthode consensuelle soit recommandée dans plusieurs études (Thomson et Snell 2013; Healy 2017; Healy et Clinch 2002; Petrova et al. 2013; Riva 2024), il est important de noter que les indicateurs utilisés dans cette approche sont subjectifs et que leur précision peut dépendre de la façon dont les questions sont interprétées par les répondants.

L'approche consensuelle présente donc des limitations, notamment la subjectivité des indicateurs qui peut entraîner des erreurs d'exclusion, où des ménages ne se reconnaissent pas comme en précarité énergétique (McKay 2004). De plus, la définition de la "chaleur adéquate" varie culturellement. Une étude de Palmer et al. a montré en 2008 un chevauchement limité entre les mesures consensuelles et celles basées sur les dépenses, avec seulement 6 % des ménages en précarité énergétique déclarant des difficultés de chauffage et près d'un tiers des répondants ayant déclaré avoir du mal à chauffer leur pièce-à-vivre qui avaient un revenu moyen ou supérieur à la moyenne.
Par conséquent, bien que la méthode consensuelle puisse fournir des informations utiles sur la prévalence de la précarité énergétique, les estimations basées sur des questions

subjectives doivent être interprétées avec prudence et peuvent ne pas toujours être entièrement précises (Thomson, Bouzarovski, et Snell 2017).

On retrouve aussi les indicateurs composites qui utilisent plusieurs des indicateurs que nous avons mentionnés plus tôt. Par exemple, Sokolowski et al. (2020) présentent l'Index de précarité Énergétique Multidimensionnel qui combine cinq indicateurs : deux indicateurs objectifs, « low income, high costs » (faible revenu, coûts élevés) et « high actual costs » (coûts réels élevés) et trois indicateurs consensuels, à savoir « not warm enough home » (c'est-à-dire incapacité à garder son logement suffisamment chaud), « housing faults » (défauts du logement) et « bill difficulties » (incapacité à payer les factures). L'étude considère qu'un ménage souffre de précarité énergétique si cela se reflète par les valeurs d'au moins deux indicateurs.

Enfin, les études plus récentes tendent à incorporer d'autres dimension de la précarité énergétiques dans leurs études comme les performances énergétiques du logement (Gouveia, Palma, et Simoes 2019) ou des caractéristiques géographiques comme dans *La précarité énergétique au Canada : une question d'inégalités sociales et géographiques* (2024) où Riva et Al. examinent le regroupement spatial de la précarité énergétique au sein des municipalités au Canada.

D'autre part, on retrouve aussi des méthodes de plus en plus complexes pour détecter la précarité énergétique.
Ainsi, Llera-Sastresa et al. (2017) ont développé l'Index Composite de Vulnérabilité Énergétique, conçu pour améliorer la gestion des logements sociaux. Les auteurs utilisent vingt indicateurs différents, qui peuvent être regroupés en quatre classes : caractéristiques du logement, caractéristiques et efficacité énergétique des appareils, coûts de l'énergie et habitudes, et caractéristiques des occupants. Il utilisent le Processus Hiérarchique Analytique, un outil de prise de décision multicritères qui leur a permis d'évaluer les facteurs qui déterminent la précarité énergétique dans les logements sociaux.
De même, Pino-Mejías et al. (2018) ont développé le "Fuel Poverty Potential Risk Index" (l'Index du Risque Potentiel de précarité Énergétique) qui s'appuie sur une méthodologie basée sur des calculs mathématiques et statistiques complexes. En utilisant des régressions linéaires multiples et des réseaux de neurones artificiels, ils ont cherché à évaluer le risque que les habitants des logements sociaux entrent en situation de précarité énergétique.

Afin de voir plus d'indices composites, voir cette étude : Siksnelyte-Butkiene et al. 2021.

3.2. Quels indicateurs utiliser pour la précarité et la vulnérabilité énergétiques dans un contexte québécois ?

3.2.1. L'indicateur 2M comme point de départ

Selon le Recensement 2021 de Statistique Canada, 4,2% des ménages au Québec (5,6 % au Canada) étaient en situation de précarité énergétique en utilisant le seuil de 10 % ou plus de leur revenu après impôts était consacré aux factures d'énergie liées au logement.

Le “seuil des 10%” est un indicateur historique encore utilisé dans énormément d’études comme celles de Statistique Canada sur la précarité énergétique au Canada (Dionne, Heisel, and Situ 2024) mais comme nous l’avons mentionné plus tôt, il est particulièrement arbitraire car n’évolue pas avec les particularités de l’étude.

Pour appliquer une approche similaire dans sa philosophie, on peut sélectionner l’indicateur 2M, lui aussi présenté plus tôt et très prisé pour sa facilité d’analyse.
Pour rappel, la mesure 2M est un indicateur de précarité énergétique qui évalue les ménages selon le pourcentage de leur revenu consacré aux dépenses énergétiques. Plus précisément, un ménage est considéré en situation de précarité énergétique si ses dépenses énergétiques dépassent un seuil déterminé de son revenu, généralement fixé à deux fois la médiane du pourcentage de revenu consacré aux dépenses énergétiques des ménages de l'échantillon.
Avec cette définition, on aboutirait à un seuil compris entre 6 et 7 % du revenu net des ménages réservé aux dépenses énergétiques au Canada et 5 et 6% au Québec au-delà duquel on peut dire qu'un ménage supporte des charges énergétiques plus élevées selon l’étude de Rezaei (2017) qui s’appuie sur l’Enquête sur les Dépenses des Ménages de Statistiques Canada.
Suite à notre méthodologie et l’obtention de nos données, nous pourrons choisir plus précisément un indicateur 2M adapté à notre échantillon des municipalités.

Néanmoins, choisir cet indicateur permet seulement d’avoir un point de départ pour notre étude car même s’il offre un seuil clair, il reste arbitraire (Castaño-Rosa et al. 2019) pour définir la précarité énergétique. En effet, bien que facile à comprendre et à communiquer, il ne prend pas en compte le fait que certains ménages consomment moins d’énergie que ce qu’ils devraient pour limiter leurs dépenses ou encore que d’autres ménages à revenus élevés aient des logements avec une faible efficacité énergétique ce qui les placerait au-dessus du seuil.

Dès lors, pour arriver à englober toute la complexité de la vulnérabilité énergétique, de nombreuses autres variables ont été proposées durant nos lectures qui pourraient servir à notre modèle et que nous allons distinguer en trois groupes en s’appuyant sur notre définition de la vulnérabilité énergétique (exposition, sensibilité, capacité d’atténuation) et que Germany’s Vulnerability Network a théorisé en 2017 dans son Vulnerability Sourcebook pour le changement climatique.

La sélection de nos variables a été effectuée grâce à une revue de littérature qui nous a permis de sélectionner les variables auxquels les différents auteurs semblaient accorder le plus d’importance, que ce soit dans leurs analyses théoriques ou l’élaboration de leur index composite. Elles pourront être retrouvées dans la section “Méthodologie” de ce mémoire.

3.2.2. Méthode pour notre indice composite

Notre indice a été modélisé en nous appuyant sur la méthode développée par Sondès Kahouli et Shinichiro Okushima en 2021. Cette méthode repose sur l'idée que la précarité

énergétique ne peut être appréhendée à travers un seul indicateur, mais s'appuie plutôt sur l'interaction de différentes dimensions influençant l'accès à l'énergie. Les auteurs font l'hypothèse que les besoins énergétiques des ménages varient selon leurs caractéristiques, rendant ainsi les seuils de précarité énergétique relatifs et non absolus. Pour cette raison, ils ont d'abord regroupé les ménages d'intérêt en groupes en fonction de facteurs climatiques, du caractère détaché ou pas de leur logement et s'ils incluaient des personnes de plus de 65 ans. Ensuite, ils ont déterminé les consommations d'énergie moyenne pour chacun de ces groupes et classé comme en situation de précarité énergétique les ménages qui consommaient moins de 60% de la médiane d'énergie de leur groupe. Pour ne prendre en compte que les ménages les plus précaires, ils ont aussi ajouté une nécessité pour les ménages de faire partie des 30% les plus pauvres de tout l'échantillon.
L'idée est donc d'un indicateur de type "seuil physique" qui met en avant les ménages restreignant leur consommation d'électricité afin de repérer ceux qui sont en situation de "heat or eat".

En nous inspirant de la méthodologie d'Okushima, nous avons adapté notre approche en utilisant un code R pour déterminer automatiquement les zones et les types de ménages. En utilisant la librairie neighbr et les variables de la vulnérabilité énergétique que nous allons sélectionnées, nous souhaitons appliquer une approche d'analyse de type k-NN (k-voyageurs les plus proches) pour identifier les ménages en fonction de leurs caractéristiques spécifiques.
Nous avons également établi des seuils pour la précarité énergétique, qui diffèrent de ceux d'Okushima. Dans notre méthodologie, nous définissons le seuil de consommation énergétique (c_k) comme 80 % de la consommation médiane d'énergie parmi les ménages similaires dans une zone donnée car les prix de l'énergie étant très bas au Québec, on suppose qu'une privation, même plus faible peut être le signe de difficultés à payer ses factures d'énergie pour un ménage. Pour le seuil de revenu (r), nous faisons référence au 40e percentile du revenu médian au lieu du 30e percentile, permettant ainsi une classification qui s'adapte mieux aux réalités économiques locales.

Pour illustrer ce concept, considérons un exemple basé sur notre méthodologie.
Nous définissons le seuil de consommation énergétique (c_k) comme 80 % de la consommation médiane d'énergie parmi les ménages semblables dans une zone donnée. Prenons, par exemple, deux groupes de municipalités au Québec : groupe A et groupe B.
Pour le groupe A de municipalités, la médiane de la consommation moyenne d'énergie pour un ménage est de 300 kWh par mois. Dans ce cas, le seuil c_k serait de 240 kWh (80 % de 300 kWh). Ainsi, une municipalité du groupe A dont le ménage moyen consomme moins de 240 kWh par mois serait considéré comme en situation de précarité énergétique, indiquant que ses ménages restreignent leur consommation pour faire face à leurs dépenses.
Pour le groupe B, imaginons que la consommation médiane d'énergie soit plus élevée car le climat y est plus froid et que cette consommation atteigne 500 kWh par mois. Dans ce cas, le seuil c_k pour le groupe B serait de 400 kWh (80 % de 500 kWh). Une municipalité du groupe B qui consomme moins de 400 kWh serait également classée comme en situation de précarité énergétique.

Cette comparaison souligne la relativité des seuils de précarité énergétique avec cet indice composite en fonction des contextes locaux. Par exemple, une municipalité du groupe A qui

consomme 350 kWh par mois ne serait pas considérée comme une zone de précarité énergétique, car elle est au dessus du seuil de 240 kWh, tandis qu'une municipalité du groupe B qui consomme 375 kWh serait considérée comme zone de précarité énergétique, car cela reste en dessous du seuil de 400 kWh.

Pour compléter cet exemple, considérons également le seuil de revenu (r). Pour notre étude, nous définissons ce seuil comme le 40e percentile du revenu médian pour l'ensemble de l'échantillon de municipalités. Par exemple, si le 40e percentile de revenu pour l'ensemble des ménages toutes municipalités de notre échantillon confondues est de 40 000 $, alors les municipalité dont le revenu annuel médian des ménages serait supérieur à 40 000 $ ne pourraient pas être considérées comme en situation de précarité énergétique, peu importe leur consommation d'électricité.

Les résultats ainsi obtenus avec cette méthode seront comparés à celle des 2M.

Collecte et Analyse des Données : Modélisation des Indicateurs de Précarité Énergétique

4. Méthodologie

Afin de comprendre la situation au Québec et d'analyser les interactions des différents facteurs potentiels de précarité énergétique, nous avons procédé à l'analyse de données en utilisant Python et R et créé plusieurs visualisations grâce à l'aide de Tableau (voir Annexe 1 pour le code).
Nous expliquons ici la méthodologie employée afin d'obtenir ces données.

4.1. Municipalités concernées

Tout d'abord, il convient de préciser que les municipalités de la région du Nord-du-Québec ont été exclues de cette analyse en raison des différences significatives dans leur paysage énergétique, notamment parce que de nombreuses communautés ne sont pas reliées au réseau électrique principal.
De plus, la ville de Montréal a été omise de nos analyses en raison de sa très grande taille (beaucoup plus grande que toutes les autres municipalités étudiées) et de la diversité et densité de population, qui nécessitent une attention particulière dans une étude séparée.
Finalement, les municipalités où le revenu médian des ménages était supérieur à 90 000 dollars par an ont été exclues en raison de leur faible représentation dans notre étude.

4.2. Présentation des bases de données

4.2.1. Base de données Hydro-Québec

La base de données Hydro-Québec nous fournit des informations sur la consommation d'électricité dans les municipalités du Québec. Cette base de données contient des informations sur la consommation mensuelle d'électricité, agrégée par secteur de consommation (résidentiel, commercial, institutionnel, industriel et agricole) pour chaque municipalité du Québec, de 2016 à 2023.
Nous avons sélectionné les données des utilisateurs résidentiels pour la période de 2016 à 2023. Elles incluent des informations essentielles telles que :

- Les noms, régions et MRC des municipalités : Chaque municipalité est associée à un ensemble de zones géographiques permettant de relier les données à des municipalités spécifiques.
- La consommation mensuelle d'électricité : Les données incluent la consommation d'électricité mensuelle en kilowattheures (kWh) pour chaque municipalité, nous

permettant d'analyser les variations saisonnières de la consommation d'électricité des ménages

Il est important de noter que, bien que la base comporte un total de 1 128 municipalités, 485 d'entre elles ne disposent d'aucune donnée de consommation en raison de préoccupations liées à l'anonymat.
En conséquence, notre étude s'est concentrée sur deux ensembles de données : l'un sur les 199 municipalités pour lesquelles nous avons des valeurs complètes et l'autre basée sur une imputation des valeurs manquantes pour atteindre un total de 364 municipalités.
Cela nous permet d'avoir une vue d'ensemble de la consommation d'électricité au Québec, tout en tenant compte des disparités entre les municipalités à différents niveaux de vulnérabilité énergétique.
La capacité à segmenter ces données en fonction des mois les plus froids et des mois non-hivernaux nous permettra de comparer les comportements de consommation énergétique et d'étudier l'impact des conditions climatiques sur la précarité énergétique.

4.2.2. Base de données de Statistique Canada

Nous avons également intégré des données démographiques et économiques issues des recensements de Statistique Canada réalisés en 2016 et 2021. Ces bases de données nous fournissent un large éventail d'informations sur les caractéristiques des populations résidant dans les municipalités québécoises.

Les données concernent notamment :

- La démographie avec des informations concernant la population totale de chaque municipalité, la répartition par âge, le sexe, et d'autres indicateurs démographiques.

- Les statistiques sur les revenus des ménages

- Des données sur la répartition des types de logements, ainsi que sur les caractéristiques des logements eux-mêmes, telles que l'ancienneté et la gravité des réparations nécessaires

Pour la faire correspondre à notre base de données Hydro-Québec, nous avons sélectionné uniquement les informations sur les municipalités qui s'y trouvaient bien qu'on ait accès à des zones géographiques plus précises comme les subdivisions de recensement.

4.2.3. Données sur le climat - Bibliothèque Meteostat

Meteostat est une bibliothèque qui permet d'accéder à des ensembles de données historiques sur les conditions météorologiques collectées par des stations météo à travers le monde.

Meteostat utilise une API pour fournir des données météorologiques accessibles à partir de serveurs. En utilisant cette API, les utilisateurs peuvent faire des requêtes pour récupérer

des données spécifiques selon différents critères, tels que la localisation géographique, le type de données souhaitées (températures, précipitations, etc.) et la période de temps.
Chaque station météorologique est associée à des coordonnées géographiques spécifiques. Pour récupérer les données pertinentes pour une municipalité donnée, le système identifie les stations situées à proximité ainsi que leurs identifiants uniques.

Les données extraites comprennent plusieurs variables climatiques clés, telles que les températures moyennes sur une période donnée.
En intégrant ces données climatiques dans notre étude, nous sommes en mesure d'évaluer de manière plus précise comment les variations des conditions météorologiques pourraient affecter la précarité énergétique.

4.2.4. Préparation des codes géographiques des municipalités

Pour faciliter le lien entre les bases de données, nous avons utilisé un fichier contenant les codes géographiques des municipalités, accessible via le site du gouvernement du Québec . Ce fichier est essentiel pour standardiser les identifiants des municipalités et assurer la cohérence des analyses.

Les étapes suivantes ont été effectuées afin d'utiliser cette base de données :

- Ajout du préfixe "24" à chaque code municipal pour qu'ils correspondent aux formats utilisés dans les données des recensements de Statistique Canada.
- Correction des noms de régions administratives en supprimant les codes entre parenthèses afin de ne conserver que le nom de la région.
- Sélection des colonnes pertinentes (mcode, munnom, mdes, regadm, mcodpos) pour l'analyse.

Cette préparation assure la compatibilité entre les bases de données d'Hydro-Québec et celles de Statistique Canada pour la fusion ultérieure des deux bases de données.

4.2.5. Données géographiques sur les municipalités

Nous avons également procédé au géocodage des municipalités en utilisant des données géographiques disponibles sur le site des Ressources naturelles du Canada. Ces données contiennent les coordonnées géographiques des municipalités du Québec et permettent d'associer les coordonnées géographiques aux municipalités, essentielles pour la création de visualisations cartographiques et obtenir les informations climatiques récupérées grâce à Meteostat.

Comme plusieurs points de latitude et de longitude sont fournis pour chaque municipalité, nous avons choisi de ne conserver que la première ligne de coordonnées pour chaque entité pour avoir des coordonnées uniques.

Cependant, il manquait les informations pour 16 municipalités. Nous avons donc apporté des corrections manuelles en ajustant les noms ou en ajoutant des caractères manquants, tels que les accents ou les traits d'union. Pour les cas plus complexes, nous avons utilisé Google Maps pour récupérer les coordonnées géographiques manquantes.

4.3. Nettoyage de la base de données d'Hydro Québec

4.3.1. Imputation

Pour maximiser la justesse de nos imputations, nous avons sélectionné toutes les municipalités qui avaient au moins cinq valeurs de consommations d'électricité (sur les 96 possibles de 2016 à 2023) et au moins une durant les mois les plus froids (de novembre à mars) ainsi qu'au moins une durant les autres mois soit 364 municipalités.

L'imputation des valeurs manquantes dans nos données de consommation d'électricité a été réalisée en plusieurs étapes et a été modélisée en fonction des tendances pour chaque région, observées grâce aux données des municipalités pour lesquelles ont avait accès à toutes les données de consommation.

Nous avons choisi un modèle sinusoïdal à trois sinusoïdes dans le cas de notre étude pour plusieurs raisons spécifiques.
Premièrement, la consommation d'électricité au Québec est fortement influencée par les saisons, surtout avec des hivers rigoureux qui entraînent une augmentation des besoins de chauffage. Un modèle à trois sinusoïdes nous permet de capter les variations de consommation sur des cycles annuels tout en tenant compte des comportements pratiques observés chez les consommateurs durant les mois les plus froids (novembre à mars) par rapport aux périodes plus tempérées (avril à octobre).
De plus, les données de consommation d'électricité peuvent refléter non seulement les saisons, mais aussi d'autres effets périodiques, tels que les variations hebdomadaires et les changements dans le comportement des ménages. En utilisant trois sinusoïdes, nous sommes en mesure de modéliser ces différents motifs.
Avant de sélectionner ce modèle, des analyses préliminaires ont montré que les données de consommation suivaient des schémas sinusoïdaux historiques, particulièrement lors des analyses des consommations de l'an dernier dans les mêmes périodes (voir annexe 3). L'approche à trois sinusoïdes a prouvé qu'elle était efficace pour capturer ces tendances, comme en témoignent les bonnes performances de prédiction lors de tests.
Le modèle sinusoïdal est aussi relativement facile à interpréter. Les coefficients des sinusoïdes (amplitudes, fréquences, phases) permettent de comprendre comment et pourquoi la consommation change sur certains intervalles de temps. Cela peut être très informatif pour des discussions sur la précarité énergétique et les implications politiques autour de l'utilisation de l'énergie.

De plus, nous avons remarqué un pic général de consommation d'électricité en janvier 2022 que nous avons manuellement ajouté pour chaque municipalité en fonction de la tendance de sa région.

L'imputation des valeurs manquantes dans nos données de consommation d'électricité a été réalisée en plusieurs étapes.
Nous avons commencé par préparer les données en extrayant et nettoyant les données de consommation d'électricité d'Hydro-Québec. Seules les municipalités avec toutes leurs valeurs de consommation non nulles ont été conservées pour modéliser notre équation de courbe afin d'éviter d'introduire des biais dans notre analyse.
Les données de consommation ont été normalisées pour chaque municipalité en fonction de ses données de consommation. Cette transformation a permis de limiter les valeurs entre 0 et 1, simplifiant ainsi l'application d'analyses statistiques et le tracé des données.
Nous avons ensuite utilisé la fonction "curve_fit" de la bibliothèque SciPy afin de déterminer l'équation de la courbe de consommation d'électricité pour chaque région. Cette méthode offre un moyen efficace d'estimer les paramètres d'une fonction en minimisant la différence entre les valeurs observées et les valeurs prédites. Pour cela, nous avons d'abord défini une fonction modèle, un triple sinus, qui représente les variations saisonnières de la consommation. La fonction est formulée pour inclure des termes d'amplitude, de fréquence et de phase, permettant de capturer les oscillations observées.
En appelant curve_fit, nous avons fourni notre fonction modèle ainsi que les données (valeurs de consommation et de temps). Nous avons également inclus des estimations initiales pour les paramètres afin de faciliter la convergence de l'algorithme d'optimisation.
La fonction utilise un algorithme d'optimisation pour ajuster les paramètres de la fonction de manière à minimiser la somme des carrés des écarts entre les valeurs observées (données réelles) et les valeurs prédites par le modèle (voir annexe pour toutes les équations de consommation par région).

Après avoir prédit les valeurs normalisées pour les municipalités et obtenu ainsi les équations de courbe de consommation d'électricité pour chaque région, un processus de normalisation puis de dénormalisation a été appliqué sur les données des municipalités aux valeurs manquantes pour ramener les valeurs à leur échelle d'origine (kWh) après avoir obtenu les valeurs imputées grâce aux équations de consommation normalisée de leurs régions respectives.
Enfin, nous avons évalué la qualité de nos imputations en calculant l'erreur absolue moyenne entre les valeurs imputées et les valeurs réelles disponibles. Pour cela, nous avons simplement utilisé notre équation d'imputation sinusoïdale et l'avons appliquée à nos municipalités sans valeurs manquantes puis avons comparé les résultats à leurs valeurs réelles. Le pourcentage d'erreur a également été calculé pour offrir une évaluation contextuelle des imputations. Des visualisations (voir Annexe 2) ont été réalisées pour comparer les courbes d'imputation et de consommation réelle pour plusieurs municipalités, permettant de vérifier visuellement la concordance des tendances.
Les résultats de l'imputation des valeurs manquantes révèlent une erreur absolue moyenne de 701205,69 kWh, correspondant à une erreur moyenne en pourcentage de 8,20 %, ce qui indique un niveau d'imputation relativement acceptable dans le contexte de notre analyse.

En utilisant cette approche combinée, nous avons pu estimer les valeurs manquantes dans notre base de données de consommation d'énergie, ce qui nous permet de mener une analyse plus approfondie de la précarité énergétique des ménages au Québec.

L'étape suivante a été réalisée sur les municipalités avec toutes les données de consommation ainsi que sur l'union de ces dernières et des municipalités aux données de consommation imputées.

4.3.2. Données obtenues

Les données de consommation ont été fusionnées avec les codes géographiques pour associer les informations de consommation aux données géographiques des municipalités. Cette fusion a été effectuée en utilisant les noms des municipalités et leur région administrative.

Des erreurs dans la fusion ont été corrigées, notamment pour des municipalités comme Courcelles, pour lesquelles les codes géographiques manquants ont été ajoutés manuellement. De même, les noms des municipalités ont été mis à jour pour refléter les noms actuels et les codes postaux ont été ajoutés.

Grâce à ces données, nous avons pu calculer la moyenne de consommation d'électricité en par municipalité pour :

- la consommation d'électricité les mois les plus froids au Québec (novembre, décembre, janvier, février, mars)
- la consommation d'électricité le reste du temps

4.4. Nettoyage des bases de données de Statistique Canada et association avec les données d'Hydro-Québec

Les données ont été filtrées pour ne conserver que celles correspondant à des codes géographiques de 7 chiffres (les municipalités), soit 3,372,942 lignes. Ensuite, les noms de municipalités ont été nettoyés pour éliminer les informations superflues. Cette opération a permis de se concentrer sur les noms principaux des municipalités

Ensuite, afin de fusionner les bases de données de 2016 et 2021, les colonnes en commun ont été sélectionnées et leur moyenne a été calculée sur les deux années. Les colonnes présentes seulement dans une des deux bases ont été conservées lorsqu'elles avaient un intérêt (ex. : Population en 2021).

Finalement, en utilisant les codes géographiques comme clé unique, cette base de données finale des variables des census a été fusionnée avec celle des consommations d'électricité obtenue plus tôt.

4.5. Obtention des données climatiques

Pour enrichir l'analyse de la vulnérabilité énergétique des ménages, nous avons intégré des données climatiques à partir de la bibliothèque Meteostat dans Python.

Une fois les données climatiques intégrées, nous avons calculé les moyennes mensuelles des températures, puis regroupé ces données pour obtenir des statistiques spécifiques à la saison hivernale (les mois de novembre, décembre, janvier, février et mars) ainsi qu'à la saison non hivernale.

Pour la saison hivernale d'une part et pour les autres mois de l'année de l'autre, nous avons calculé la température moyenne.

Grâce à cette base de données finale, nous avons accès à plusieurs variables qui nous ont permis d'étudier les caractéristiques de la précarité énergétique au Québec.

4.6. Sélection des variables

Grâce aux bases de données précédentes, nous avons sélectionné les variables adaptées à notre structure de la vulnérabilité : l'exposition, l'adaptabilité et la sensibilité.
Pour l'exposition et l'adaptabilité, nous avons choisi d'effectuer des régressions linéaires pour bien confirmer l'influence de nos variables sur la précarité énergétique.
A noter que toutes les régressions suivantes sont construites à partir des données des municipalités sans valeurs imputées.
De plus, toutes les régressions linéaires et les modèles de cette partie sont calculés pour les ménages dont le revenu annuel est inférieur à 90 000 pour avoir des résultats représentatifs.

4.6.1. Variables qui évaluent le risque d'exposition

L'efficacité énergétique du logement est l'un des premiers facteurs cités de la précarité énergétique après les hautes factures d'électricité et le faible revenu et on le retrouve dans de nombreux indices composites publiés (Boardman 1991, Llera-Sastresa et al. 2017, Scarpellini et al. 2015, Riva 2024).
Afin de vivre dans des logements décents et confortables, les ménages ont besoin de services énergétiques de base (chauffage, éclairage, eau chaude, etc.) qui dépendent énormément des caractéristiques du logement, que ce soit son état ou sa période de construction qui peuvent affecter son isolation par exemple (Chen et Feng 2022, Nojedehi et al. 2024). En effet, les personnes touchées par la précarité énergétique ont tendance à vivre dans des logements plus anciens de moins bonne qualité, qui ont pu être construits avec des niveaux d'isolation inférieurs et des taux d'infiltration d'air plus élevés, et qui n'ont peut-être pas été correctement entretenus (Rezaei 2017).

Lors de nos analyses de l'efficacité énergétique des logements québécois, nous avions ainsi à disposition le pourcentage des logements des municipalités par date de construction (avant 1960 puis de 1961 à nos jours par incrément de 20 ans) ainsi que le pourcentage de logements dans la municipalité qui avaient des réparations urgentes requises.
Pour évaluer la corrélation entre la date de construction du logement et la proportion du revenu consacrée à l'énergie du ménage, nous avons regroupé les logements construits avant 1981 sur notre variable dépendante.

D’autre part, nous avons comparé le pourcentage de logements requérant des réparations majeures à la part de revenu consacrée à l’électricité au sein d’une municipalité.

Nous avons ainsi effectué des régressions linéaires robustes montrant que la date de construction (Annexe 4.a) ainsi que le pourcentage de bâtiments ayant besoin de réparations majeures (Annexe 4.b) étaient bien corrélés à l'augmentation de la part de revenus consacrée à l’énergie dans nos ménages et que dans les deux cas, plus ils augmentent, plus la part du revenu des ménages consacrée à l’électricité augmente significativement.
Pour le logement, nous avions seulement accès aux variables de Statistique Canada et nous sommes donc limités aux réparations et à la date de construction mais d’autres variables pour le logement possiblement corrélées à la précarité énergétique ont été relevées dans nos lectures comme la performance énergétique des appareils ménagers ou le type de chauffage utilisé (Llera-Sastresa et al. 2017).

Une autre variable d’exposition que nous avons choisi d’observer est le climat car il a aussi été énormément cité dans nos lectures par son impact sur les coûts en chauffage pour les ménages (Middlemiss 2022; Nord et Kantor 2006; Gouveia, Palma, et Simoes 2019).
Dans ce cas aussi, une régression linéaire robuste a montré une corrélation significative entre la température moyenne en hiver et la part du revenu consacré à l’électricité de nos ménages (Annexe 4.c). Ainsi, plus la température est basse dans une municipalité, plus la part du revenu des ménages consacrée à l’électricité augmente.

4.6.2. Variables qui évaluent la capacité d’adaptation

Les revenus faibles ou instables sont les premières variables citées lorsqu’on parle de précarité énergétique ou dans le calcul d’indicateurs de précarité énergétique (Gouveia, Palma, et Simoes 2019).
Ainsi, nous avons récupéré les revenus moyens après impôts des ménages des municipalités et avons effectué des régressions linéaires robustes montrant que le revenu médian était significativement corrélé à la part de revenu consacrée à l’électricité (Annexe 4.d). Plus il augmentait, plus la part du revenu allouée aux dépenses en électricité diminuait de manière significative.

4.6.3. Variables qui évaluent la sensibilité

Au-delà des aspects socio-économiques comme le revenu, il est essentiel de prendre en compte des facteurs qui augmentent la sensibilité des ménages à la précarité énergétique et dont l’impact n’est pas nécessairement celui de corrélation mais d’intensification des difficultés associés à une situation de précarité. Pour cette raison, nous avons choisi de ne pas effectuer de régression robuste linéaire pour ce type de variable.

L’âge est un facteur important de sensibilité dans la vulnérabilité énergétique, ne serait-ce que pour la santé plus fragile des personnes de 65 ans et plus. En effet, elles sont les plus à même de mourir lors des grands froids (National Institute on Aging 2024).

De plus, non seulement elles tendent à rester plus souvent à l'intérieur mais elles souffrent plus gravement des conséquences du froid ou d'une chaleur extrême. Si elles ne peuvent pas garder leur logement à une température confortable, l'impact de la précarité énergétique peut grandement les affecter.
En effet, la précarité énergétique peut affecter la santé par trois mécanismes principaux. Tout d'abord, un chauffage insuffisant peut entraîner des problèmes tels que la condensation, la moisissure, l'humidité (CMD), ainsi que le froid à l'intérieur, ce qui peut avoir des effets directs sur la santé. Ensuite, le fait de consacrer davantage de ressources financières à l'énergie peut réduire le budget disponible pour d'autres besoins essentiels, comme l'alimentation. Enfin, le stress provoqué par l'incapacité à régler les factures de chauffage ou par la présence visible de moisissure et d'humidité peut également nuire à la santé (De Vries et Blane 2013).

D'autres variables auraient pu être prises en compte dans cette catégorie telles que les familles monoparentales ou les personnes en situation de handicap (Middlemiss 2022) mais soit nous n'avions pas accès à ces variables soit le lien direct avec la sensibilité à la précarité énergétique était limité dans la littérature.

Tableau 2 : Récapitulatif des régressions linéaires robustes pour les variables sélectionnées dans notre modèle

Variables	Base de données	Estimation des régressions linéaires robustes	p-value des régressions linéaires robustes
Revenu	Statistique Canada	-1.63E-05	6.27E-08
Pourcentage date de construction avant 1981	Statistique Canada	0.008267	3.57E-05
Pourcentage de réparations majeures requises	Statistique Canada	0.03718	0.000267
Taux de personnes de plus de 65 ans	Statistique Canada	NA	NA
Température moyenne hiver	Statistique Canada	-0.05776	0.00699

Consommation d'électricité	Hydro Québec	NA	NA

Finalement, nous pouvons aussi retrouver dans le tableau qui suit (Tableau 3) les statistiques descriptives de nos variables d'intérêt pour les municipalités sans valeur manquante et l'échantillon de municipalités incluant des valeurs imputées.
A noter qu'on retrouvera des valeurs moyennes positives de température en hiver pour certaines municipalités car nous avons ciblé les mois de novembre à mars.

Tableau 3 : Statistiques descriptives pour les municipalités sans valeur manquante et avec valeurs imputées

Variable	Moyenne	Médiane	Ecart-Type	Minimum	Maximum
Population en 2021	12 281	2 038	45 445	277	54 9459
Population en 2021 (imputées)	12 308	2 323	39 457	285	43 8366
Revenu médian des ménages après impôts ($)	56 482	54 916	9 692	38 016	87 320
Revenu médian des ménages après impôts ($) (imputées)	56 632	55 200	10 204	38 016	87 320
Taille moyenne des ménages	2.25	2.25	0.21	1.75	2.85
Taille moyenne des ménages (imputées)	2.25	2.25	0.20	1.85	2.70
Revenus des ménages consacrés à l'électricité (%)	3.07	2.70	1.34	1.33	13.84
Revenus des ménages consacrés à l'électricité (%) (imputées)	3.02	2.69	1.33	1.33	13.84
Coût de l'électricité par ménage par an	1 712	1 525	755	742	9 171
Coût de l'électricité ($) par ménage par an (imputées)	1 693	1 520	793	832	9 171
Consommation d'électricité par ménage par an (kWh)	21 611	19 815	7 309	11 064	93 745
Consommation d'électricité par ménage par an (kWh) (imputées)	21 434	19 764	7 677	12 415	93 745
Logements construits avant 1981 (%)	58.58	59.91	14.17	18.54	86.21
Logements construits avant 1981 (%) (imputées)	57.89	58.63	15.13	18.54	86.21
Logements nécessitant des réparations majeures (%)	7.23	6.55	3.11	1.35	20.27

Logements nécessitant des réparations majeures (%) (imputées)	7.31	6.59	3.32	1.56	20.27
Température moyenne en hiver (°C)	-5.15	-5.21	1.92	-10.90	3.81
Température moyenne en hiver (°C) (imputées)	-5.18	-5.15	1.66	-10.89	3.18
Personnes âgées de plus de 65 ans (%)	12.99	12.61	3.89	3.15	27.87
Personnes âgées de plus de 65 ans (%) (imputées)	12.76	12.46	3.70	6.13	21.97

4.7. Détermination des k-voisins

Grâce aux variables sélectionnées, nous pouvons appliquer la méthode d'Okushima consistant à trouver les municipalités les plus ressemblantes à la nôtre (ses k-voisins) et déterminer une consommation nécessaire pour les ménages de chaque municipalité. En dessous de cette consommation ck et en dessous d'un seuil de revenu r défini, nous la considérerons comme pauvre énergétiquement

4.7.1. Chargement des librairies

Pour commencer, nous avons chargé les librairies nécessaires dans R. Nous avons utilisé neighbr pour appliquer l'algorithme des k-voisins les plus proches (k-NN).

La librairie neighbr est un package R conçu pour faciliter l'implémentation de l'algorithme des k-plus proches voisins, couramment appelé k-NN (k-Nearest Neighbors). Cette méthode est largement utilisée dans les domaines de l'apprentissage automatique et de l'analyse de données pour la classification et la régression, en se basant sur la proximité des observations.

4.7.2. Sélection des Variables et définition du nombre de voisins (k)

Plusieurs variables continues ont ensuite été sélectionnées pour évaluer la précarité énergétique au sein d'une municipalité :

- Le pourcentage de logement ayant des réparations majeures requises
- Pourcentage personnes âgées (>65 ans)
- Pourcentage de logement construits avant 1981
- Températures moyennes hivernales

En définissant le nombre de voisins (k), fixé à 10, nous avons utilisé la méthode k-NN pour identifier les municipalités similaires.

4.7.3. Détermination des seuils de précarité énergétique

Un aspect fondamental de notre approche est la détermination des seuils de précarité énergétique. Le seuil de consommation (c_k) a été sélectionné comme étant 80% de la consommation médiane des ménages comparables dans la même zone. Parallèlement, le seuil de revenu (r) a été établi au 40% des municipalités avec le plus bas revenu médian par ménage pour l'ensemble du pays.

Nous avons identifié les municipalités considérées comme sensibles à la précarité énergétique en vérifiant si leur consommation d'énergie était inférieure à c_k et si leur revenu médian était inférieur à r.

5. Résultats

Nous allons maintenant nous pencher sur les résultats de notre méthode des k-voisins pour déterminer la précarité énergétique des municipalités au Québec.

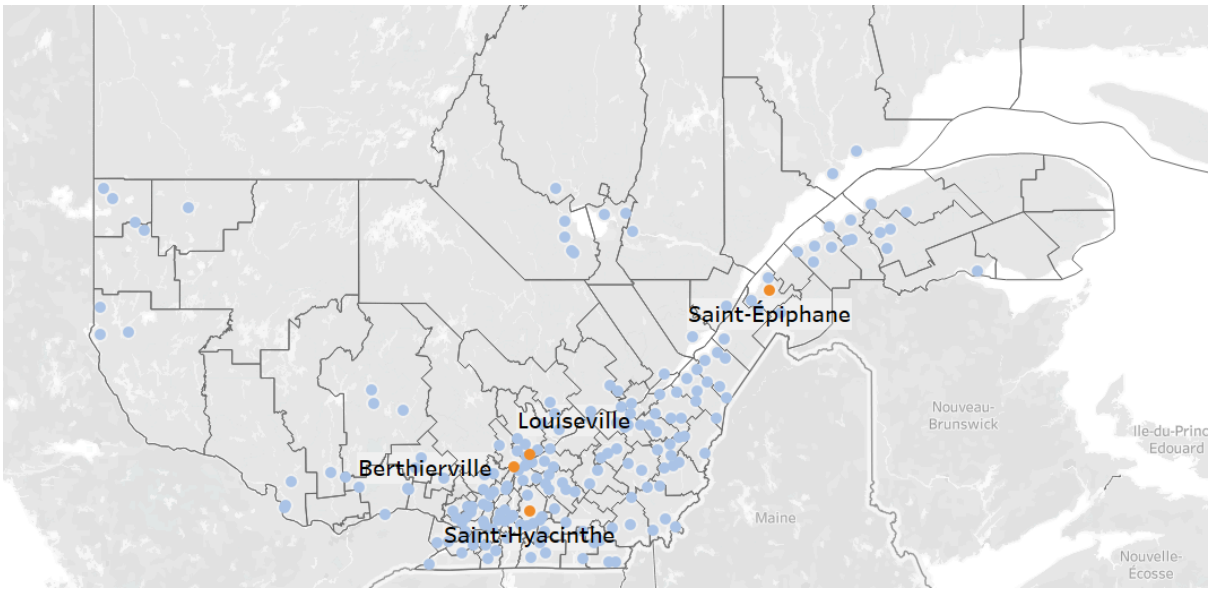
Pour la comparaison avec l'indicateur 2M, nous avons choisi un seuil de 5,38 % du revenu moyen des ménages consacré à l'électricité pour nos municipalités sans valeur manquante. Ce taux correspond à 5,4 % lorsque l'on inclut des valeurs imputées. Nous avons défini cet indicateur comme étant le double de la médiane du pourcentage des revenus des ménages après impôts consacrés à l'électricité.

5.1. Municipalités sans valeur manquante

5.1.1. Méthode des k-voisins

Dans l'échantillon de municipalités sans valeurs manquantes, quatre municipalités ont été identifiées comme étant en situation de précarité énergétique selon la méthode des k-voisins : Saint-Épiphanie, Louiseville, Berthierville et Saint-Hyacinthe.

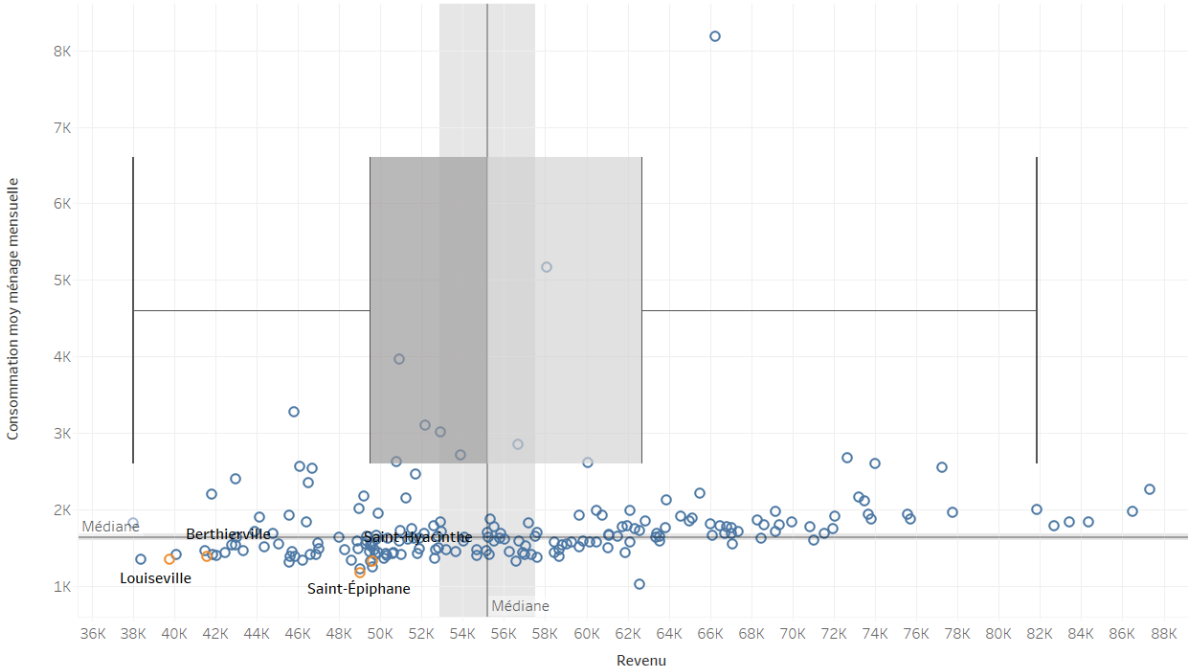
Carte 1 : Municipalités en situation de précarité énergétique identifiées grâce à la méthode des k-voisins pour les municipalités sans valeurs manquantes



Une observation générale à partir des graphiques suivants montre que les municipalités ayant un revenu par ménage supérieur à 64 000 $ par an après impôts se situent toutes au-dessus de la médiane en termes de consommation d'électricité.

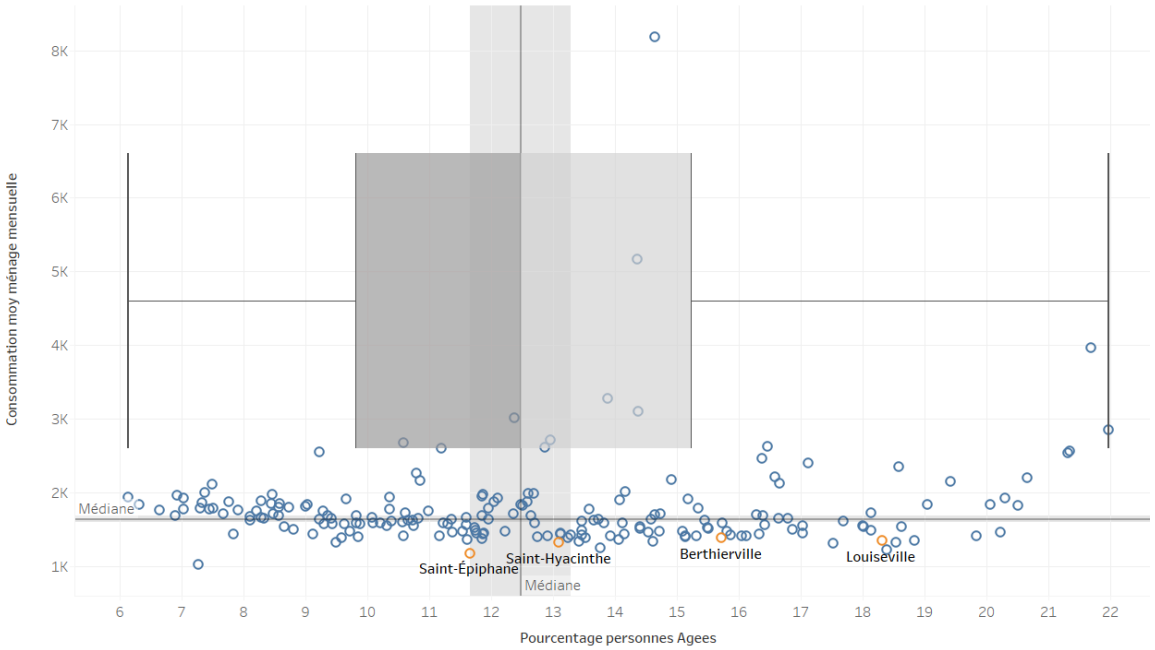
Concernant les municipalités identifiées comme étant en situation de précarité énergétique, bien qu'une exigence de revenus inférieure au 40e percentile ait été appliquée pour notre indicateur, deux de ces municipalités, Louiseville et Berthierville, affichent un revenu par ménage particulièrement bas, se classant respectivement au 3e et 6e rang des municipalités ayant les revenus médians les plus bas après impôt. En outre, les municipalités déclarées en situation de précarité énergétique consomment moins d'électricité que leurs voisines, ce qui se reflète dans leur statut parmi les municipalités ayant la plus faible consommation d'énergie par ménage au Québec.

Nuage de points 1 : Municipalités selon la consommation mensuelle moyenne d'électricité par ménage en kWh en fonction du revenu($) (méthode des k-voisins pour les municipalités sans valeurs manquantes)



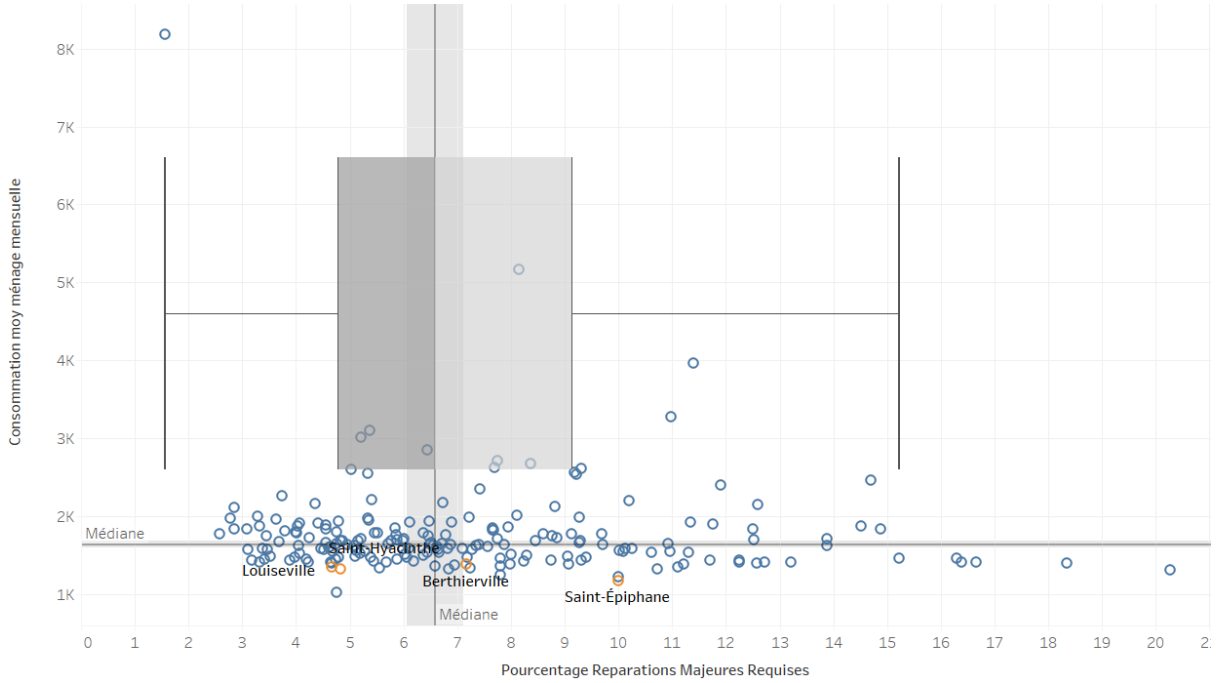
Deux de nos quatre villes (Berthierville et Louiseville) sont dans le troisième quartile en termes de pourcentage de personnes âgées tandis qu'une autre (Saint-Hyacinthe) a un pourcentage de personnes âgées supérieur à la moyenne .

Nuage de points 2 : Municipalités selon la consommation mensuelle moyenne d'électricité par ménage en kWh en fonction du pourcentage de personnes âgées (méthode des k-voisins pour les municipalités sans valeurs manquantes)



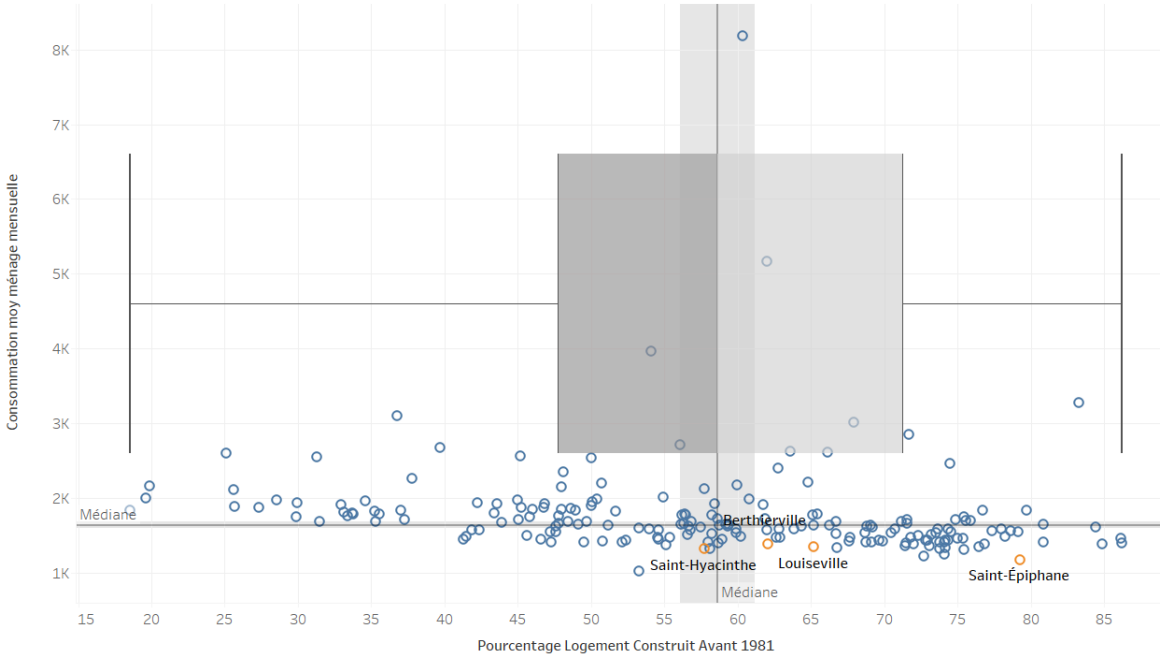
En ce qui concerne le pourcentage de logements nécessitant de réparations majeures requises, deux municipalités se trouvent dans le troisième quartile et deux en dessous de la médiane.

Nuage de points 3 : Municipalités selon la consommation mensuelle moyenne d'électricité par ménage en kWh en fonction du pourcentage de réparations majeures requises (méthode des k-voisins pour les municipalités sans valeurs manquantes)



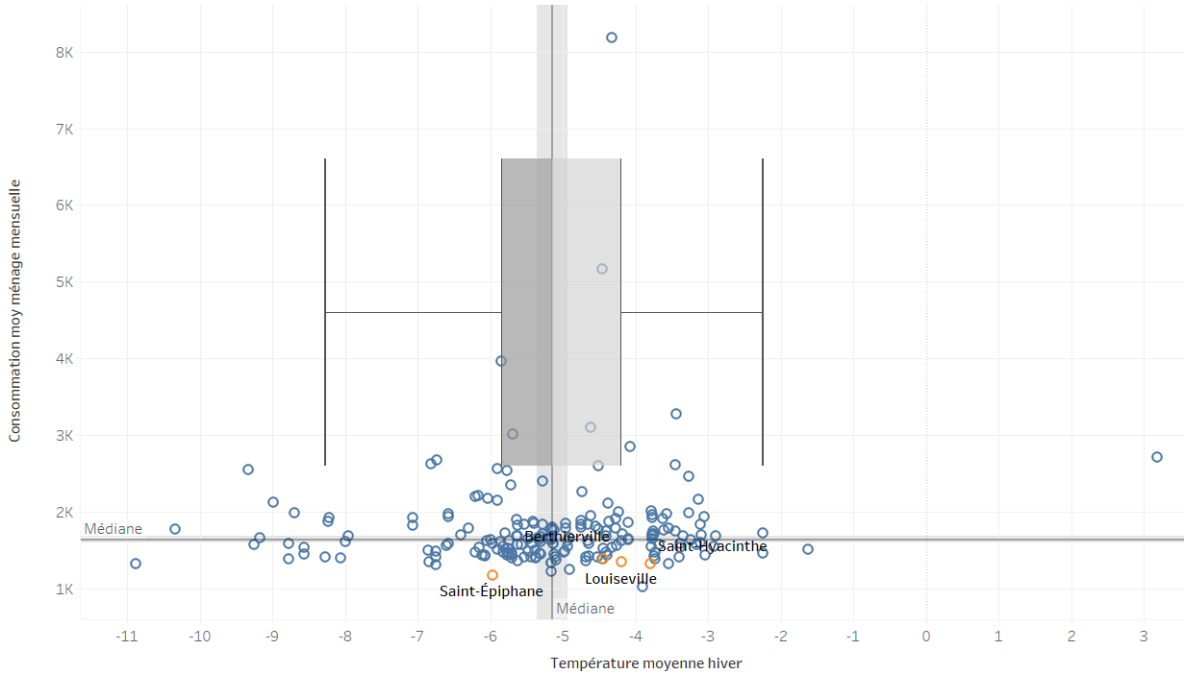
Trois municipalités (Saint-Epiphane, Louiseville, Berthierville) en situation de précarité énergétique ont un pourcentage de logements construits avant 1981 supérieur à la médiane des municipalités.

Nuage de points 4 : Municipalités selon la consommation d'électricité par ménage mensuelle moyenne en kWh en fonction du pourcentage de logements construits avec 1981 (méthode des k-voisins pour les municipalités sans valeurs manquantes)



Trois des quatres villes ont une température moyenne en hiver supérieure à la médiane des autres municipalités.

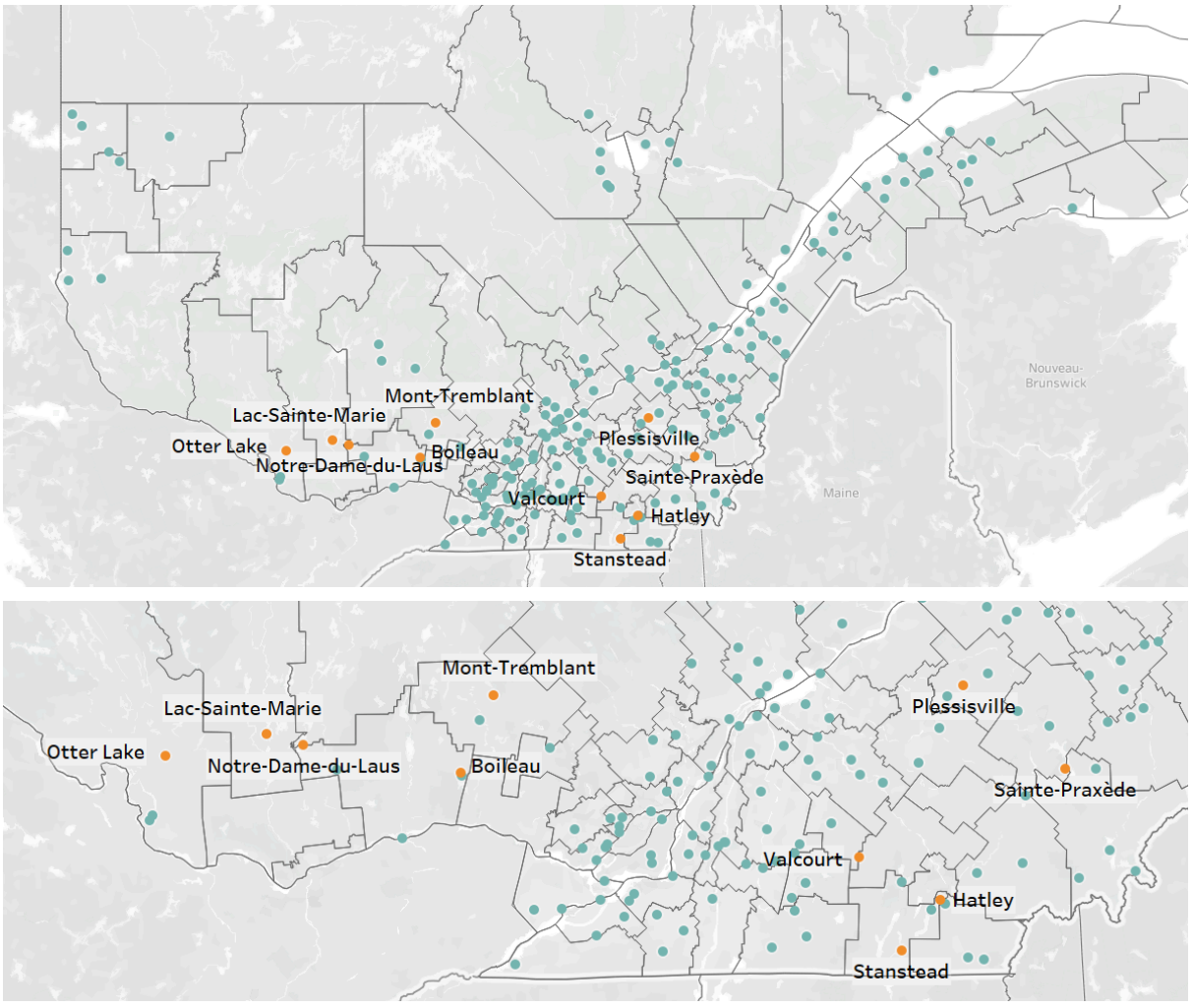
Nuage de points 5 : Municipalités selon la consommation d'électricité par ménage mensuelle moyenne en kWh en fonction de la température moyenne en hiver(°C) (méthode des k-voisins pour les municipalités sans valeurs manquantes)



5.1.2. Indice 2M

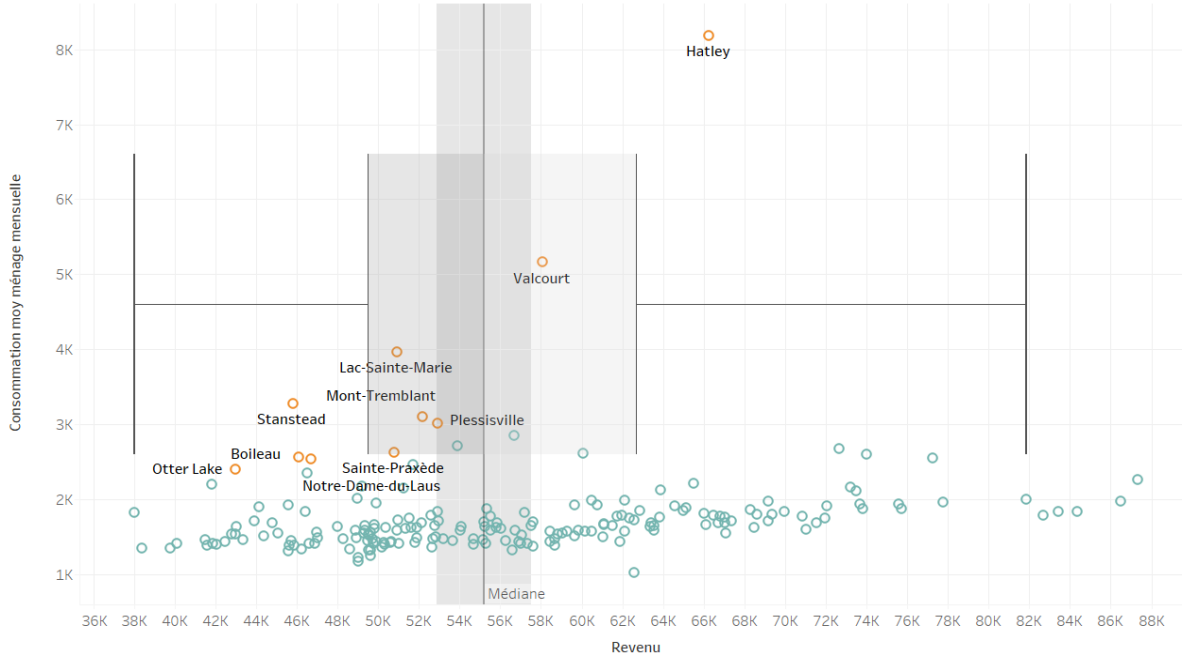
En sélectionnant deux fois la médiane du pourcentage des revenus consacrés à l'électricité (indicateur 2M) pour les ménages, on arrive à 5.38% comme seuil de précarité énergétique. Dix municipalités se trouvent au-dessus de ce seuil : Otter Lake, Lac-Sainte-Marie, Notre-Dame-du-Laus, Boileau, Mont-Tremblant, Plessisville, Valcourt, Hatley, Stanstead et Sainte-Praxède.

Cartes 2.a et 2.b : Municipalités en situation de précarité énergétique identifiées avec l'indicateur 2M pour les municipalités sans valeurs manquantes



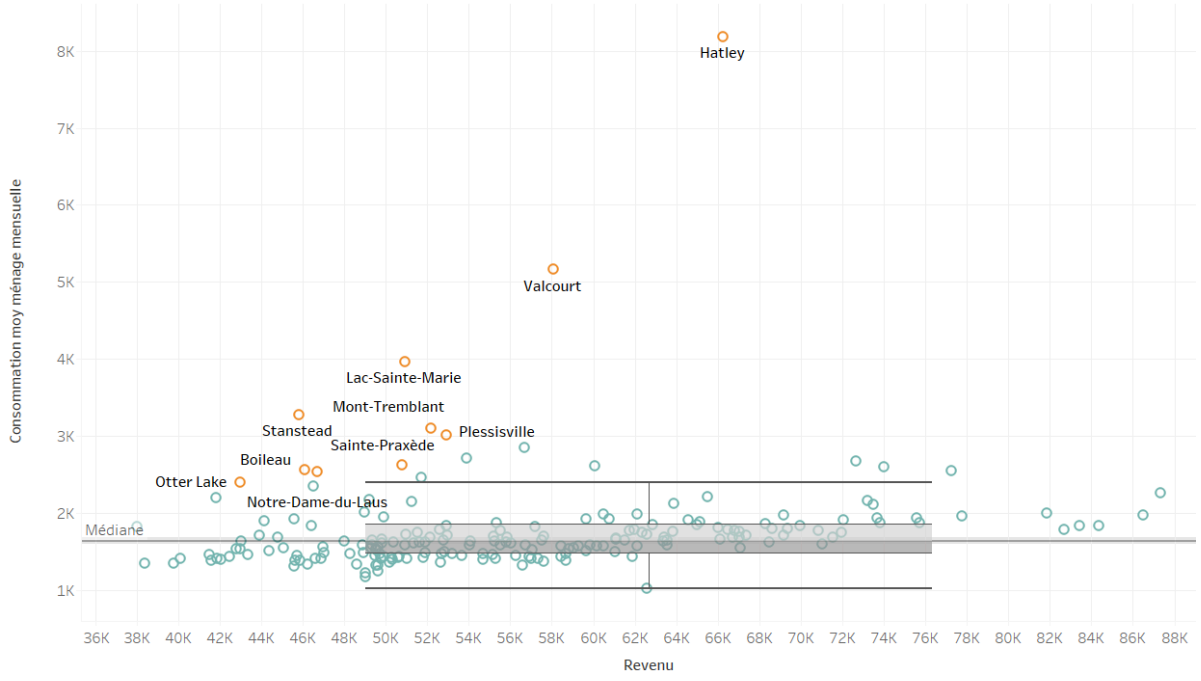
Huit de ces municipalités se situent en dessous de la médiane du revenu moyen après impôts des ménages qui y résident.

Nuage de points 6 : Municipalités selon la consommation d'électricité par ménage mensuelle moyenne en kWh en fonction du revenu($) (indicateur 2M pour les municipalités sans valeurs manquantes)



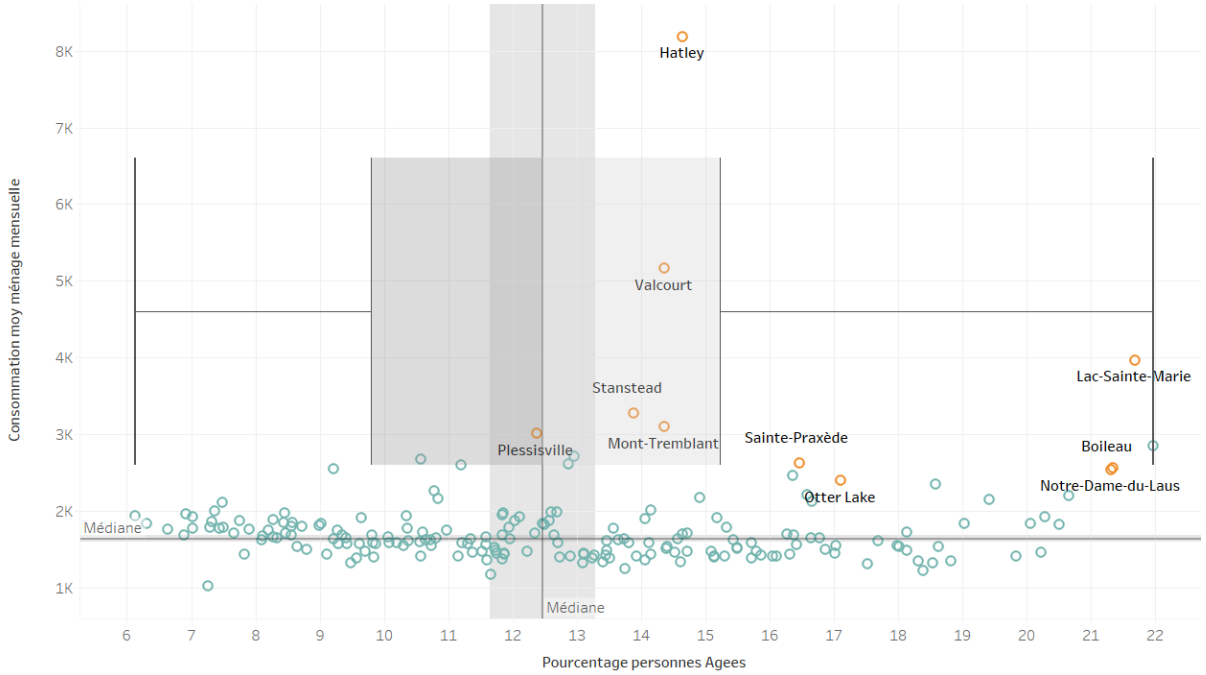
De plus, la totalité de ces municipalités se trouve bien au-dessus de la médiane de consommation moyenne d'électricité des ménages contrairement à l'indicateur avec la méthode des voisins.

Nuage de points 7 : Municipalités selon la consommation d'électricité par ménage mensuelle moyenne en kWh en fonction du revenu($) (indicateur 2M pour les municipalités sans valeurs manquantes)



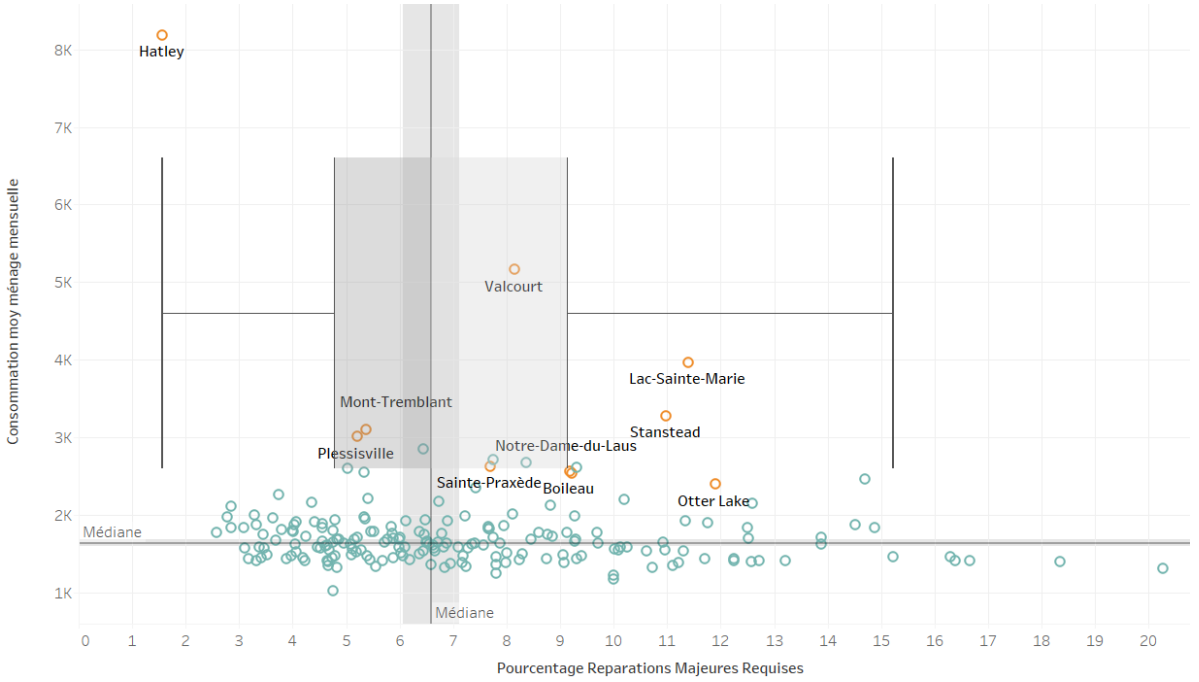
Neuf des dix municipalités sont au-dessus de la médiane en pourcentage de personnes âgées dont trois qui sont dans le top quatre des municipalités avec le plus haut pourcentage de personnes âgées.

Nuage de points 8 : Municipalités selon la consommation d'électricité par ménage mensuelle moyenne en kWh en fonction du pourcentage de personnes âgées (indicateur 2M pour les municipalités sans valeurs manquantes)



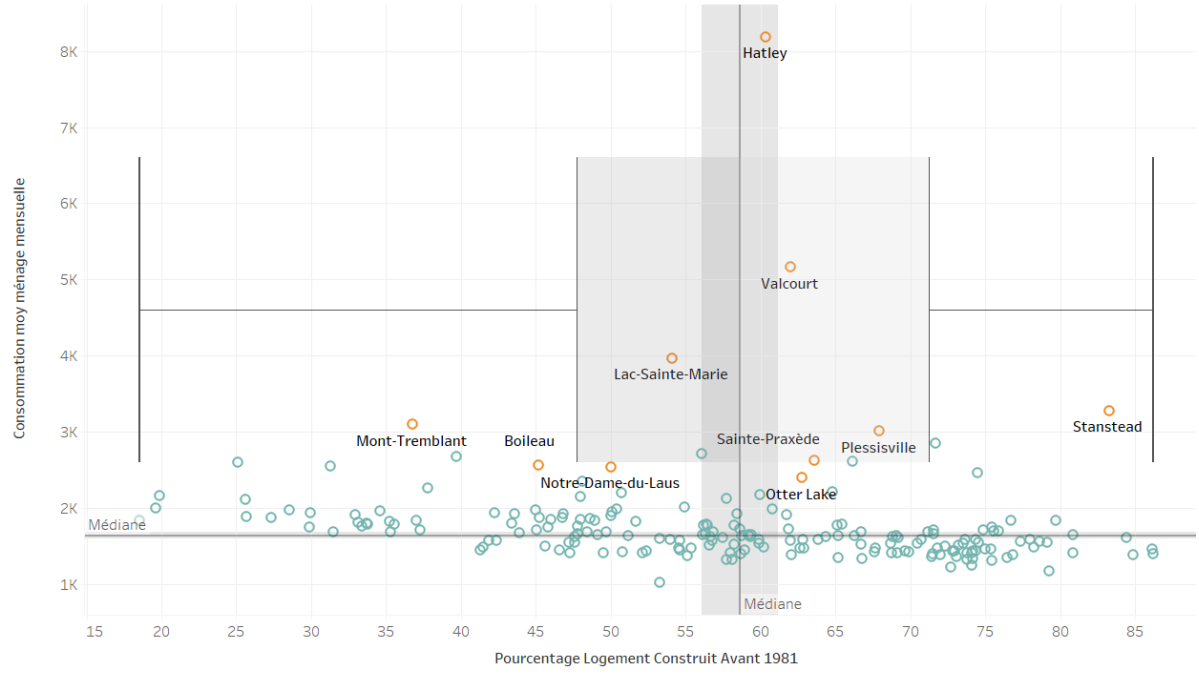
Sept municipalités ont un pourcentage de logements nécessitant des réparations majeures supérieur à la médiane, dont cinq dans le troisième quartile.

Nuage de points 9 : Municipalités selon la consommation d'électricité par ménage mensuelle moyenne en kWh en fonction du pourcentage de réparations majeures requises (indicateur 2M pour les municipalités sans valeurs manquantes)



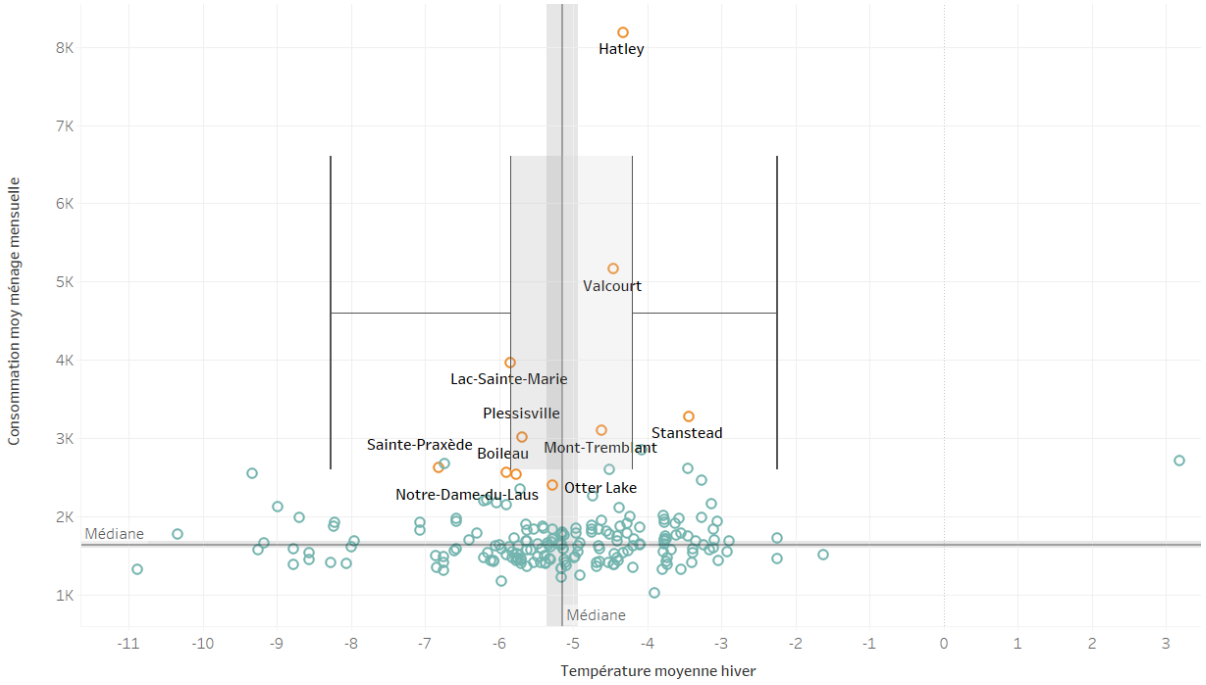
On ne remarque pas de distribution particulière des municipalités en situation de précarité énergétique par rapport à leur pourcentage de logements construits avec 1981 : quatre d'entre elles ont un pourcentage inférieur à la médiane, six d'entre elles supérieur.

Nuage de points 10 : Municipalités selon la consommation d'électricité par ménage mensuelle moyenne en kWh en fonction du pourcentage de logements construits avant 1981 (indicateur 2M pour les municipalités sans valeurs manquantes)



On ne remarque pas non plus de distribution particulière des municipalités en fonction de la température moyenne en hiver.

Nuage de points 11 : Municipalités selon la consommation d'électricité par ménage mensuelle moyenne en kWh en fonction de la température moyenne en hiver (indicateur 2M pour les municipalités sans valeurs manquantes)



Finalement, sur toutes les visualisations précédentes, on voit qu'Hatley est une municipalité à part des autres : non seulement son revenu par ménage est le plus élevé de toutes, dépassant de près de 10 000 $ par an celui de la deuxième municipalité, Valcourt, mais en plus, c'est la seule municipalité avec un faible pourcentage de logements nécessitant des réparations majeures.

5.1.3. Statistiques descriptives comparatives

Pour analyser les profils des municipalités sans valeur manquante en situation de précarité énergétique selon l'indicateur 2M et notre méthode k-NN, on peut s'appuyer sur le tableau suivant.

La population moyenne et médiane des municipalités identifiées par la méthode k-NN est significativement plus élevée (respectivement 17 450 et 5 863) que celle identifiée par l'indicateur 2M (respectivement 2 435 et 1 035). Cela suggère que la méthode k-NN inclut des municipalités plus grandes, tandis que l'indicateur 2M semble cibler des zones moins peuplées.

Les résultats montrent aussi que le revenu médian des ménages après impôts est plus faible dans les municipalités via la méthode k-NN (45 002 $) comparé à la méthode 2M (51 273 $).

Tableau 4 : Statistiques descriptives pour les municipalités sans valeur manquante en situation de précarité énergétique selon la méthode k-NN ou l'indicateur 2M

Variable	Moyenne	Médiane	Ecart-Type	Minimum	Maximum
Population en 2021 (méthode k-NN)	17 450	5 863	26 659	836	57 239
Population en 2021 (2M)	2 435	1 035	3 275	351	10 992
Revenu médian des ménages après impôts ($) (méthode k-NN)	45 002	45 300	5 046	39 794	49 615
Revenu médian des ménages après impôts ($) (2M)	51 273	50 848	6 840	42 998	66 254
Taille moyenne des ménages (méthode k-NN)	2.04	2.00	0.15	1.90	2.25
Taille moyenne des ménages (2M)	2.11	2.15	0.18	1.85	2.35
Pourcentage des revenus des ménages consacrés à l'électricité (méthode k-NN)	2.49	2.56	0.47	1.91	2.93
Pourcentage des revenus des ménages consacrés à l'électricité (2M)	7.37	6.22	2.79	5.08	13.84
Coût de l'électricité par ménage par an (méthode k-NN)	1 103	1 131	120	936	1 216
Coût de l'électricité par ménage par an (2M)	3 925	3 270	2 132	2 208	9 171
Consommation d'électricité par ménage par an (kWh) (méthode k-NN)	15 696	16 002	1 233	13 955	16 827
Consommation d'électricité par ménage par an (kWh) (2M)	43 017	36 682	20 615	26 414	93 745
Pourcentage de logements construits avant 1981 (méthode k-NN)	66.06	63.61	9.33	57.74	79.29
Pourcentage de logements construits avant 1981 (2M)	58.60	61.18	12.92	36.78	83.27
Pourcentage de logement nécessitant des réparations majeures (méthode k-NN)	6.66	5.99	2.50	4.67	10.00
Pourcentage de logement nécessitant des réparations majeures (2M)	8.06	8.66	3.24	1.56	11.89
Température moyenne en hiver (°C) (méthode k-NN)	-4.60	-4.32	0.95	-5.97	-3.81
Température moyenne en hiver (°C) (2M)	-5.22	-5.49	0.99	-6.81	-3.44
Pourcentage des personnes âgées de plus de 65 ans (méthode k-NN)	14.69	14.40	2.94	11.65	18.31
Pourcentage des personnes âgées de plus de 65 ans (2M)	16.75	15.55	3.49	12.37	21.68

La taille moyenne des ménages est légèrement plus élevée dans les municipalités utilisant l'indicateur 2M (2.11) que celles utilisant la méthode k-NN (2.04).

Comme nous l'avions fait remarquer dans les parties précédentes, les ménages dans les municipalités sous l'indicateur 2M consacrent une part beaucoup plus importante de leur revenu à l'électricité (7.37 %) que ceux classés par la méthode k-NN (2.49 %).

Les coûts d'électricité par ménage par an révèlent également une nette différence. Les ménages sous l'indicateur 2M ont des coûts beaucoup plus élevés (3 925 $) par rapport à ceux identifiés par la méthode k-NN (1 103 $), soulignant à nouveau les défis financiers liés à la consommation d'énergie dans ces municipalités.

Cela est lié à la consommation d'électricité par ménage par an qui est également bien supérieure dans les municipalités ayant recouru à la méthode 2M (43 017 kWh) que dans celles sous la méthode k-NN (15 696 kWh). Cela peut indiquer non seulement des besoins énergétiques plus élevés, mais aussi que ces ménages peuvent vivre dans des logements moins efficaces énergétiquement.

Le pourcentage de logements nécessitant des réparations majeures est légèrement plus élevé dans la méthode 2M (8.06 %) par rapport à la méthode k-NN (6.66 %). Cela pourrait refléter un contexte de logement moins favorable pour les ménages identifiés par la 2M.

La température moyenne en hiver est plus basse dans les municipalités identifiées par la méthode 2M (-5.22 °C) par rapport à celles utilisant la méthode k-NN (-4.60 °C).

Les municipalités identifiées par la méthode 2M présentent également un pourcentage élevé de personnes âgées de plus de 65 ans (16.75 %) comme celles de la méthode k-NN (14.69 %).

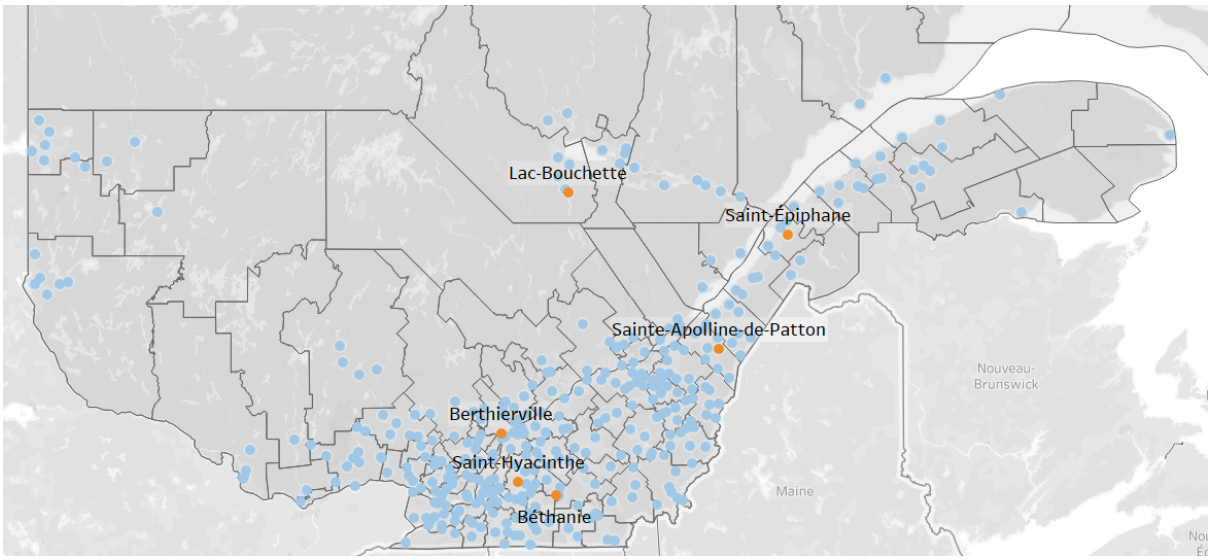
5.2. Municipalités avec valeurs imputées

5.2.1. Méthode des k-voisins

En ce qui concerne l'échantillon de municipalités aux valeurs imputées, six municipalités apparaissent en situation de précarité énergétique avec la méthode des k-voisins, dont trois de celles identifiées dans notre échantillon sans valeurs manquantes : Saint-Épiphane, Sainte-Apolline de Patton, Berthierville, Saint-Hyacinthe, Lac Bouchette et Béthanie.

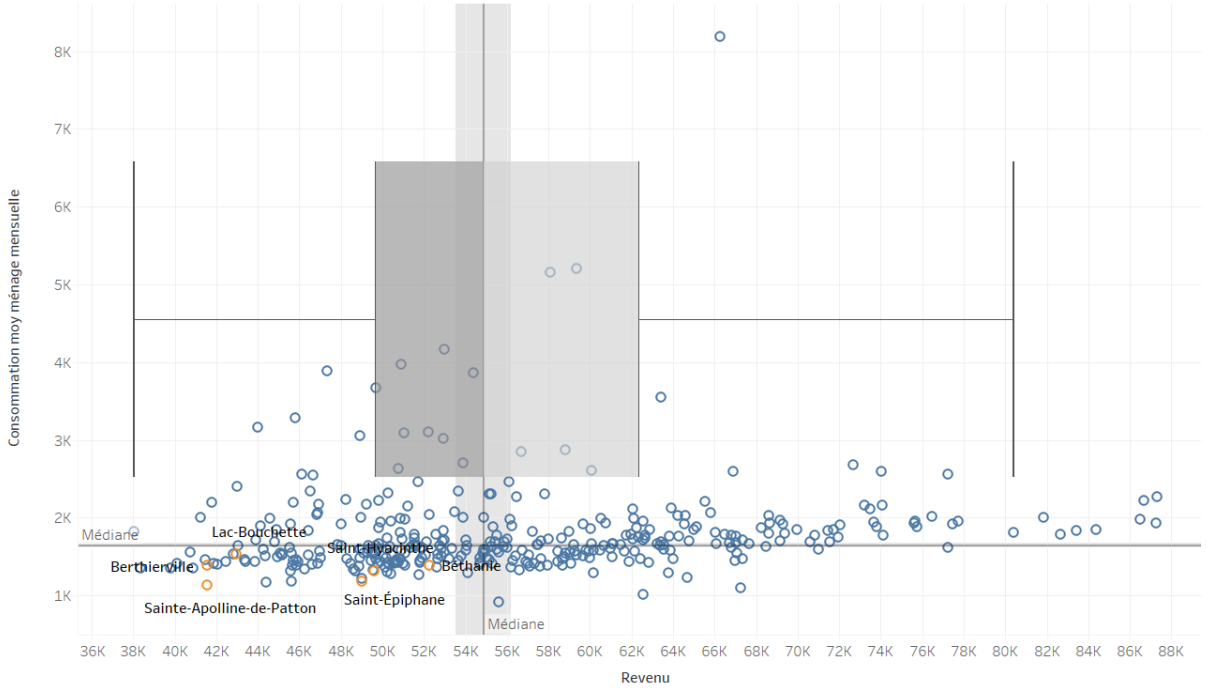
Géographiquement parlant, ces municipalités sont dispersées à travers le Québec, couvrant à la fois des régions urbaines (Saint-Hyacinthe) et rurales.

Carte 3 : Municipalités en situation de précarité énergétique identifiées grâce à la méthode des k-voisins pour les municipalités avec valeurs imputées



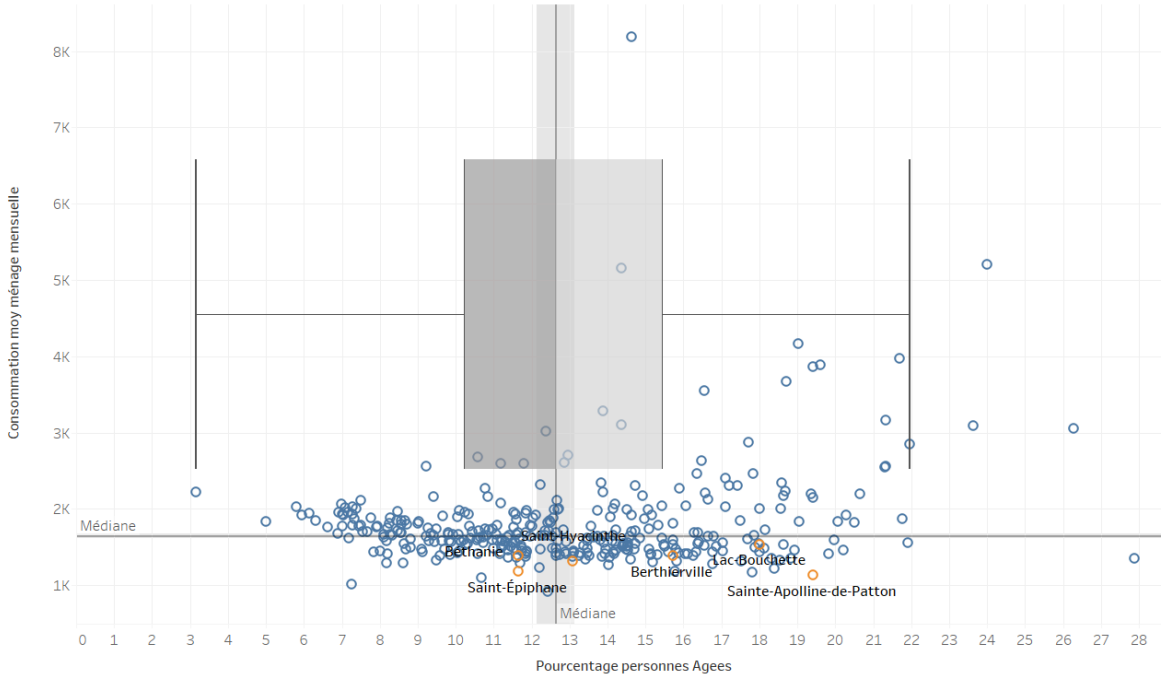
La construction même de notre indicateur nécessite que nos municipalités aient un faible revenu et une faible consommation d'électricité : ainsi, comme pour les municipalités sans valeurs imputées, toutes les municipalités mises en lumière par l'indicateur de la méthode des voisins sont en dessous de la médiane du revenu moyen après impôts par ménage en et en dessous de la médiane de consommation d'électricité.
Trois ont des revenus particulièrement faibles (inférieurs à 44 000$ par ménage) : Berthierville, Lac-Bouchette, Sainte-Apolline-de-Patton.

Nuage de points 12 : Municipalités selon la consommation d'électricité par ménage mensuelle moyenne en kWh en fonction du revenu($) (méthode des k-voisins pour les municipalités avec valeurs imputées)



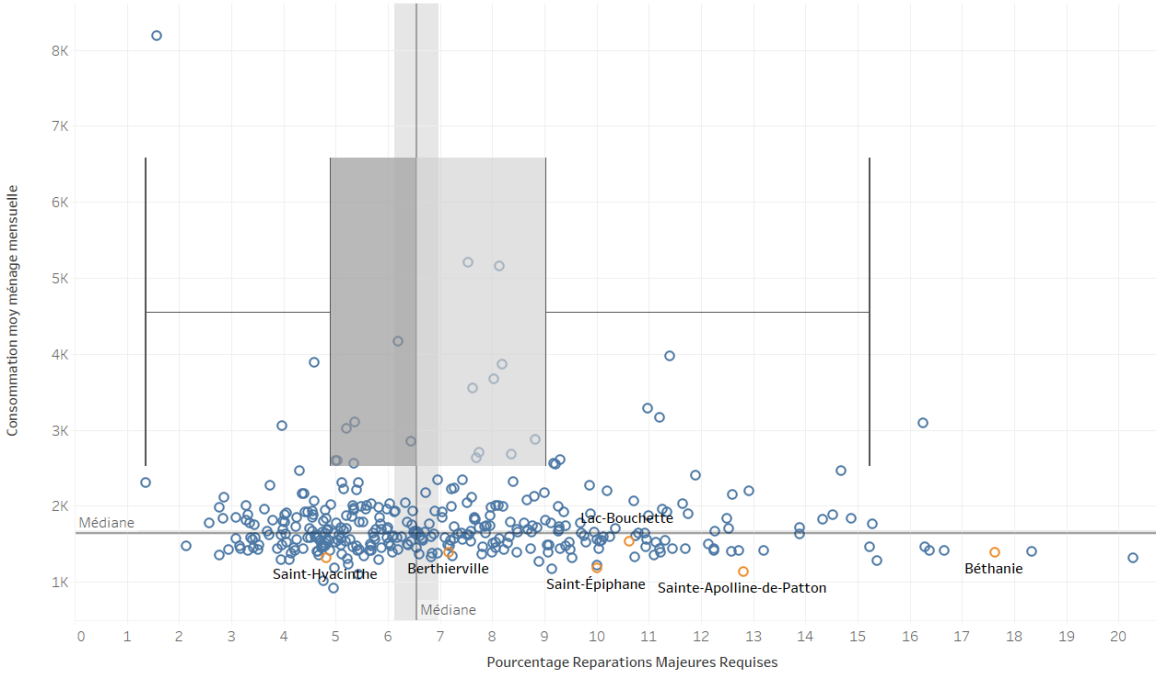
Quatre de ces municipalités ont un pourcentage de personnes âgées supérieur à la médiane.

Nuage de points 13 : Municipalités selon la consommation d'électricité par ménage mensuelle moyenne en kWh en fonction du pourcentage de personnes âgées (méthode des k-voisins pour les municipalités avec valeurs imputées)



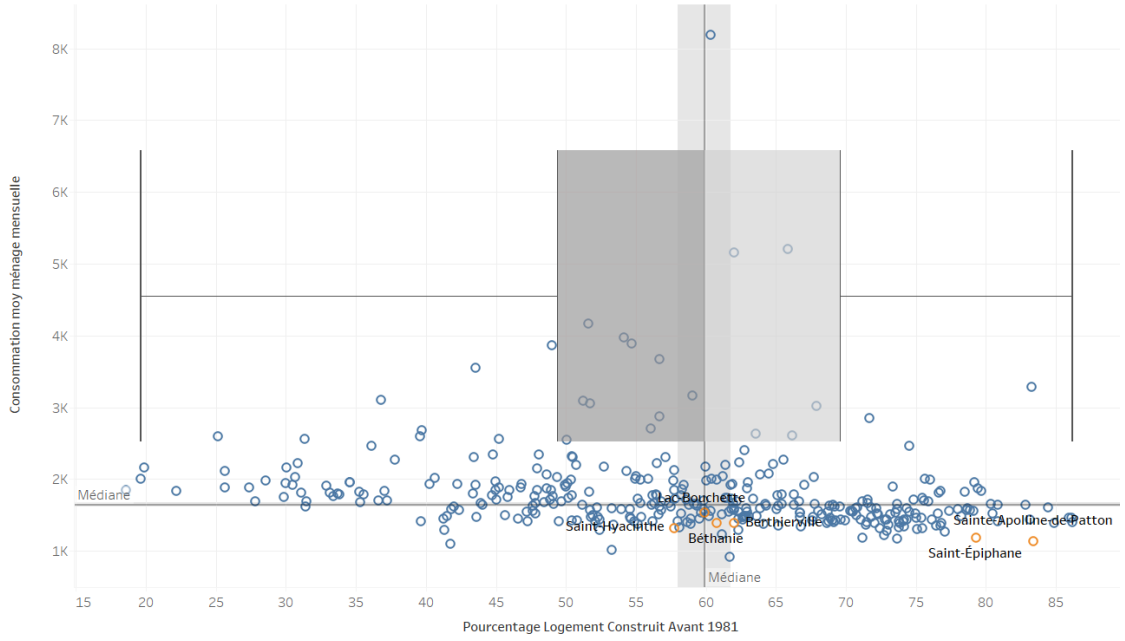
Cinq municipalités ont un pourcentage de logements nécessitant des réparations majeures supérieur à la médiane des réparations majeures requises de toutes les municipalités.

Nuage de points 14 : Municipalités selon la consommation d'électricité par ménage mensuelle moyenne en kWh en fonction du pourcentage de réparations majeures requises (méthode des k-voisins pour les municipalités avec valeurs imputées)



Cinq municipalités en situation de précarité énergétique ont aussi un pourcentage de logements construits avant 1981 supérieur à la médiane des municipalités du Québec.

Nuage de points 15 : Municipalités selon la consommation d'électricité par ménage mensuelle moyenne en kWh en fonction du pourcentage de logement construit avant 1981 (méthode des k-voisins pour les municipalités avec valeurs imputées)

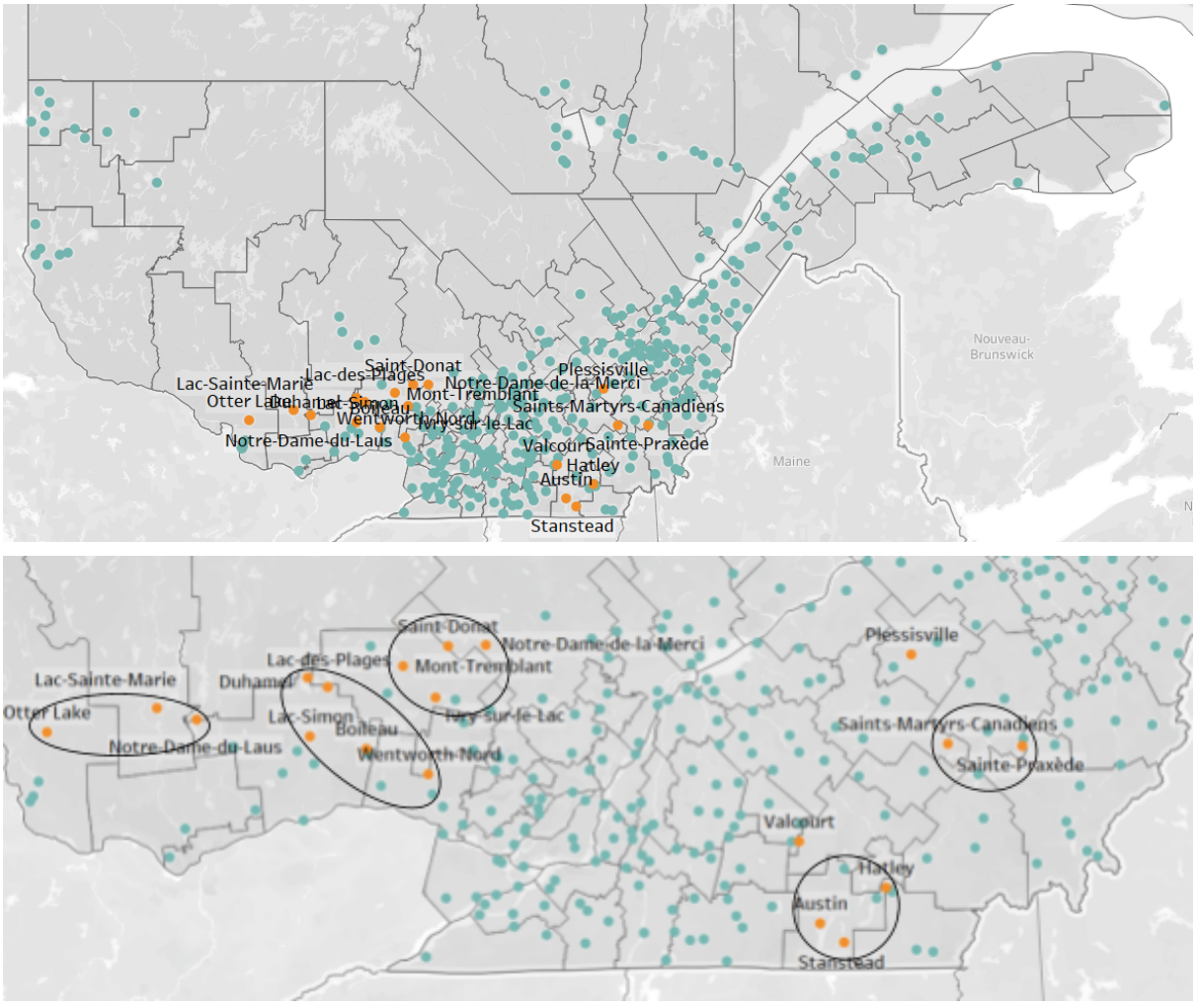


5.2.2. Indice 2M

En comparaison, la méthode 2M met en évidence 18 municipalités en situation de précarité énergétique pour nos municipalités imputées : Lac-Simon, Boileau, Lac-des-Plages, Notre-Dame-de-la-Merci, Valcourt, Lac-Sainte-Marie, Notre-Dame-du-Laus, Plessisville, Saint-Donat, Otter Lake, Mont-Tremblant, Sainte-Praxède, Saints-Martyrs-Canadiens, Duhamel, Wentworth-Nord Hatley, Austin, Stanstead.

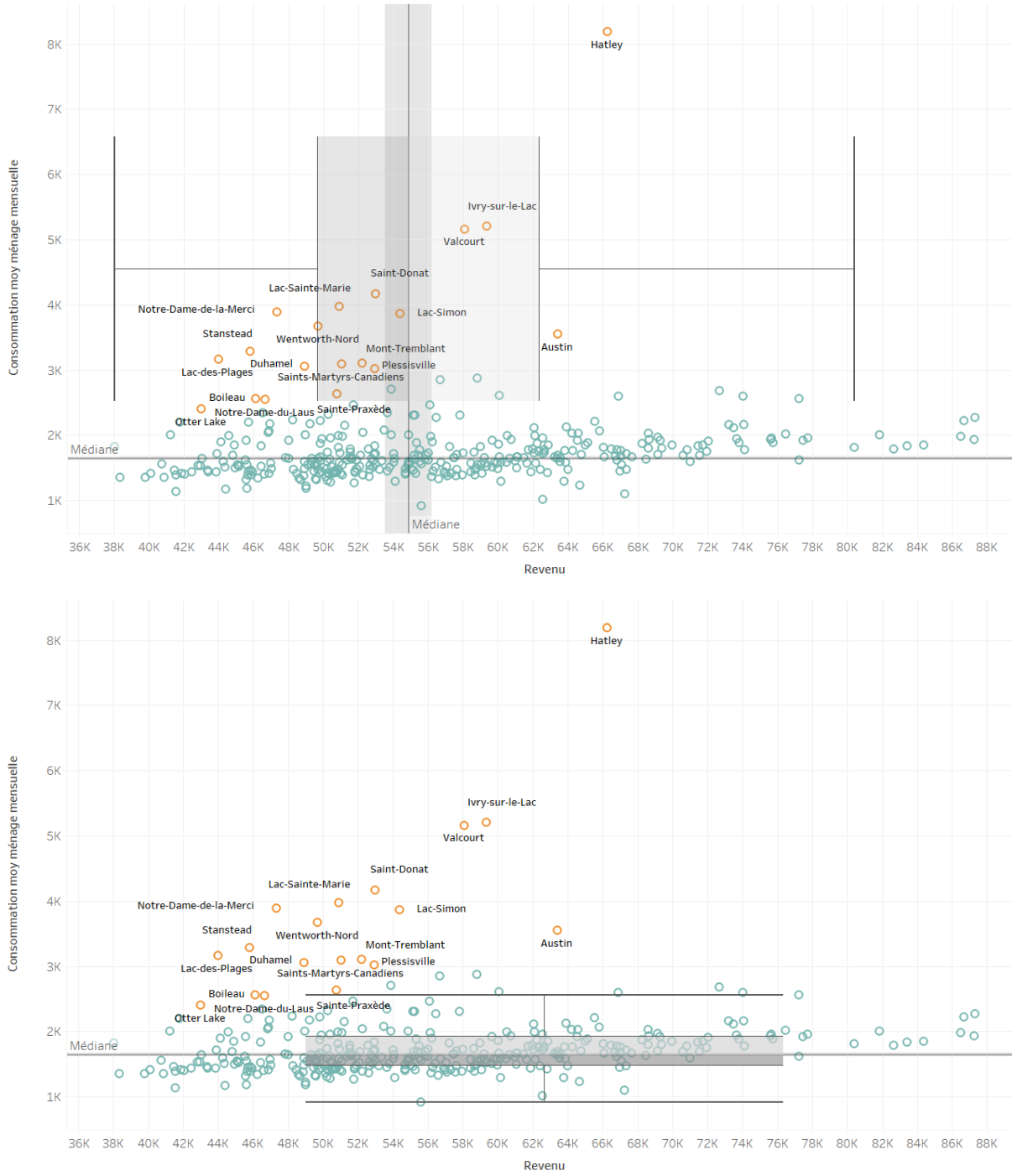
On peut distinguer des groupes géographiques de municipalités en situation de précarité énergétique lorsqu'on se penche sur la carte du Québec.

Cartes 4.a et 4.b : Municipalités en situation de précarité énergétique selon l'indice 2M pour les municipalités avec valeurs imputées



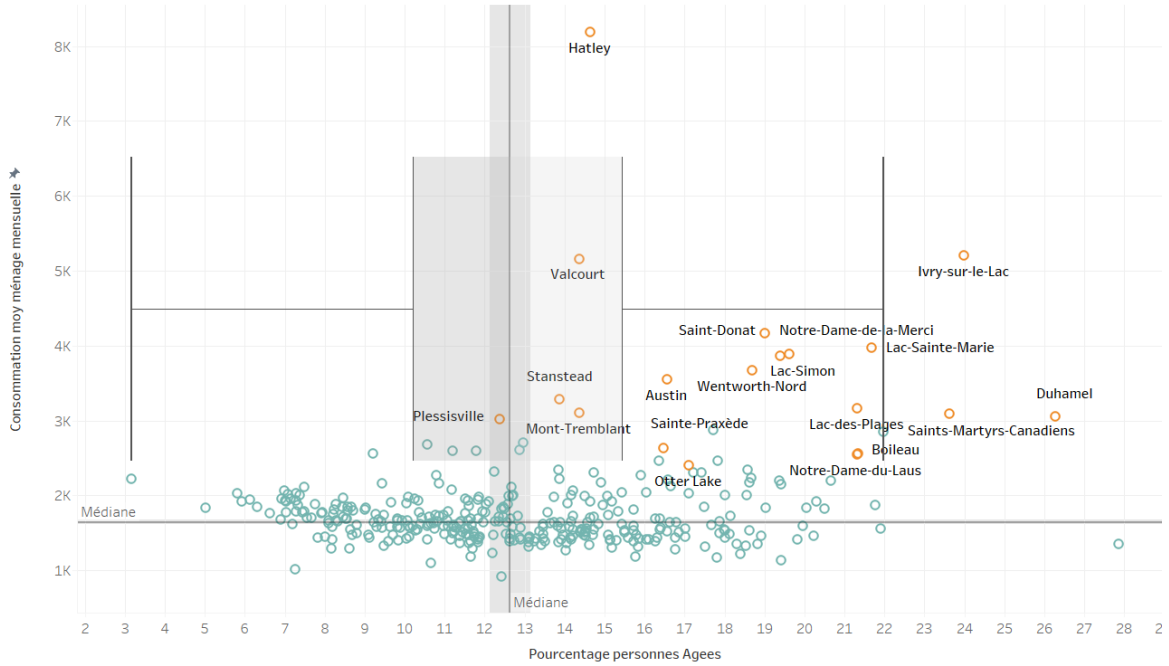
Toutes nos municipalités exceptées quatre (Valcourt, Ivry-sur-le-Lac, Austin et Hatley) ont un revenu moyen par ménage après impôts inférieur à la médiane des municipalités du Québec et une consommation d'électricité bien supérieure à la médiane.

Nuage de points 16.a et 16.b : Municipalités selon la consommation d'électricité par ménage mensuelle moyenne en kWh en fonction du revenu ($) (indicateur 2M pour les municipalités avec valeurs imputées)



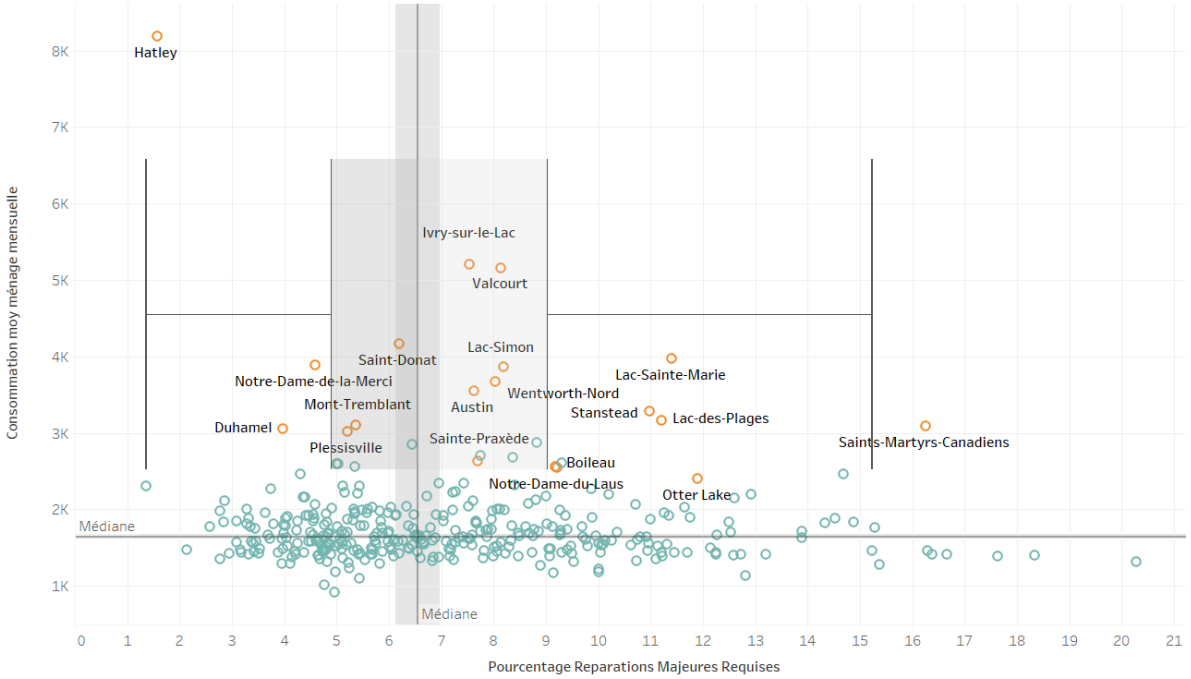
Exceptée une municipalité (Plessisville), toutes les municipalités en situation de précarité énergétique ont un pourcentage de personnes âgées supérieur à la médiane des municipalités du Québec.

Nuage de points 17 : Municipalités selon la consommation d'électricité par ménage mensuelle moyenne en kWh en fonction du pourcentage de personnes âgées (indicateur 2M pour les municipalités avec valeurs imputées)



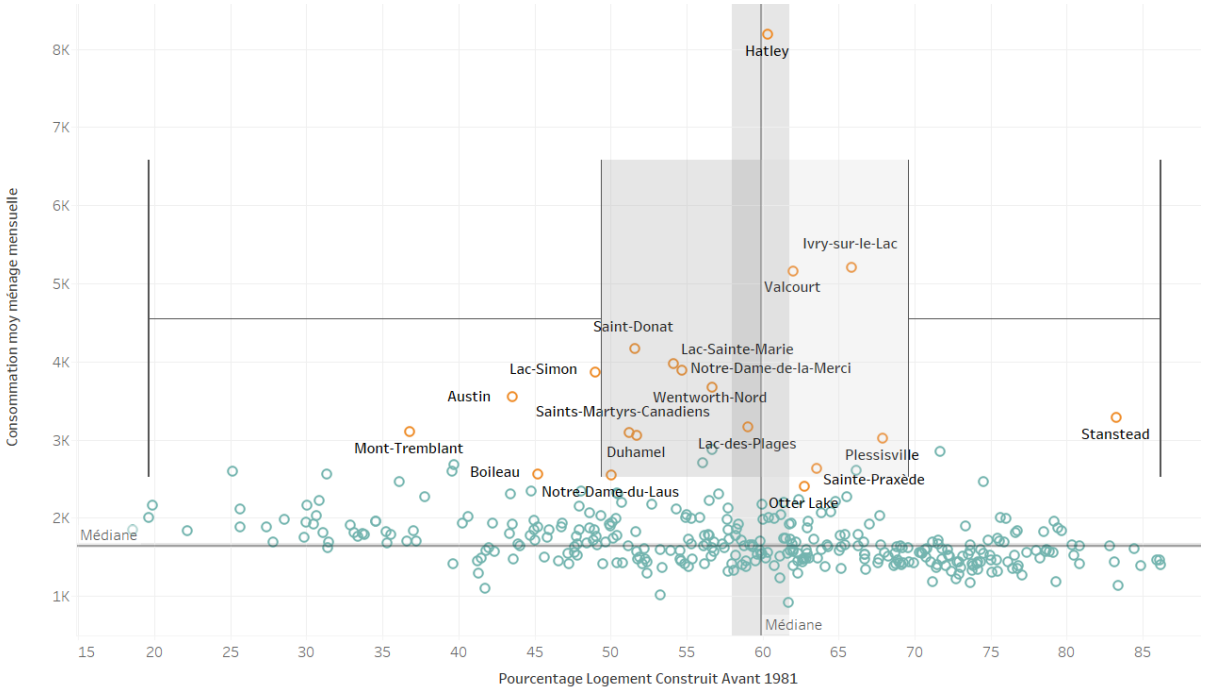
Douze municipalités déclarées en situation de précarité énergétique ont aussi un pourcentage de logements nécessitant des réparations majeures supérieur à la moyenne au Québec.

Nuage de points 18 : Municipalités selon la consommation d'électricité par ménage mensuelle moyenne en kWh en fonction du pourcentage de réparations majeures (indicateur 2M pour les municipalités avec valeurs imputées)



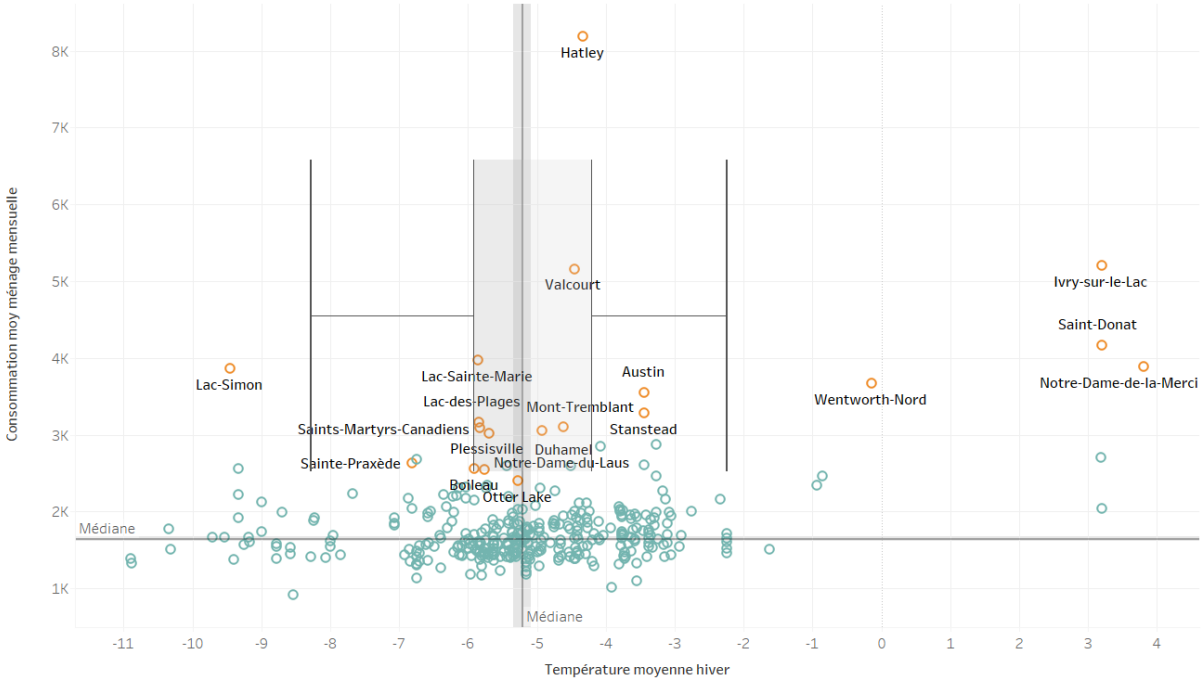
Une majorité de ces municipalités a une proportion de logements construits avant 1981 inférieur à la médiane.

Nuage de points 19 : Municipalités selon la consommation d'électricité par ménage mensuelle moyenne en kWh en fonction du pourcentage de logements construits avant 1981 (indicateur 2M pour les municipalités avec valeurs imputées)



Enfin, ces municipalités ont des températures hivernales avec des distributions variables : neuf sont en dessous de la médiane et 10 au dessous avec trois municipalités (Ivry-sur-le-Lac, Saint-Donat, Notre-Dame-de-la-Merci) qui ont les températures moyennes en hiver les plus hautes du Québec.

Nuage de points 20 : Municipalités selon la consommation d'électricité par ménage mensuelle moyenne en kWh en fonction de la température moyenne en hiver (indicateur 2M pour les municipalités avec valeurs imputées)



5.2.3. Statistiques descriptives comparatives

Pour analyser les profils des municipalités avec valeurs imputées en situation de précarité énergétique selon l'indicateur 2M et notre méthode k-NN, on peut s'appuyer sur la tableau suivant.

Tableau 5 : Statistiques descriptives pour les municipalités avec valeurs imputées en situation de précarité énergétique selon la méthode k-NN ou l'indicateur 2M

Variable	Moyenne	Médiane	Ecart-Type	Minimum	Maximum
Population en 2021 (méthode k-NN)	10 750	1 010	22 824	350	57 239
Population en 2021 (2M)	1 994	1 049	2 589	277	10 992
Revenu médian des ménages après impôts ($) (méthode k-NN)	46 165	46 002	4 687	41 544	52 256
Revenu médian des ménages après impôts ($) (2M)	51 359	50 848	6 220	42 998	66 254
Taille moyenne des ménages (méthode k-NN)	2.12	2.08	0.22	1.90	2.50
Taille moyenne des ménages (2M)	2.01	1.98	0.18	1.75	2.35
Pourcentage des revenus des ménages consacrés à l'électricité (méthode k-NN)	2.46	2.32	0.52	1.91	3.26
Pourcentage des revenus des ménages consacrés à l'électricité (2M)	7.41	6.93	2.11	5.08	13.84
Cout de l'électricité par ménage par an (méthode k-NN)	1 128	1 183	203	849	1 402
Coût de l'électricité par ménage par an (2M)	3 881	3 508	1 582	2 208	9 171
Consommation d'électricité par ménage par an (kWh) (méthode k-NN)	15 846	16 501	2 165	12 667	18 621
Consommation d'électricité par ménage par an (kWh) (2M)	42 591	38 990	15 300	26 414	93 745
Pourcentage de logements construits avant 1981 (méthode k-NN)	67.19	61.41	11.13	57.74	83.40
Pourcentage de logements construits avant 1981 (2M)	55.75	54.40	10.42	36.78	83.27
Pourcentage de logement nécessitant des réparations majeures (méthode k-NN)	10.51	10.31	4.47	4.82	17.62
Pourcentage de logement nécessitant des réparations majeures (2M)	8.15	8.08	3.46	1.56	16.25
Température moyenne en hiver (°C) (méthode k-NN)	-5.67	-5.21	1.80	-8.57	-3.81
Température moyenne en hiver (°C) (2M)	-4.16	-5.10	3.33	-9.45	3.81
Pourcentage des personnes âgées de plus de 65 ans (méthode k-NN)	14.92	14.40	3.32	11.65	19.41
Pourcentage des personnes âgées de plus de 65 ans (2M)	18.45	18.86	3.76	12.37	26.27

La population moyenne des municipalités identifiées par la méthode k-NN (10 750) est significativement plus élevée que celle des municipalités identifiées par l'indicateur 2M (1 994). Cette différence indique que la méthode k-NN inclut des municipalités ayant une population généralement plus importante, tout comme observé précédemment dans les données non imputées.

Le revenu médian des ménages après impôts pour les municipalités identifiées par la méthode k-NN est de 46 165 $, comparé à 51 359 $ pour les municipalités classées par l'indicateur 2M. Cela renforce l'idée que les municipalités k-NN sont plus exposées à des niveaux de précarité énergétique, en raison de leur revenu médian relativement plus bas, ce qui est en accord avec les résultats précédemment observés.

La taille moyenne des ménages est de 2.12 pour les municipalités de la méthode k-NN contre 2.01 pour celles à l'indicateur 2M. Cette tendance d'une taille de ménage légèrement plus élevée au sein des k-NN reste cohérente avec l'observation antérieure, ce qui peut influencer la consommation énergétique globale au sein de chaque municipalité.

Le pourcentage de revenu consacré à l'électricité est de 2.46 % pour les valeurs imputées des k-NN, tandis qu'il s'élève à 7.41 % pour celles de l'indicateur 2M. Cela corrobore les conclusions précédentes, indiquant que les ménages associés à l'indicateur 2M rencontrent des charges énergétiques plus lourdes même lorsque leurs revenus médian sont plus élevés.

Les coûts d'électricité par ménage par an pour les municipalités sous la méthode k-NN se chiffrent à 1 128 $, alors que ceux des municipalités classées par l'indicateur 2M atteignent 3 881 $. Cela est aligné une nouvelle fois avec les résultats précédents.

De même, on constate à nouveau que la consommation d'électricité par ménage par an est de 15 846 kWh pour les municipalités k-NN, alors qu'elle est de 42 591 kWh pour celles utilisant l'indicateur 2M.

Le pourcentage de logements nécessitant des réparations majeures est de 10.51 % pour les données imputées contre 8.06 % pour les données 2M. Cela indique, tout comme observé précédemment, que les municipalités identifiées par 2M sont confrontées à un contexte de logement potentiellement moins favorable.

Enfin, comme pour les municipalités sans valeur manquante, le pourcentage de personnes âgées (plus de 65 ans) dans les municipalités en situation de précarité énergétique est très élevé dans les deux cas (14.92 % pour la méthode k-NN et 18.45 % pour celles à l'indicateur 2M).

5.3. Discussion des résultats

Dans cette analyse, l'intersection entre l'approche des k-voisins et la méthode 2M pour identifier la précarité énergétique s'avère nulle, c'est-à-dire que ces deux indicateurs n'ont détecté aucune municipalité en commun comme étant en situation de précarité énergétique (tableau 6).

Tableau 6 : Récapitulatif de la liste des municipalités en situation de précarité énergétique selon la méthode k-NN et l'indicateur 2M (note: une municipalité surlignée ne se retrouve pas dans l'ensemble des municipalités avec valeur imputées)

Méthode	Liste de municipalités considérées en situation de précarité énergétique	Nombre de municipalités
k-NN (municipalités sans valeur manquante)	Saint-Épiphanie, Louiseville, Berthierville, Saint-Hyacinthe	4
2M (municipalités sans valeur manquante)	Otter Lake, Lac-Sainte-Marie, Notre-Dame-du-Laus, Boileau, Mont-Tremblant, Plessisville, Valcourt, Hatley, Stanstead, Sainte-Praxède	10
k-NN (municipalités avec valeurs imputées)	Saint-Épiphane, Sainte-Apolline de Patton, Berthierville, Saint-Hyacinthe, Lac Bouchette, Béthanie	6
2M (municipalités avec valeurs imputées)	Lac-Simon, Boileau, Lac-des-Plages, Notre-Dame-de-la-Merci, Valcourt, Lac-Sainte-Marie, Notre-Dame-du-Laus, Plessisville, Saint-Donat, Otter Lake, Mont-Tremblant, Sainte-Praxède, Saints-Martyrs-Canadiens, Duhamel, Wentworth-Nord Hatley, Austin, Stanstead	18

L'absence d'intersection entre ces deux approches suggère que chaque méthode capture des aspects différents de la vulnérabilité énergétique dans les municipalités.

L'approche des k-voisins se concentre sur un petit groupe de municipalités voisines en fonction des indicateurs de vulnérabilité énergétique sélectionnés. Elle évalue la situation relative d'une municipalité par rapport à ses voisines immédiates. Ainsi, si toutes les municipalités du groupe sont similaires en termes de richesse, une municipalité ne se démarquera pas forcément comme pauvre, même si elle l'est selon d'autres critères. Elle va avoir aussi tendance à favoriser les municipalités avec des ménages qui se restreignent en termes de consommation d'électricité.

La méthode 2M, en revanche, compare chaque municipalité à l'ensemble des municipalités, fournissant une vue d'ensemble et mettant en évidence celles qui se démarquent significativement par rapport à la moyenne régionale. Cette méthode va plutôt favoriser des ménages qui ne peuvent se permettre de consommer peu d'électricité et un ajout de seuil de revenu dans la méthode 2M pourrait porter plus d'attention sur les municipalités avec les ménages les plus pauvres, affinant ainsi la pertinence des interventions et évitant de prendre en compte des municipalités comme Hatley avec un haut revenu par ménage.
Ces résultats justifient une approche duale dans les politiques publiques, combinant des interventions localisées, fondées sur des comparaisons entre proches voisins, avec des efforts de type macro pour adresser des inégalités structurelles reflétées par les indicateurs globaux.

L'analyse des résultats en général met d'ailleurs en lumière des différences significatives dans les caractéristiques démographiques, économiques et de consommation d'énergie entre les municipalités identifiées par la méthode des k-voisins et celles utilisant l'indicateur 2M.

Pour commencer, les différences en termes de consommation d'électricité entre les deux méthodologies sont notables. Alors que les municipalités identifiées par la méthode k-NN affichent généralement une consommation en dessous de la médiane, celles de l'indicateur 2M mettent en évidence une consommation plus élevée, indiquant des ménages qui doivent dépenser largement pour des ressources énergétiques.

Par ailleurs, la population moyenne des municipalités identifiées par la méthode k-NN (10 750) est significativement plus élevée que celle des municipalités selon l'indicateur 2M (1 994). Cela nous donne une piste de réflexion pour expliquer les différentes caractéristiques de consommation d'électricité entre les ménages des municipalités de la méthode 2M et celle de la méthode k-NN.
Cela pourrait indiquer a priori que les municipalités plus grandes disposent de davantage de ressources ou d'infrastructures énergétiques de meilleure qualité, tandis que les municipalités moins peuplées, souvent considérées comme plus vulnérables, peuvent souffrir d'une prévalence plus élevée de logements anciens et moins efficaces sur le plan énergétique. Cependant, il est important de noter que les logements des municipalités identifiées par notre méthode k-NN tendent à être plus anciens (avec 67,19 % des logements construits avant 1981, contre 55,75 % pour l'indicateur 2M) et nécessitent également plus de réparations majeures (10,51 % contre 8,15 %).

Bien qu'une partie de la différence de consommation d'électricité puisse être expliquée par la proportion de personnes âgées (14,92 % pour la méthode k-NN contre 18,45 % pour l'indicateur 2M), il est légitime de considérer que la forte consommation d'électricité observée dans les municipalités en situation de précarité énergétique, selon l'indicateur 2M, par rapport à la faible consommation des municipalités sélectionnées par la méthode k-NN, peut être influencée par d'autres variables qui n'ont pas été étudiées.

Ainsi, il est raisonnable de penser que la forte population des municipalités identifiées par la méthode des k-voisins (k-NN) peut être attribuée à leur caractère urbain. Ces municipalités, comme Saint-Hyacinthe et Berthierville, offrent des infrastructures et des services qui attirent

une population plus nombreuse et, souvent, une diversité de logements répondant à des besoins variés. Dans ces environnements urbains, une proportion plus élevée de résidents vit en location, dans des habitations de type duplex ou appartements, ce qui leur permet de modérer leur consommation d'énergie. Par exemple, les logements en hauteur ou en colocation peuvent faciliter le partage des coûts de chauffage, rendant ainsi l'accès à une consommation d'énergie plus abordable.

En revanche, les municipalités identifiées par l'indicateur 2M, comme Valcourt ou Mont-Tremblant, tendent à être plus rurales. Ces régions, souvent moins densément peuplées, présentent des logements qui sont souvent des maisons individuelles, ce qui entraîne des dépenses énergétiques significativement plus élevées pour atteindre un confort thermique adéquat. Les ménages qui y résident peuvent être contraints de dépenser une proportion élevée de leurs revenus pour le chauffage, en raison des distances plus grandes à parcourir pour accéder aux ressources, ainsi que de la dépendance à des systèmes de chauffage moins efficaces.

On peut aussi imaginer que l'indicateur 2M semble également indiquer des ménages qui, sans être en situation de précarité générale, tendent à avoir des difficultés à gérer leur consommation d'électricité. C'est alors peut-être vers ces ménages qu'il faudrait diriger les programmes comme Econologis visant à les éduquer sur une consommation plus appropriée et à promouvoir la sobriété énergétique.
Les ménages des municipalités identifiées avec la méthode k-NN sont quant à eux plutôt des ménages qui se restreignent déjà et qui bénéficieraient réellement de mesures d'aide économiques, comme l'interfinancement.
Ce dernier se définit comme le mécanisme par lequel les entreprises contribuent financièrement à hauteur de 15% aux dépenses en électricité — et plus particulièrement en chauffage électrique — des ménages québécois (Pineau 2024).
Ainsi, un accès plus ciblé à l'interfinancement pour les municipalités identifiées par la méthode k-NN pourrait s'avérer plus efficace. En effet, ces municipalités, souvent plus vulnérables, pourraient avoir un besoin plus urgent d'un soutien financier direct pour alléger leurs factures d'électricité, tandis que les municipalités plus prospères pourraient ne pas nécessiter le même niveau d’aide.
Un ajout de seuil de revenu dans la méthode 2M pourrait également porter plus d’attention sur les municipalités avec les ménages les plus pauvres, affinant ainsi la pertinence des interventions et évitant de prendre en compte des municipalités comme Hatley, qui ont un haut revenu par ménage.

De ce fait, ces variations géographiques et structurelles mettent en évidence l'importance d'adapter les politiques et les interventions ciblées, tenant compte non seulement des caractéristiques démographiques, mais aussi des spécificités des environnements urbains et ruraux qui influencent les conditions de vie et les besoins énergétiques des ménages.

Conclusion

Cette étude a examiné la précarité et la vulnérabilité énergétiques au Québec, en explorant leurs définitions et les indicateurs pertinents. Nos recherches combinent une analyse quantitative utilisant les données d'Hydro-Québec et de Statistique Canada avec une revue de littérature nationale et internationale. Nous avons proposé un nouvel indice de vulnérabilité multidimensionnel s'appuyant sur la structure de la vulnérabilité en prenant en compte des facteurs d'exposition, d'adaptabilité et de sensibilité à la précarité énergétique. Cet indice a permis d'identifier les municipalités confrontées à une vulnérabilité énergétique accrue, offrant une approche plus nuancée que les mesures traditionnelles comme les indicateurs à 10 % ou 2M.

La méthode des k plus proches voisins et la méthode 2M ont donné des résultats distincts, soulignant l'importance de tenir compte à la fois des facteurs contextuels locaux et des disparités économiques plus larges. Nos résultats suggèrent que les politiques existantes ne parviennent pas à traiter adéquatement la nature multiforme de la vulnérabilité énergétique, soulignant la nécessité d'une approche plus holistique tenant compte de plusieurs sujets. D'abord, des données plus granulaires sur les caractéristiques du logement, les schémas de consommation d'énergie et les données démographiques des ménages amélioreraient considérablement la précision et la portée des évaluations futures. L'intégration de données autodéclarées permettrait de mieux saisir les expériences subjectives de la vulnérabilité énergétique.
De plus, notre analyse a révélé des tendances démographiques associées à une vulnérabilité énergétique accrue (âge, type de logement, milieu rural par rapport à urbain, etc.). Les programmes de soutien ciblés devraient tenir compte de ces facteurs.
La vulnérabilité énergétique doit être explicitement intégrée aux politiques sociales et économiques plus larges du Québec. En effet, des solutions plus complètes et équitables ne peuvent passer que par une définition officielle de la précarité énergétique dans la province.

Finalement, pour approfondir la compréhension de la précarité et de la vulnérabilité énergétiques au Québec, des recherches futures pourraient intégrer des données qualitatives, telles que des entrevues ou des groupes de discussion, afin de mieux cerner les expériences vécues par les ménages concernés.
En effet, un aspect fondamental de notre approche est la détermination des seuils de précarité énergétique. Le seuil de consommation (c_k) a été défini comme 80 % de la consommation médiane des ménages comparables dans la même zone, tandis que le seuil de revenu (r) a été établi au 40e percentile des municipalités avec le plus bas revenu médian par ménage.
Cependant, cette approche présente des limitations, car les valeurs de c_k, r et même le choix du nombre de voisins dans la méthode k-NN peuvent sembler arbitraires ou subjectives. Ces choix méthodologiques nécessitent une réflexion approfondie et pourraient potentiellement influencer les résultats de l'analyse.
Ainsi, nous suggérons que des travaux futurs investiguent ces dimensions en explorant des seuils plus dynamiques et en intégrant des méthodes d'analyse alternatives qui pourraient

offrir des perspectives différentes sur la précarité énergétique. Une telle recherche pourrait inclure l'ajustement des seuils en fonction des variations régionales ou saisonnières, ou l'exploration d'approches basées sur des données plus qualitatives pour enrichir la compréhension des facteurs contextuels influençant la précarité énergétique au Québec.

Une perspective longitudinale, suivant l'évolution de la vulnérabilité énergétique au fil du temps et l'impact des différentes interventions politiques, serait également précieuse pour éclairer les décisions futures. Enfin, il serait pertinent d'approfondir l'analyse des disparités régionales, en considérant les spécificités de chaque région du Québec pour mieux adapter les stratégies d'intervention.

Pour finir, nous aimerions insister sur l'aspect incomplet de cette étude. L'analyse quantitative présentée ici ne peut pleinement appréhender la complexité de la précarité énergétique au Québec et n'étudie que quelques aspects du sujet. Par exemple, elle ne saisit pas la situation des ménages habitant des logements précaires (cabanons, véhicules aménagés, etc.), confrontés à des difficultés d'accès aux services essentiels et souvent sous-représentés dans les données disponibles. De plus, elle ne prend pas en compte les crises du logement et du climat que traversent le Québec en ce moment et qui complexifient d'autant plus le sujet.
Cette limite souligne le besoin de méthodes de recherche plus diversifiées pour une compréhension globale et une action politique plus efficace.

Nous espérons que nos travaux contribueront à l'élaboration de politiques et d'interventions plus pertinentes pour soutenir les ménages vulnérables, peu importe leur situation.

Bibliographie

“Access and Affordability – Topics - IEA.” International Energy Agency. Accessed October 20, 2024. https://www.iea.org/topics/access-and-affordability.

Al Kez, Dlzar, Aoife Foley, Christopher Lowans, et Dylan Furszyfer Del Rio. 2024a. « Energy Poverty Assessment: Indicators and Implications for Developing and Developed Countries ». *Energy Conversion and Management* 307 (mai):118324. https://doi.org/10.1016/j.enconman.2024.118324.

2024b. « Energy Poverty Assessment: Indicators and Implications for Developing and Developed Countries ». *Energy Conversion and Management* 307 (mai):118324. https://doi.org/10.1016/j.enconman.2024.118324.

Bentley, Rebecca, Lyrian Daniel, Yuxi Li, Emma Baker, et Ang Li. 2023. « The Effect of Energy Poverty on Mental Health, Cardiovascular Disease and Respiratory Health: A Longitudinal Analysis ». *The Lancet Regional Health - Western Pacific* 35 (juin):100734. https://doi.org/10.1016/j.lanwpc.2023.100734.

Beslay, Christophe, Romain Gournet, et Marie-Christine Zélem. 2010. « Analyse sociotechnique comparée des dispositifs de réduction des situations de précarité énergétique et construction de stratégies d’intervention ciblées ». https://inis.iaea.org/collection/NCLCollectionStore/_Public/49/034/49034364.pdf.

Boardman, Brenda. 1991. *Fuel poverty: from cold homes to affordable warmth*. London ; New York: Belhaven Press.

Bouzarovski, Stefan, et Saska Petrova. 2015. « A Global Perspective on Domestic Energy Deprivation: Overcoming the Energy Poverty–Fuel Poverty Binary ». *Energy Research & Social Science* 10 (novembre):31-40. https://doi.org/10.1016/j.erss.2015.06.007.

Briot-Arthur, Sarah, Valerie Fournier, et Bruno Lee. 2024. « Quantifying Energy Poverty Vulnerability with Minimal Data – A Building Energy Simulation Approach ». *Energy and Buildings* 309 (avril): 114062. https://doi.org/10.1016/j.enbuild.2024.114062.

Burns, Iain. 2023. « 1 in 7 Canadian Households Choosing between Heating and Eating, Survey Finds ». PentictonNow. 2 novembre 2023. https://www.pentictonnow.com/watercooler/health/news/Health/1_in_7_Canadian_households_choosing_between_heating_and_eating_survey_finds/.

Buse, Chris, Louise Aubin, Natasha Fearing, Kiran Ghai, et Meagan Tibbits. 2023. « Assessing exposure, sensitivity, and adaptive capacity in Peel region ». Peel Region. https://www.peelregion.ca/climate-change/_media/climate-change-health-report.pdf.

Castaño-Rosa, Raúl, Jaime Solís-Guzmán, Carlos Rubio-Bellido, et Madelyn Marrero. 2019. « Towards a Multiple-Indicator Approach to Energy Poverty in the European Union: A Review ». *Energy and Buildings* 193 (juin):36-48. https://doi.org/10.1016/j.enbuild.2019.03.039.

CBC. 2023. “Cost of energy forcing many to go without necessities, StatsCan survey says.” November 16, 2023. https://www.cbc.ca/news/canada/calgary/statistics-canada-cost-of-living-affordability-crisis-1.7029769 .

Chen, Keyu, et Chao Feng. 2022. « Linking Housing Conditions and Energy Poverty: From a Perspective of Household Energy Self-Restriction ». *International Journal of Environmental Research and Public Health* 19 (14): 8254. https://doi.org/10.3390/ijerph19148254.

“Cold Weather Safety for Older Adults | National Institute on Aging.” 2024. National Institute on Aging. https://www.nia.nih.gov/health/safety/cold-weather-safety-older-adults.

Conservation Council of New Brunswick. “FINAL: Energy Poverty Factsheet 2024.” 2024. https://www.conservationcouncil.ca/wp-content/uploads/2023/04/FINAL-Energy-Poverty-Factsheet-2024.pdf.

Croon, T.M., J.S.C.M. Hoekstra, M.G. Elsinga, F. Dalla Longa, et P. Mulder. 2023. « Beyond Headcount Statistics: Exploring the Utility of Energy Poverty Gap Indices in Policy Design ». *Energy Policy* 177 (juin): 113579. doi:10.1016/j.enpol.2023.113579.

« curve_fit ». s. d. Python. https://docs.scipy.org/doc/scipy/reference/generated/scipy.optimize.curve_fit.html.

Das, Runa R., Mari Martiskainen, Lindsey M. Bertrand, et Julie L. MacArthur. 2022. « A Review and Analysis of Initiatives Addressing Energy Poverty and Vulnerability in Ontario, Canada ». *Renewable and Sustainable Energy Reviews* 165 (septembre):112617. https://doi.org/10.1016/j.rser.2022.112617.

Das, Runa R., Mari Martiskainen, et Grace Li. 2022. « Quantifying the Prevalence of Energy Poverty across Canada: Estimating Domestic Energy Burden Using an Expenditures Approach ». *Canadian Geographies / Géographies Canadiennes* 66 (3): 416-33. https://doi.org/10.1111/cag.12750.

Day, Rosie, Gordon Walker, et Neil Simcock. 2016. « Conceptualising Energy Use and Energy Poverty Using a Capabilities Framework ». *Energy Policy* 93 (juin):255-64. https://doi.org/10.1016/j.enpol.2016.03.019.

De Vries, R., et D. Blane. 2013. « Fuel Poverty and the Health of Older People: The Role of Local Climate ». *Journal of Public Health* 35 (3): 361-66. https://doi.org/10.1093/pubmed/fds094.

Department for Communities. “Fuel poverty.” Department for Communities. Accessed October 23, 2024. https://www.communities-ni.gov.uk/topics/housing/fuel-poverty.

Dionne, Salomé, David Heisel, and Jerry Situ. 2024. “Housing Statistics in Canada Estimation of Energy Poverty Rates Using the 2021 Census of Population.”

Statistique Canada. https://www150.statcan.gc.ca/n1/pub/46-28-0001/2024001/article/00001-eng.htm .

EPAH. 2022. « EPAH Energy Poverty National Indicators: Insights for a More Effective Measuring ». https://energy-poverty.ec.europa.eu/observatory/publications/epah-energy-poverty-national-indicators-insights-more-effective-measuring.

“EU statistics on income and living conditions - Eurostat.” European Commission. Accessed October 20, 2024. https://ec.europa.eu/eurostat/web/microdata/european-union-statistics-on-income-and-living-conditions.

Fonds de Recherche du Québec. 2023. “Prévenir et réduire la précarité énergétique : un enjeu de logement, de santé et d’équité pour la transition énergétique au Québec”. https://frq.gouv.qc.ca/projet/prevenir-et-reduire-la-precarite-energetique-un-enjeu-de-logement-de-sante-et-dequite-pour-la-transition-energetique-au-quebec/

“Fuel poverty methodology handbook 2020 LIHC.” 2020. GOuvernement du Royaume-Uni. https://assets.publishing.service.gov.uk/media/603fcdaee90e077dd08f15e6/Fuel_Poverty_Methodology_Handbook_2020_LIHC.pdf.

“Getting the measure of fuel poverty Executive summary.” 2012. Gouvernement du Royaume-Uni https://assets.publishing.service.gov.uk/media/5a79c0fde5274a684690bdc8/4664-exec-summary-fuel-pov-final-rpt.pdf .

González-Eguino, Mikel. 2015. « Energy Poverty: An Overview ». *Renewable and Sustainable Energy Reviews* 47 (juillet):377‑85. https://doi.org/10.1016/j.rser.2015.03.013.

Gouveia, João Pedro, Pedro Palma, et Sofia G. Simoes. 2019. « Energy Poverty Vulnerability Index: A Multidimensional Tool to Identify Hotspots for Local Action ». *Energy Reports* 5 (novembre):187‑201. https://doi.org/10.1016/j.egyr.2018.12.004.

Gouvernement du Canada. 2024. « Quebec Geographical Names Data ». https://natural-resources.canada.ca/earth-sciences/geography/download-geographical-names-data/9245.

Gouvernement du Québec. “Legisquebec.” L-7 - Loi visant à lutter contre la pauvreté et l'exclusion sociale. https://www.legisquebec.gouv.qc.ca/fr/document/lc/L-7.

Gouvernement du Québec. “Plan directeur en transition énergétique | Innovation et transition énergétiques.” Transition énergétique. https://transitionenergetique.gouv.qc.ca/plan-directeur-en-transition-energetique.

Gouvernement du Québec. 2024. « Liste des municipalités (csv) ». https://www.donneesquebec.ca/recherche/fr/dataset/repertoire-des-municipalites-du-quebec/resource/19385b4e-5503-4330-9e59-f998f5918363.

Government of Canada, Statistics Canada. 2023. « Le Quotidien — Enquête sociale canadienne : utilisation d’énergie ». 30 octobre 2023. https://www150.statcan.gc.ca/n1/daily-quotidien/231030/dq231030b-fra.htm.

Healy, Jonathan, et J. Peter Clinch. 2002. « FUEL POVERTY IN EUROPE: A CROSS-COUNTRY ANALYSIS USING A NEW COMPOSITE MEASUREMENT ». University College Dublin. https://www.researchgate.net/publication/237226014_FUEL_POVERTY_IN_EUROPE_A_CROSS-COUNTRY_ANALYSIS_USING_A_NEW_COMPOSITE_MEASUREMENT.

Healy, Jonathan D. 2017. *Housing, Fuel Poverty and Health*. 0 éd. Routledge. https://doi.org/10.4324/9781315253183.

“Hydro-Québec | L'hydroélectricité québécoise : propre, renouvelable et faible en GES.” Hydro-Quebec. Accessed October 20, 2024. https://www.hydroquebec.com/a-propos/notre-energie.html.

Hydro-Québec. 2022. « Historique des données de consommation d’électricité au Québec par secteur d’activité (municipalités) pour la période du 1er janvier 2016 au 31 décembre 2023 ». https://www.hydroquebec.com/documents-donnees/donnees-ouvertes/historique-consommation-secteur-activite/.

Institut de la Statistique du Québec. 2023. « Personnes en situation de faible revenu ». https://statistique.quebec.ca/vitrine/egalite/dimensions-egalite/revenu/personnes-situation-faible-revenu.

Kahouli, Sondès, et Shinichiro Okushima. 2021. « Regional Energy Poverty Reevaluated: A Direct Measurement Approach Applied to France and Japan ». *Energy Economics* 102 (octobre):105491. https://doi.org/10.1016/j.eneco.2021.105491.

Kantamneni, Abhi, and Brendan Haley. 2021. “A national energy poverty strategy for Canada?” Efficiency Canada. https://www.efficiencycanada.org/national-energy-poverty-strategy/.

“Le plan de mise en œuvre.” 2024. Gouvernement du Québec. https://www.quebec.ca/gouvernement/politiques-orientations/plan-economie-verte/plan-mise-en-oeuvre.

Llera-Sastresa, Eva, Sabina Scarpellini, Pilar Rivera-Torres, Juan Aranda, Ignacio Zabalza-Bribián, et Alfonso Aranda-Usón. 2017. « Energy Vulnerability Composite Index in Social Housing, from a Household Energy Poverty Perspective ». *Sustainability* 9 (5): 691. https://doi.org/10.3390/su9050691.

McKay, Stephen. 2004. « Poverty or Preference: What Do ‘Consensual Deprivation Indicators’ Really Mean? » *Fiscal Studies* 25 (2): 201-23. https://doi.org/10.1111/j.1475-5890.2004.tb00102.x

Mein, Tatiana Farane, André Luiz Veiga Gimenes, Miguel Edgar Morales Udaeta, Eduardo Mario Dias, et Stefania Gomes Relva. 2020. « Issues in Energy Vulnerability Assessment: Looking for a Sustainable Choice of Natural Resource for Power Generation ». In *The First World Energies Forum—Current and Future Energy Issues*, 30. MDPI. https://doi.org/10.3390/WEF-06929.

« Meteostat Python Package ». 2024. Python. https://pypi.org/project/meteostat/.

Middlemiss, Lucie. 2022. « Who Is Vulnerable to Energy Poverty in the Global North, and What Is Their Experience? » *WIREs Energy and Environment* 11 (6): e455. https://doi.org/10.1002/wene.455.

Moore, Richard. 2012. « Definitions of Fuel Poverty: Implications for Policy ». *Energy Policy* 49 (octobre):19-26. https://doi.org/10.1016/j.enpol.2012.01.057.

« Neighbr ». s. d. R. https://cran.r-project.org/web/packages/neighbr/vignettes/neighbr-help.html.

Nojedehi, Pedram, Burak Gunay, William O'Brien, Maya Papineau, Elie Azar, Marcel Schweiker, Gülsu Ulukavak Harputlugil, et Nese Ganiç Saglam. 2024. « Examining Disparities in Energy Poverty and Indoor Environmental Quality Satisfaction among Canadian Households ». *Energy Policy* 194 (novembre):114319. https://doi.org/10.1016/j.enpol.2024.114319.

Nord, Mark, et Linda S Kantor. 2006. « Seasonal Variation in Food Insecurity Is Associated with Heating and Cooling Costs among Low-Income Elderly Americans ». *The Journal of Nutrition* 136 (11): 2939-44. https://doi.org/10.1093/jn/136.11.2939.

Nussbaumer, Patrick, Morgan Bazilian, et Vijay Modi. 2012. « Measuring Energy Poverty: Focusing on What Matters ». *Renewable and Sustainable Energy Reviews* 16 (1): 231-43. https://doi.org/10.1016/j.rser.2011.07.150.

Palmer, Guy, Tom MacInnes, et Peter Kenway. 2008. « Cold and Poor: An Analysis of The Link between Fuel Poverty and Low Income ». New Policy Institute. https://www.fuelpovertylibrary.info/content/cold-and-poor-analysis-link-between-fuel-poverty-and-low-income-0.

Perreault, Mathieu. 2024. "La précarité énergétique touche davantage les plus pauvres." LaPresse.ca, April 26, 2024. https://www.lapresse.ca/actualites/sciences/2024-04-26/la-precarite-energetique-touche-davantage-les-plus-pauvres.php.

Petrova, Saska, Michael Gentile, Ilkka Henrik Mäkinen, et Stefan Bouzarovski. 2013. « Perceptions of Thermal Comfort and Housing Quality: Exploring the Microgeographies of Energy Poverty in Stakhanov, Ukraine ». *Environment and Planning A: Economy and Space* 45 (5): 1240-57. https://doi.org/10.1068/a45132.

Pineau, P.-O., et J. Whitmore. 2024. « État de l'énergie au Québec 2024 ». Chaire de gestion du secteur de l'énergie, HEC Montréal. https://energie.hec.ca/wp-content/uploads/2024/03/EEQ2024_WEB.pdf.

Pino-Mejías, Rafael, Alexis Pérez-Fargallo, Carlos Rubio-Bellido, et Jesús A. Pulido-Arcas. 2018. « Artificial Neural Networks and Linear Regression Prediction Models for Social Housing Allocation: Fuel Poverty Potential Risk Index ». *Energy* 164 (décembre):627-41. https://doi.org/10.1016/j.energy.2018.09.056.

Primc, Kaja, Miha Dominko, et Renata Slabe-Erker. 2021. « 30 Years of Energy and Fuel Poverty Research: A Retrospective Analysis and Future Trends ». *Journal of Cleaner Production* 301 (juin):127003. https://doi.org/10.1016/j.jclepro.2021.127003.

Rezaei, Maryam. 2017. « Power to the People : Thinking (and Rethinking) Energy Poverty in British Columbia, Canada ». https://doi.org/10.14288/1.0351974.

Riva, Mylene. 2024a. « La précarité énergétique au Canada : une question d'inégalités sociales et géographiques ». *VRM – Villes Régions Monde* Répertoire de recherche Villes, climat et inégalités (avril). https://www.vrm.ca/la-precarite-energetique-au-canada-une-question-dinegalites-sociales-et-geographiques/#:~:text=En%20effet%2C%20un%20m%C3%A9nage%20canadien,et%20coll.%2C%202012).

Riva, Mylene. 2024b. « Précarité énergétique : un déterminant négligé de la santé et de la résilience climatique au Canada ». *VRM – Villes Régions Monde* Répertoire de recherche Villes, climat et inégalités. https://www.vrm.ca/precarite-energetique-un-determinant-neglige-de-la-sante-et-de-la-resilience-climatique-au-canada/.

Riva, Mylene, Ellen Grubbs, et Sebastien Breau. 2024. « The Geography of Energy Poverty in Canada: Spatial Clustering and Inequalities at the Municipal Level ». *Energy Policy* 195 (décembre):114298. https://doi.org/10.1016/j.enpol.2024.114298

.

Riva, Mylene, Sophie Kingunza Makasi, Philippe Dufresne, Kimberley O'Sullivan, et Megan Toth. 2021. « Energy Poverty in Canada: Prevalence, Social and Spatial

Distribution, and Implications for Research and Policy ». *Energy Research & Social Science* 81 (novembre):102237. https://doi.org/10.1016/j.erss.2021.102237.

Scarpellini, Sabina, Pilar Rivera-Torres, Inés Suárez-Perales, et Alfonso Aranda-Usón. 2015. « Analysis of Energy Poverty Intensity from the Perspective of the Regional Administration: Empirical Evidence from Households in Southern Europe ». *Energy Policy* 86 (novembre):729-38. https://doi.org/10.1016/j.enpol.2015.08.009.

Siksnelyte-Butkiene, Indre, Dalia Streimikiene, Vidas Lekavicius, et Tomas Balezentis. 2021. « Energy Poverty Indicators: A Systematic Literature Review and Comprehensive Analysis of Integrity ». *Sustainable Cities and Society* 67 (avril):102756. https://doi.org/10.1016/j.scs.2021.102756.

Sokołowski, Jakub, Piotr Lewandowski, Aneta Kiełczewska, et Stefan Bouzarovski. 2020. « A Multidimensional Index to Measure Energy Poverty: The Polish Case ». *Energy Sources, Part B: Economics, Planning, and Policy* 15 (2): 92-112. https://doi.org/10.1080/15567249.2020.1742817.

Statistique Canada. 2017. « Recensement de 2016 ». https://www150.statcan.gc.ca/n1/en/catalogue/98-401-X2016044.

Statistique Canada. 2022. « Recensement de 2021 ». https://www150.statcan.gc.ca/n1/en/catalogue/98-316-X2021001.

Statistique Canada. 2024. « Low income statistics by age, sex and economic family type (Québec) ». https://www150.statcan.gc.ca/t1/tbl1/en/tv.action?pid=1110013501&pickMembers%5B0%5D=1.14&cubeTimeFrame.startYear=2015&cubeTimeFrame.endYear=2022&referencePeriods=20150101%2C20220101

Tardy, François, et Bruno Lee. 2019. « Building Related Energy Poverty in Developed Countries – Past, Present, and Future from a Canadian Perspective ». *Energy and Buildings* 194 (juillet):46-61. https://doi.org/10.1016/j.enbuild.2019.04.013.

“The Vulnerability Sourcebook.” 2014. https://health.bmz.de/wp-content/uploads/page/Vulnerability_Sourcebook__1_.pdf.

Thomson, Harriet, Stefan Bouzarovski, et Carolyn Snell. 2017. « Rethinking the Measurement of Energy Poverty in Europe: A Critical Analysis of Indicators and Data ». *Indoor and Built Environment* 26 (7): 879-901. https://doi.org/10.1177/1420326X17699260.

Thomson, Harriet, et Carolyn Snell. 2013. « Quantifying the Prevalence of Fuel Poverty across the European Union ». *Energy Policy* 52 (janvier):563-72. https://doi.org/10.1016/j.enpol.2012.10.009.

Trinomics. 2016. « Selecting Indicators to Measure Energy Poverty ». Commission Européenne. https://trinomics.eu/project/selecting-indicators-to-measure-energy-poverty/

Rylan. 2023. “Electricity Prices in Canada 2023.” energyhub.org. https://www.energyhub.org/electricity-prices/.

« Vulnerability Reduction and the Community-Based Approach ». 2003. In *Natural Disaster and Development in a Globalizing World*, par Katrina Allen. Hoboken: Taylor and Francis.

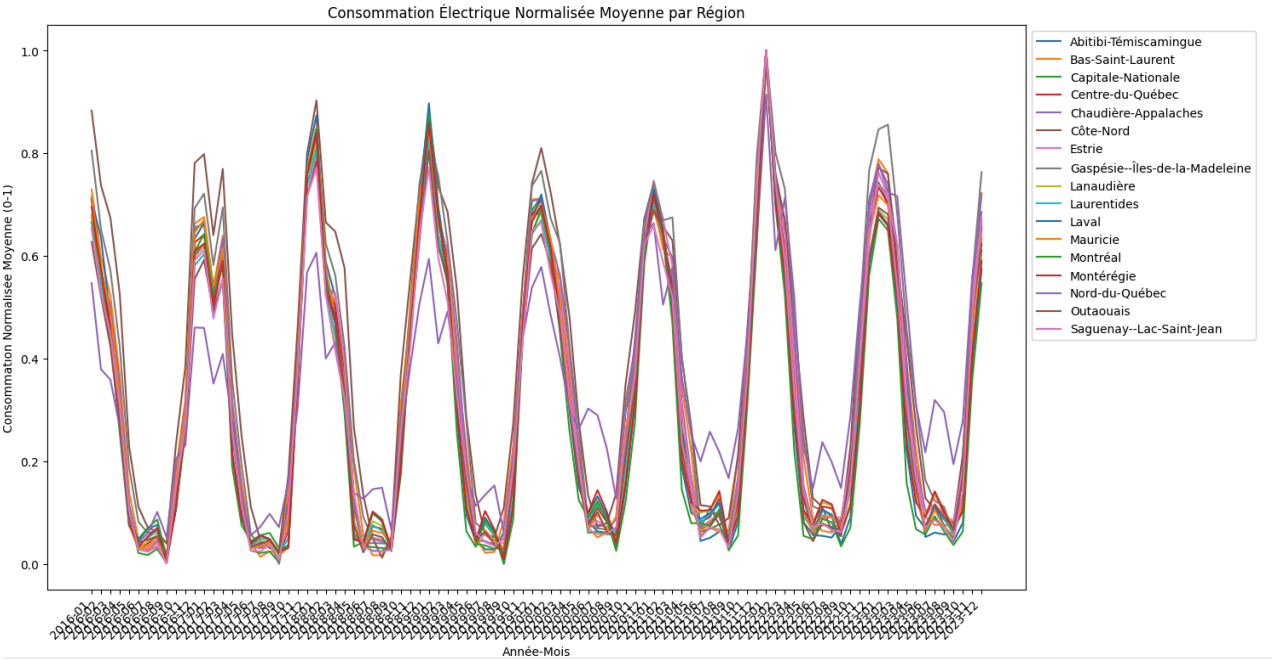
Xu, Wangzi, Baiwei Xie, Bailu Lou, Weizheng Wang, et Yuzi Wang. 2022. « Assessing the effect of energy poverty on the mental and physical health in China—Evidence from China family panel studies ». *Frontiers in Energy Research* 10 (septembre):944415. https://doi.org/10.3389/fenrg.2022.944415.

Annexe

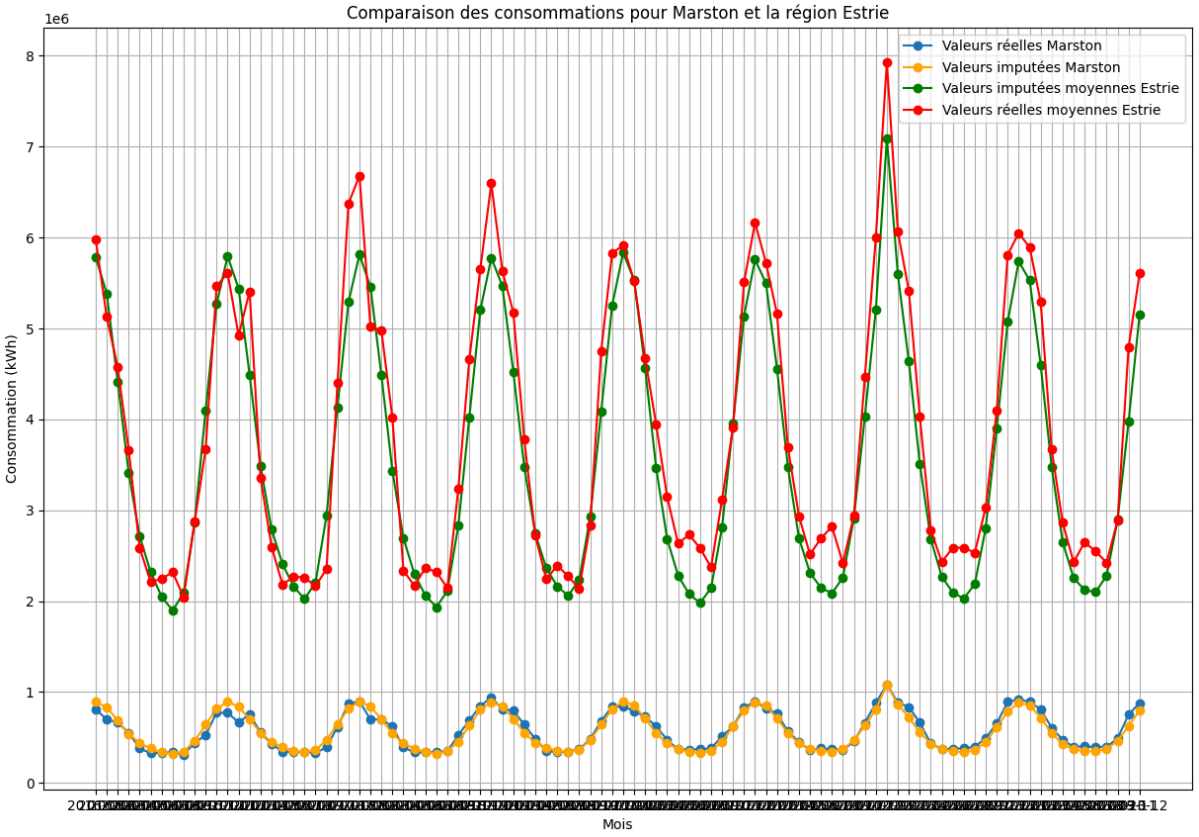
Annexe 1: Code

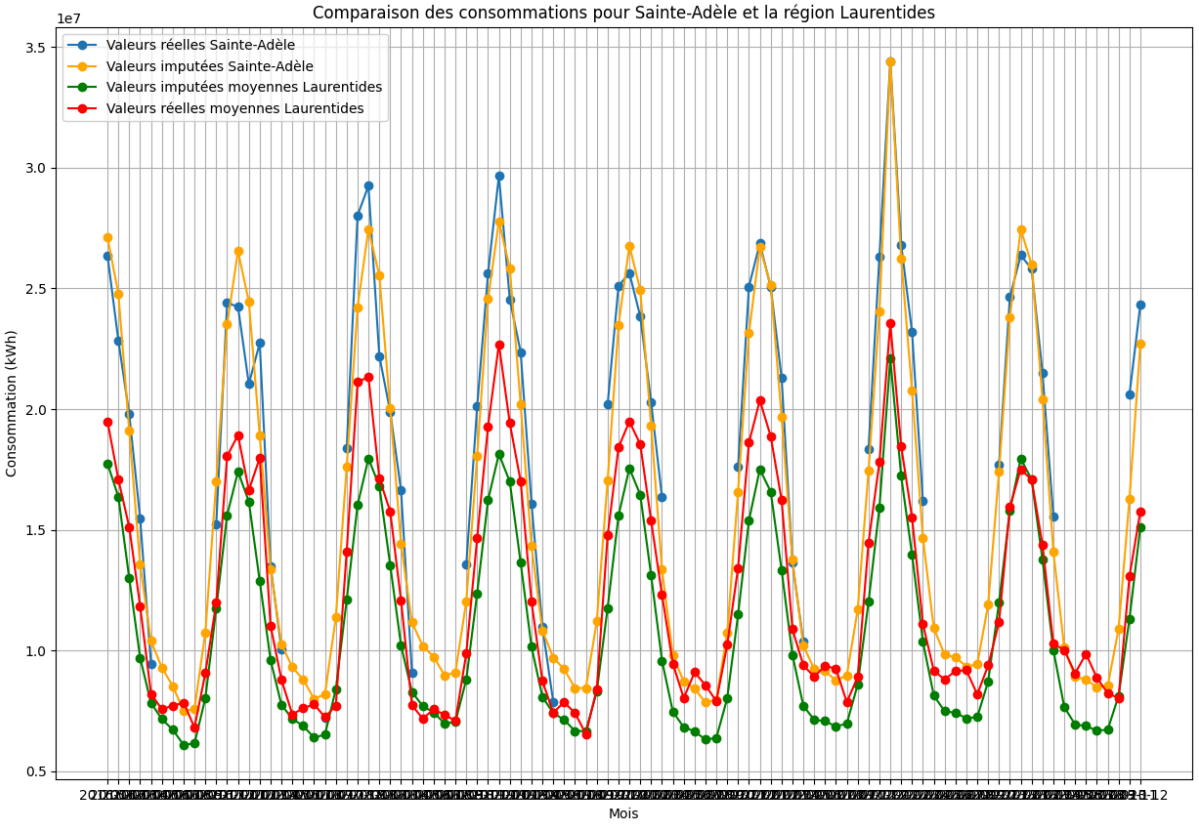
L'entièreté du code peut être retrouvé dans ce dossier Google Colab suivant : https://drive.google.com/drive/folders/1a_ksBX6GamUoZyeEZ4DNrZKhvvatECcN?usp=drive_link ainsi que dans les pages qui suivent l'annexe 4.

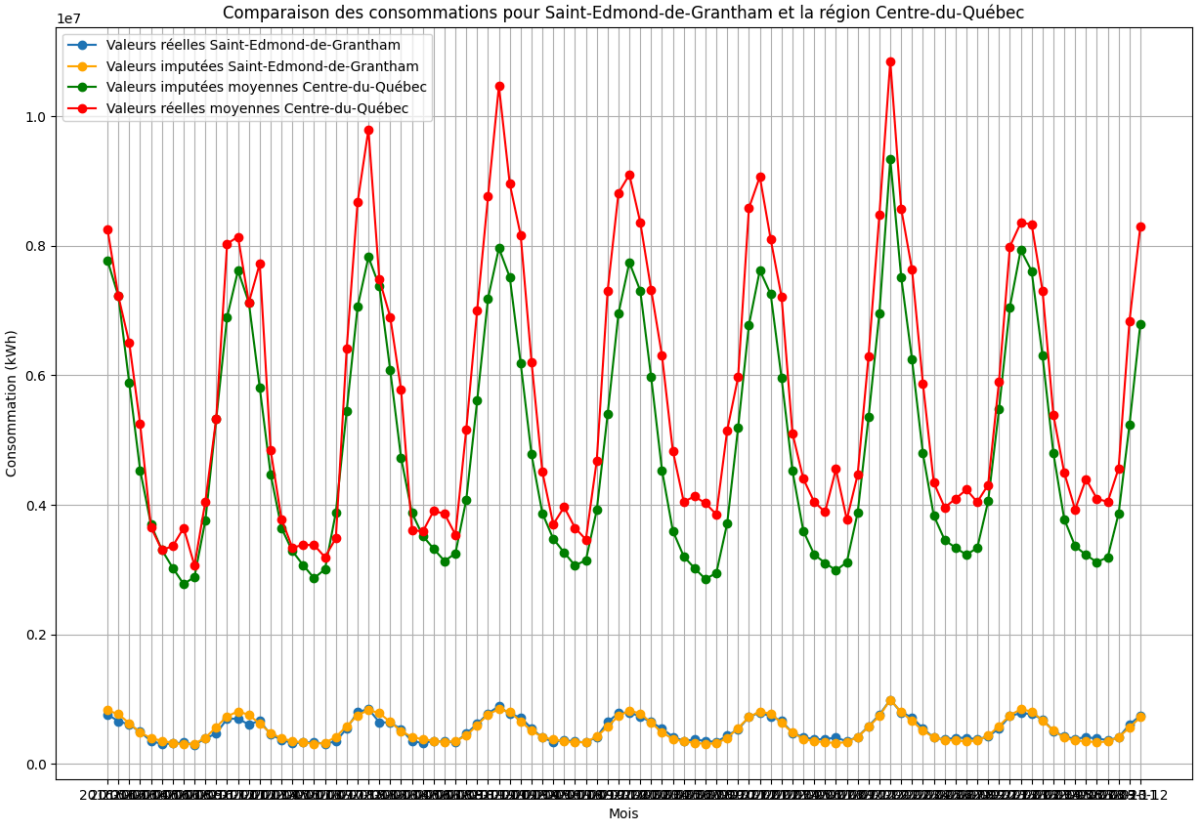
Annexe 2 : Courbes de la consommation normalisée moyenne des municipalités par région



Annexe 3 : Exemples de municipalités sélectionnées aléatoirement pour lesquelles on compare leurs valeurs réelles de consommation d'électricité à leurs valeurs imputées (ainsi que celles de leur région)





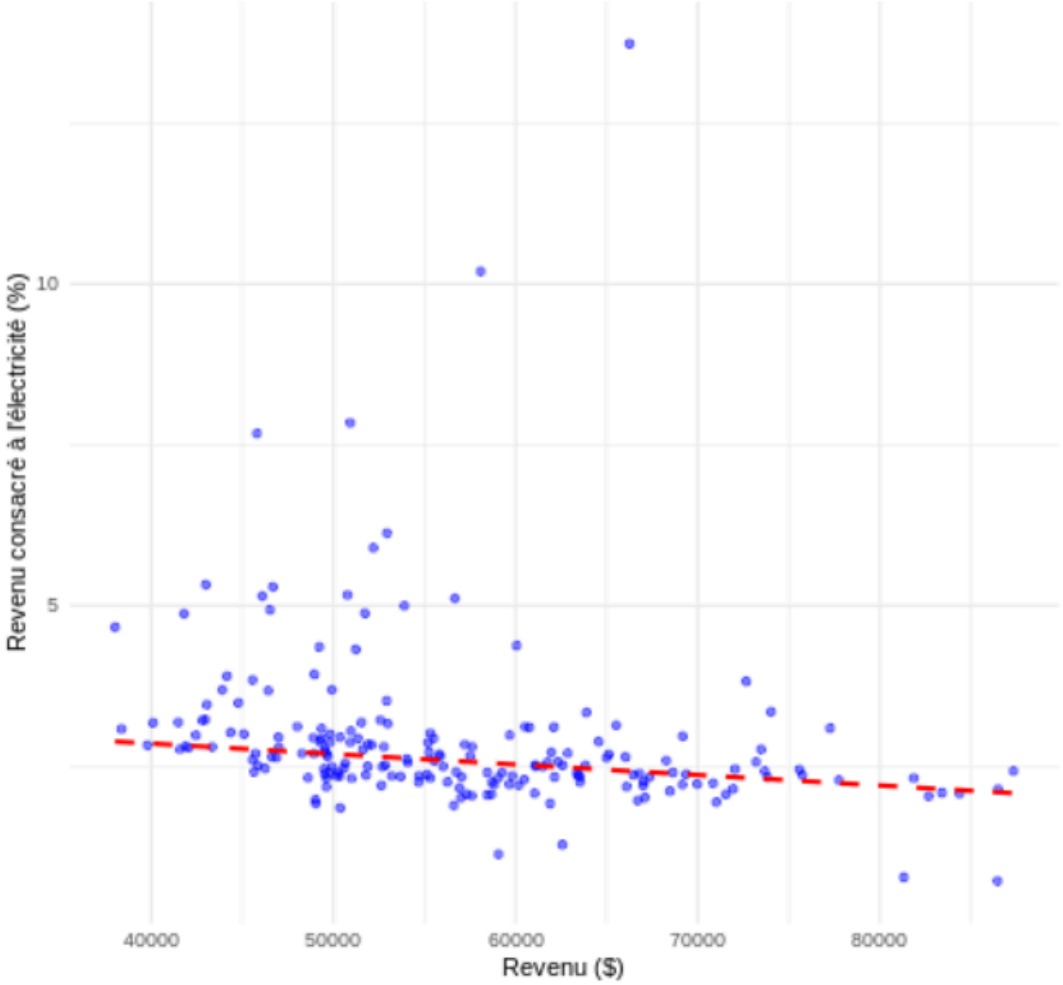

Comparaison des consommations pour Saint-Edmond-de-Grantham et la région Centre-du-Québec
1e7
Valeurs réelles Saint-Edmond-de-Grantham
Valeurs imputées Saint-Edmond-de-Grantham
Valeurs imputées moyennes Centre-du-Québec
Valeurs réelles moyennes Centre-du-Québec
Consommation (kWh)
0.0
0.2
0.4
0.6
0.8
1.0
Mois

Annexe 4 : Régressions linéaires

Annexe 4.1 : régressions linéaires robustes pour la variable “revenu”

Annexe 4.1.1 : Résultats des régressions linéaires robustes pour la variable “revenu” pour la part du revenu consacré à l’électricité

Nuage de points des municipalités selon le revenu après impôts médian des ménages en abscisse et du pourcentage du revenu consacré à l’électricité en ordonnée

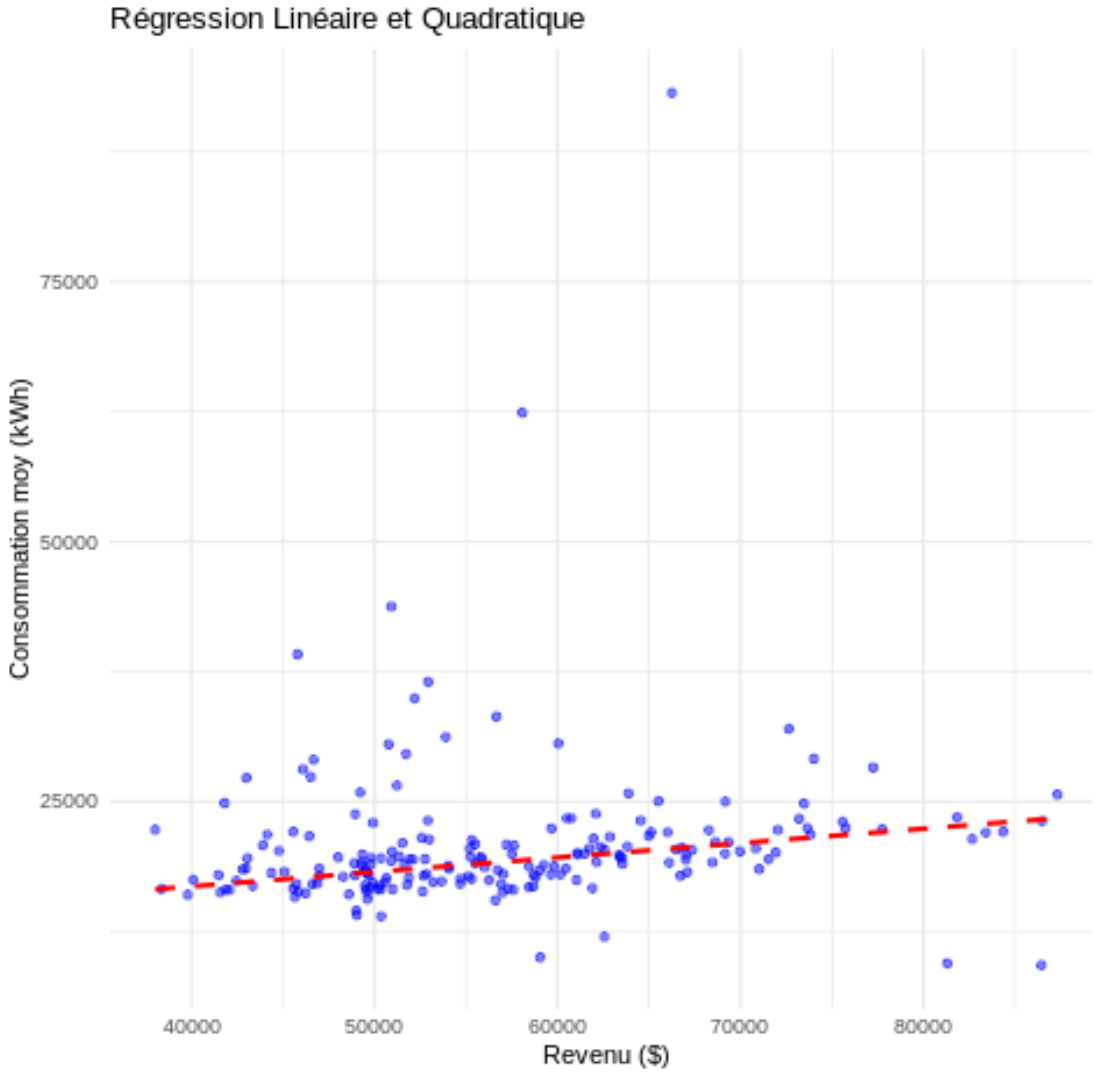


Sortie R de la régression linéaire robuste revenu après impôts médian des ménages comme variable explicative du pourcentage du revenu consacré à l’électricité en ordonnée

```
Call:
lmRob(formula = pourc_rev_conso ~ revenu, data = data_sul

Residuals:
     Min       1Q   Median       3Q      Max
-1.41031 -0.22635  0.01603  0.34932 11.30884

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept)  3.508e+00  1.686e-01  20.808  < 2e-16 ***
revenu      -1.630e-05  2.897e-06  -5.626 6.27e-08 ***
```

Annexe 4.1.2 : Régressions linéaires robustes pour la variable “revenu” mais pour les dépenses d’électricité en kWh

Nuage de points des municipalités selon le revenu après impôts médian des ménages en abscisse et de la consommation moyenne d’électricité en kWh en ordonnée



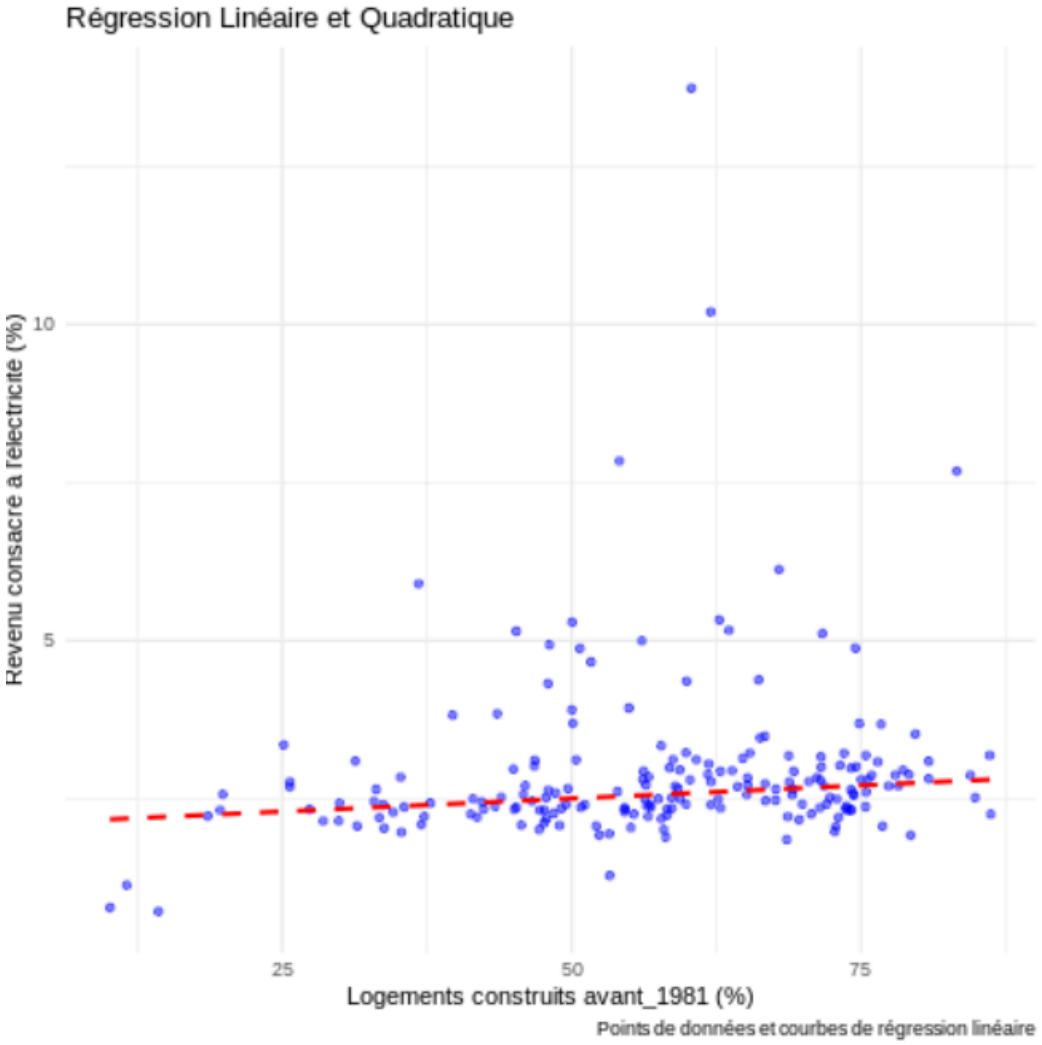
Sortie R de la régression linéaire robuste revenu après impôts médian des ménages comme variable explicative de la consommation moyenne d'électricité en kWh en ordonnée

```
Call:
lmRob(formula = consommation_moy ~ revenu, data = data_sub)

Residuals:
     Min       1Q   Median       3Q      Max
-14011.8  -1160.7    150.6   1712.3  72668.9

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept) 1.128e+04  8.863e+02  12.724   <2e-16 ***
revenu      1.387e-01  1.532e-02   9.056   <2e-16 ***
```

Annexe 4.2 : Résultats des régressions linéaires robustes pour la variable "logements construits avant 1981"

Annexe 4.2.1 : Résultats des régressions linéaires robustes pour la variable "logements construits avant 1981" pour la part du revenu consacré à l'électricité

Nuage de points des municipalités selon le pourcentage de logements construits avant 1981 en abscisse et du pourcentage du revenu consacré à l'électricité en ordonnée



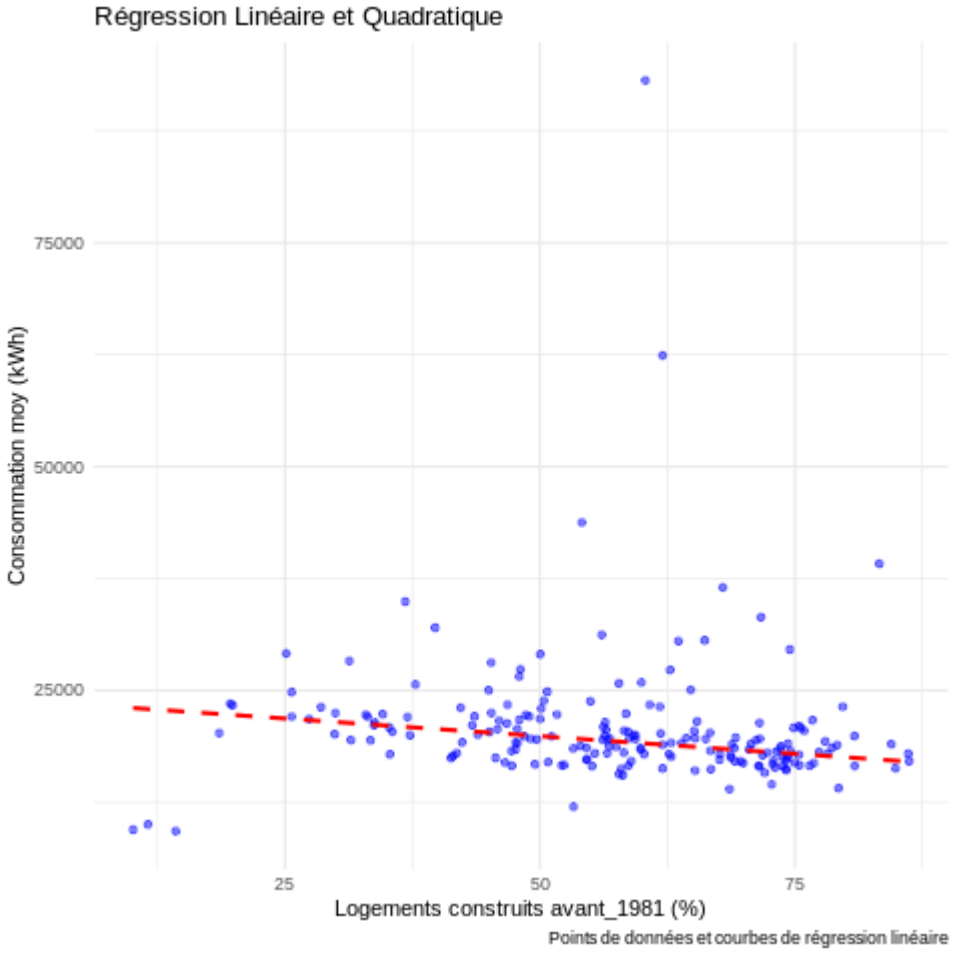
Sortie R de la régression linéaire robuste du le pourcentage de logements construits avant 1981 comme variable explicative du pourcentage du revenu consacré à l'électricité en ordonnée

```
Call:
lmRob(formula = pourc_rev_conso ~ logement_construit_avant_1981,
    data = data_sub)

Residuals:
    Min      1Q  Median      3Q     Max
-1.4907 -0.1997  0.0360  0.3832 11.1474

Coefficients:
                              Estimate Std. Error t value Pr(>|t|)
(Intercept)                   2.090478   0.117560  17.782  < 2e-16 ***
logement_construit_avant_1981 0.008267   0.001954   4.231 3.57e-05 ***
```

Annexe 4.2.2 : Résultats des régressions linéaires robustes pour la variable “logements construits avant 1981” mais pour les dépenses d’électricité en kWh

N.B : Dans les graphiques suivants, on observe que, plus la proportion de bâtiments construits avant 1981 augmente, plus la consommation d'électricité moyenne des ménages de cette municipalité diminue. Ce constat contredit le consensus établi par la recherche sur la pauvreté énergétique.

Nuage de points des municipalités selon le pourcentage de logements construits avant 1981 en abscisse et la consommation moyenne d’électricité en kWh en ordonnée

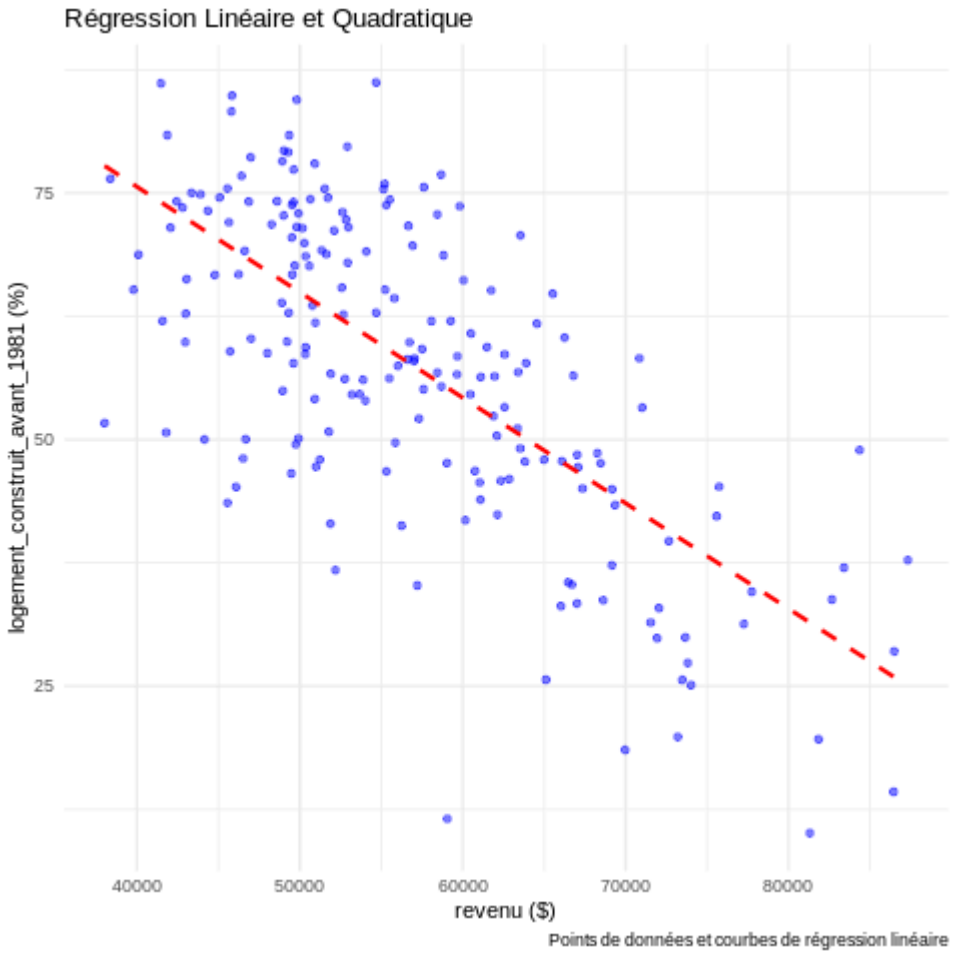


Sortie R de la régression linéaire robuste du le pourcentage de logements construits avant 1981 comme variable explicative de la consommation moyenne d’électricité en kWh en ordonnée

```
Call:
lmRob(formula = consommation_moy ~ logement_construit_avant_1981,
    data = data_sub)

Residuals:
   Min     1Q Median     3Q    Max
-13590  -1260    252   1799  74083

Coefficients:
                             Estimate Std. Error t value Pr(>|t|)
(Intercept)                  23804.29     647.38   36.77  < 2e-16 ***
logement_construit_avant_1981   -78.73      10.77   -7.31 6.53e-12 ***
```

Nous pouvons expliquer ces résultats de la manière suivante : il est possible que cette variable soit fortement corrélée au revenu. En effet, les ménages les plus pauvres, résidant généralement dans des logements plus anciens, soient contraints de limiter leur consommation d'électricité, même si leurs besoins théoriques sont plus élevés .
Cette hypothèse semble être soutenue par les résultats d'une régression linéaire robuste avec le revenu comme variable explicative et le pourcentage de logements construits avant 1981 comme variable dépendante (voir nuage de points qui suit).

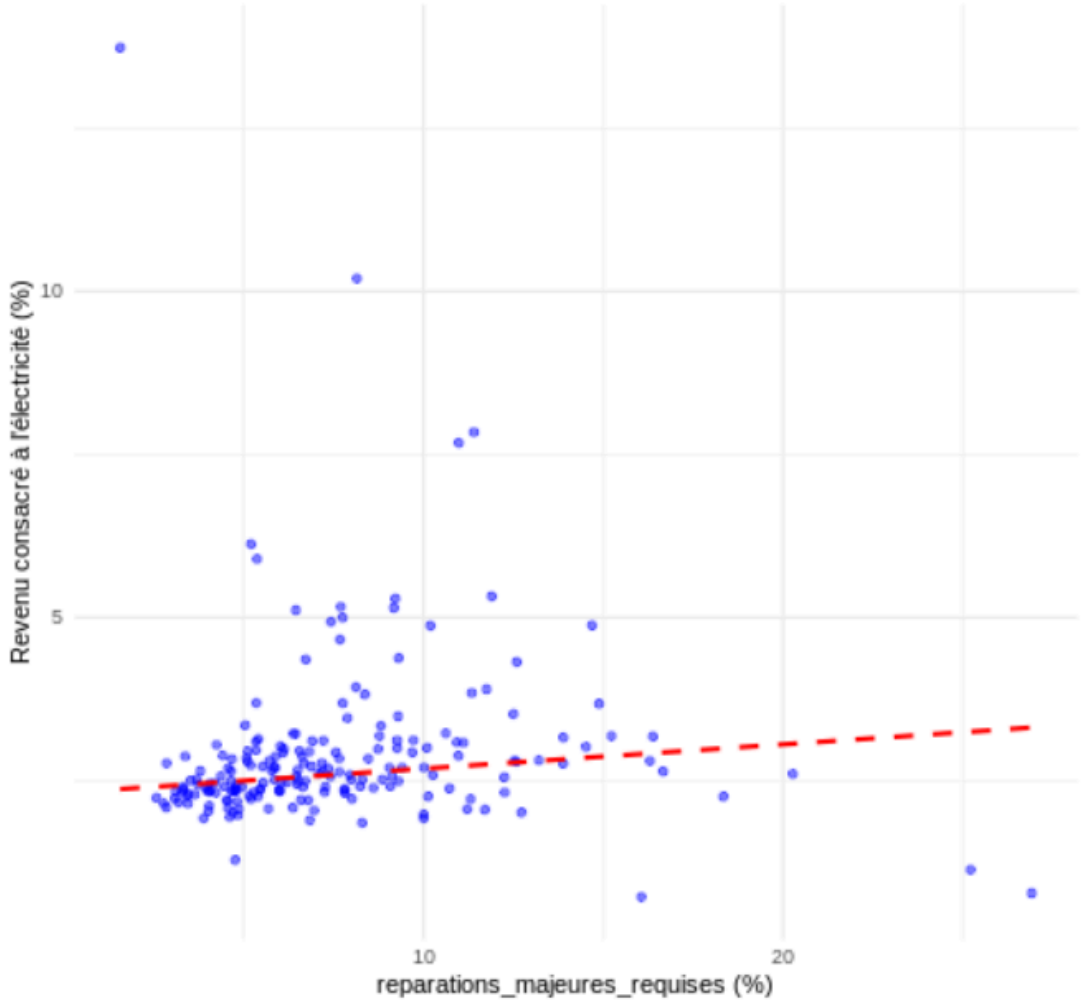
Nuage de points des municipalités selon le revenu en abscisse et le pourcentage de logements construits avant 1981 en ordonnée



Annexe 4.3 : Résultats des régressions linéaires robustes pour la variable “réparations majeures requises”

Annexe 4.3.1 : Résultats des régressions linéaires robustes pour la variable “réparations majeures requises” pour la part du revenu consacré à l’électricité

Nuage de points des municipalités selon le pourcentage de réparations majeures requises en abscisse et du pourcentage du revenu consacré à l’électricité en ordonnée



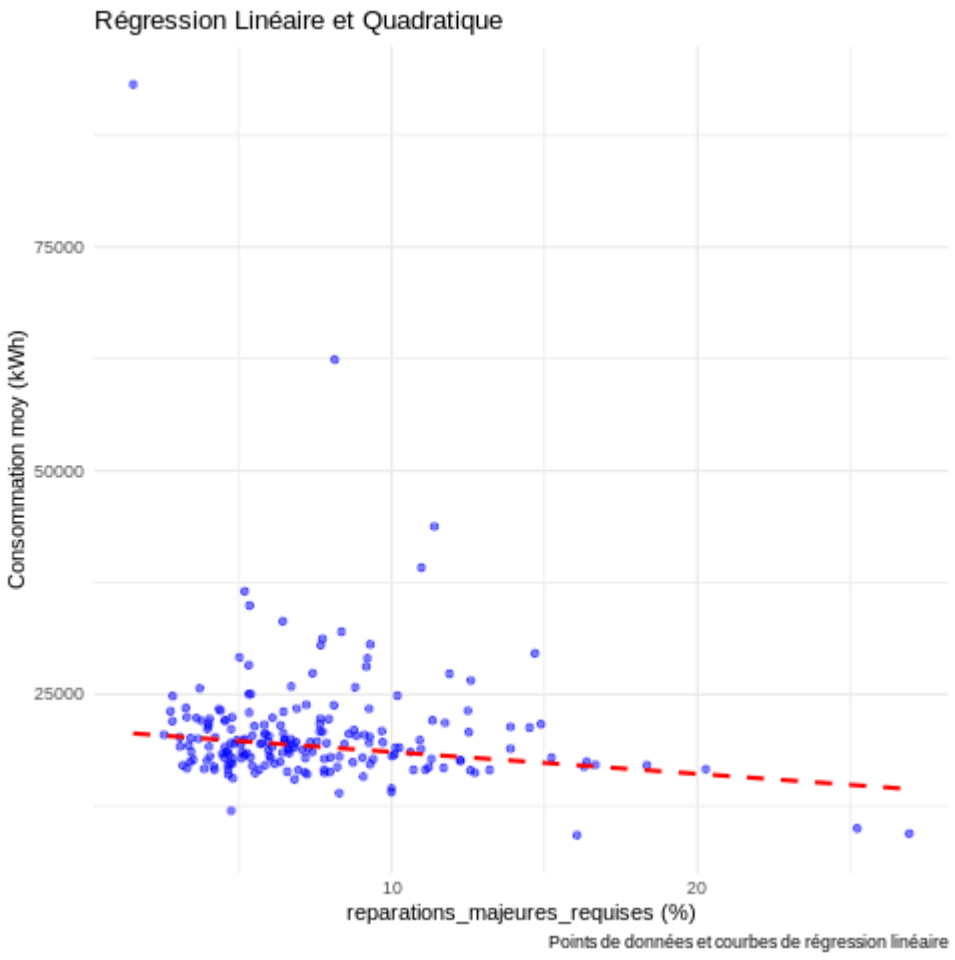
Sortie R de la régression linéaire robuste du pourcentage de réparations majeures requises comme variable explicative du pourcentage du revenu consacré à l’électricité

```
Call:
lmRob(formula = pourc_rev_conso ~ reparations_majeures_requises,
    data = data_sub)

Residuals:
      Min        1Q    Median        3Q       Max
-2.537002 -0.204016 -0.005636  0.384887 11.365607

Coefficients:
                             Estimate Std. Error t value Pr(>|t|)
(Intercept)                   2.31291    0.07877  29.364  < 2e-16 ***
reparations_majeures_requises 0.03718    0.01001   3.713 0.000267 ***
```

Annexe 4.3.2 : Résultats des régressions linéaires robustes pour la variable “réparations majeures requises” mais pour les dépenses d’électricité en kWh

NB : De la même manière que pour les logements construits avant 1981, on remarque une différence entre nos résultats et le consensus concernant le lien entre la consommation d’électricité et les réparations que nécessitent les logements d’une zone. En effet, selon ce dernier, plus il y a de réparations majeures requises, plus la consommation d’électricité augmente. Or, ce n’est pas le cas ici.

Nuage de points des municipalités selon le pourcentage de réparations majeures requises en abscisse et la consommation moyenne d’électricité en kWh en ordonnée

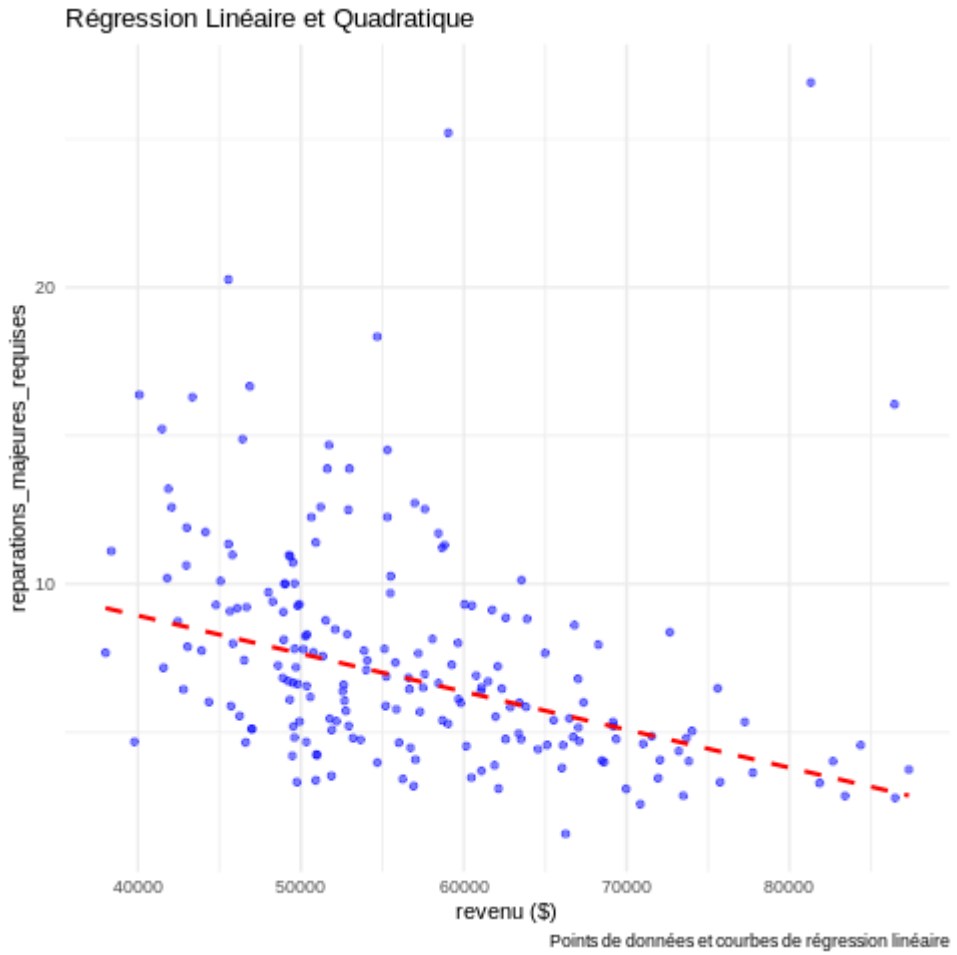


Sortie R de la régression linéaire robuste du pourcentage de réparations majeures requises comme variable explicative et la consommation moyenne d'électricité en kWh en ordonnée

```
Call:
lmRob(formula = consommation_moy ~ reparations_majeures_requises,
    data = data_sub)

Residuals:
    Min      1Q  Median      3Q     Max
-7854.2 -1515.2   107.4  2253.5 72517.2

Coefficients:
                            Estimate Std. Error t value Pr(>|t|)
(Intercept)                 21002.51     510.33  41.154  < 2e-16 ***
reparations_majeures_requises  -245.12      62.43  -3.926 0.000119 ***
```

Là aussi, ous pouvons expliquer ces résultats en avançant que les ménages les plus pauvres, résidant généralement dans des logements en plus mauvais état, soient contraints de limiter leur consommation d'électricité, même si leurs besoins théoriques sont plus élevés .
Cette hypothèse semble être soutenue par les résultats d'une régression linéaire robuste avec le revenu comme variable explicative et le pourcentage de logements nécessitant des réparations majeures comme variable dépendante (voir nuage de points qui suit).

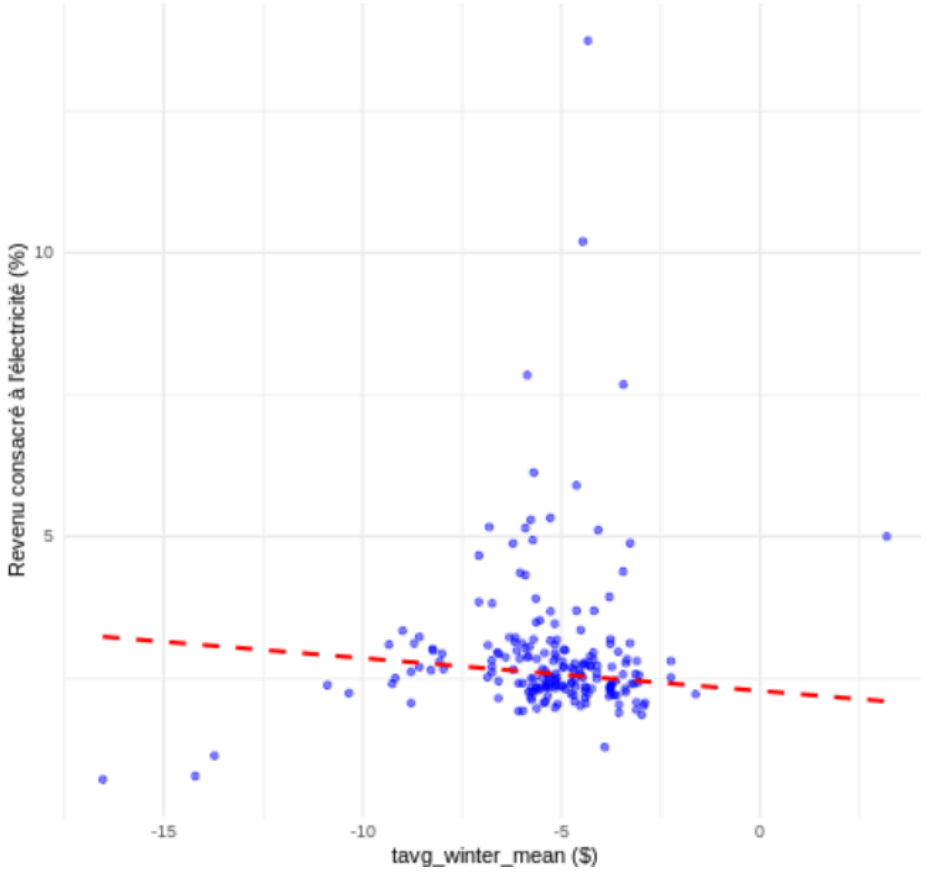
Nuage de points des municipalités selon le revenu en abscisse et le pourcentage de logements nécessitant des réparations majeures en ordonnée



Annexe 4.4 : Résultats des régressions linéaires robustes pour la variable “tavg_winter_mean” (température moyenne en hiver)

Annexe 4.4.1 : Résultats des régressions linéaires robustes pour la variable “tavg_winter_mean” (température moyenne en hiver) pour la part du revenu consacré à l’électricité

Nuage de points des municipalités selon la température moyenne en hiver des municipalités en abscisse et du revenu consacré à l’électricité en ordonnée

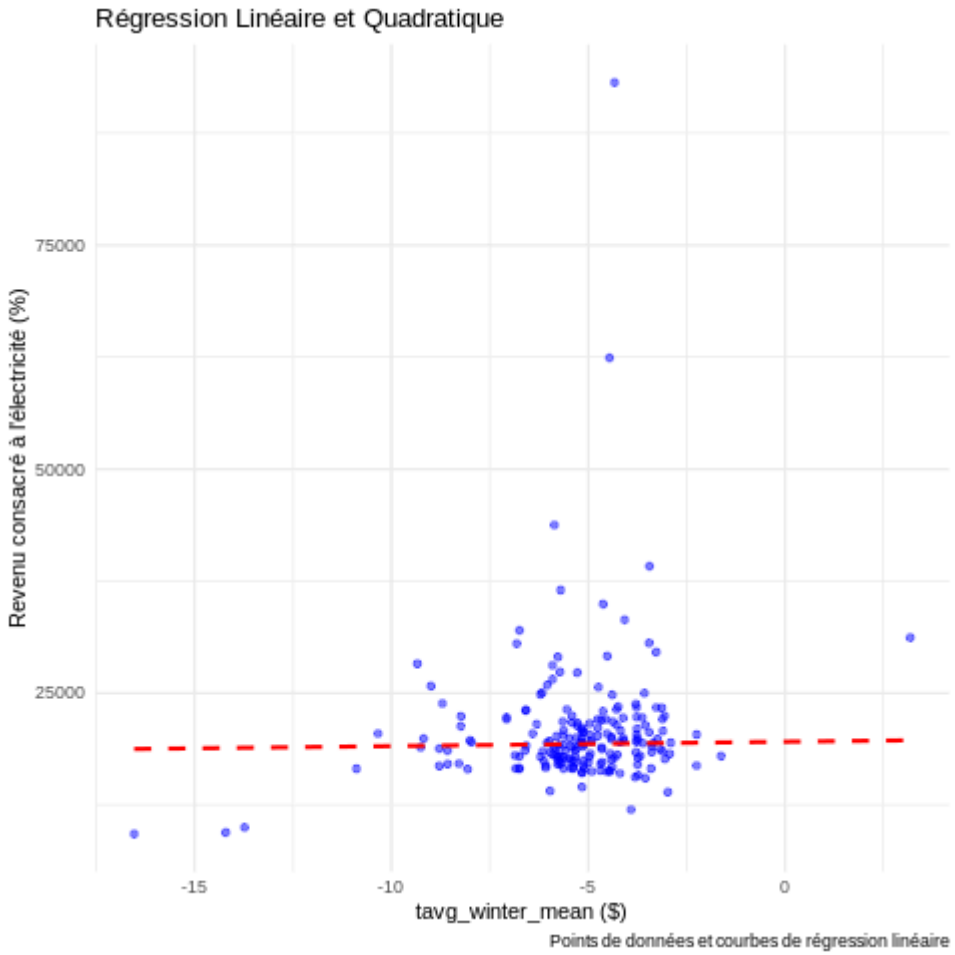


Sortie R de la régression linéaire robuste de la température moyenne en hiver comme variable explicative du pourcentage du revenu consacré à l'électricité

```
Call:
lmRob(formula = pourc_rev_conso ~ tavg_winter_mean, data = data_sub)

Residuals:
     Min       1Q   Median       3Q      Max
-2.51467 -0.24102  0.01163  0.40192 11.20895

Coefficients:
                 Estimate Std. Error t value Pr(>|t|)
(Intercept)       2.27752    0.11556  19.709  < 2e-16 ***
tavg_winter_mean -0.05776    0.02119  -2.726  0.00699 **
```

Annexe 4.4.2 : Résultats des régressions linéaires robustes pour la variable "tavg_winter_mean" (température moyenne en hiver) pour la part du revenu consacré à l'électricité

Nuage de points des municipalités selon la température moyenne en hiver des municipalités en abscisse et la consommation moyenne d'électricité en kWh en ordonnée



Sortie R de la régression linéaire robuste de la température moyenne en hiver comme variable explicative de la consommation moyenne d'électricité en kWh en ordonnée

```
Call:
lmRob(formula = consommation_moy ~ tavg_winter_mean, data = data_sub)

Residuals:
     Min       1Q   Median       3Q      Max
-9486.13 -1764.26    32.53  2482.66 73794.42

Coefficients:
                Estimate Std. Error t value Pr(>|t|)
(Intercept)     19555.03     766.72  25.505   <2e-16 ***
tavg_winter_mean   49.11     141.63   0.347    0.729
```

CODE

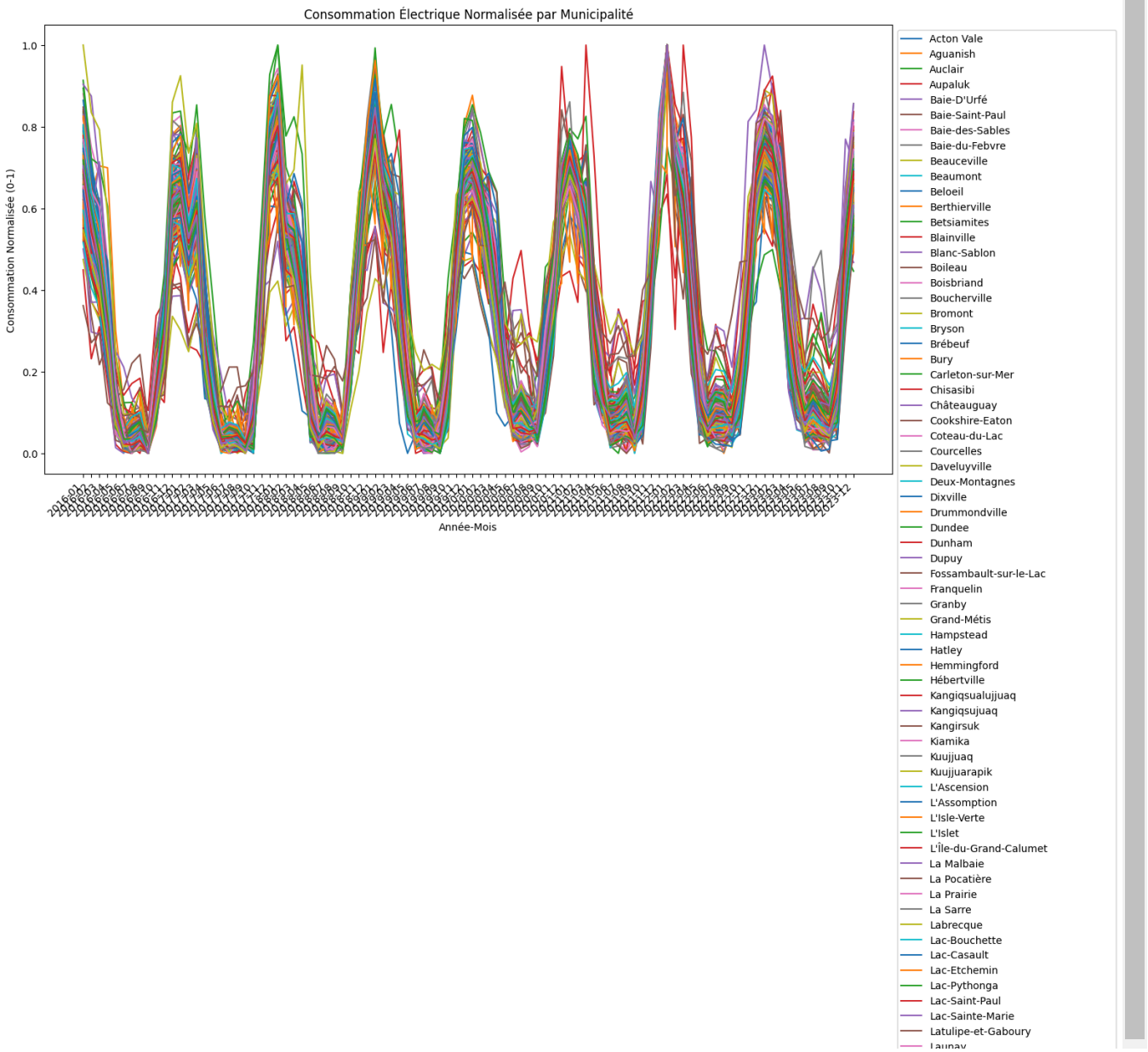
drive + librairies

↳ *5 cellules masquées*

exploration muncipalités connues + sin

```
# garder seulement les NOM_MUNICIPALITE qui ont tous leur Total (kWh) non null+ retirer Montréal

municipalities_with_all_non_null_total = elec_res.groupby('NOM_MUNICIPALITE')['Total (kWh)'].apply(lambda x: x.notnull().all()).loc[lambd
elec_res_no_null = elec_res[elec_res['NOM_MUNICIPALITE'].isin(municipalities_with_all_non_null_total)]

# Retirer 'Montréal'
elec_res_no_null = elec_res_no_null[elec_res_no_null['NOM_MUNICIPALITE'] != 'Montréal']


# faire un graphique en abcisse ANNEE_MOIS et en ordonnée la consommation normalisée sur la municipalité (entre 0 et 1) des Total (kWh) d

from sklearn.preprocessing import MinMaxScaler


elec_grouped = elec_res_no_null.groupby(['NOM_MUNICIPALITE', 'ANNEE_MOIS', 'REGION_ADM_QC_TXT'])['Total (kWh)'].sum().reset_index()

# Normaliser les valeurs

scaler = MinMaxScaler()
def apply_scaler(group):
    group['Consommation_Normalisee'] = scaler.fit_transform(group[['Total (kWh)']])
    return group

elec_grouped_normalized = elec_grouped.groupby('NOM_MUNICIPALITE').apply(apply_scaler).reset_index(drop=True)
# Create a plot
import matplotlib.pyplot as plt

plt.figure(figsize=(15, 8))

for municipality in elec_grouped_normalized['NOM_MUNICIPALITE'].unique():
  municipality_data = elec_grouped_normalized[elec_grouped_normalized['NOM_MUNICIPALITE'] == municipality]
  plt.plot(municipality_data['ANNEE_MOIS'], municipality_data['Consommation_Normalisee'], label=municipality)

plt.xlabel('Année-Mois')
plt.ylabel('Consommation Normalisée (0-1)')
plt.title('Consommation Électrique Normalisée par Municipalité')
plt.xticks(rotation=45, ha='right')
plt.legend(loc='upper left', bbox_to_anchor=(1, 1))
plt.tight_layout()
plt.show()
```

```
<ipython-input-7-eead0eeb3bec>:15: DeprecationWarning: DataFrameGroupBy.apply operated on the grouping columns. This behavior is
  elec_grouped_normalized = elec_grouped.groupby('NOM_MUNICIPALITE').apply(apply_scaler).reset_index(drop=True)
<ipython-input-7-eead0eeb3bec>:30: UserWarning: Tight layout not applied. The bottom and top margins cannot be made large enough
  plt.tight_layout()
```




```
elec_grouped.head()
```

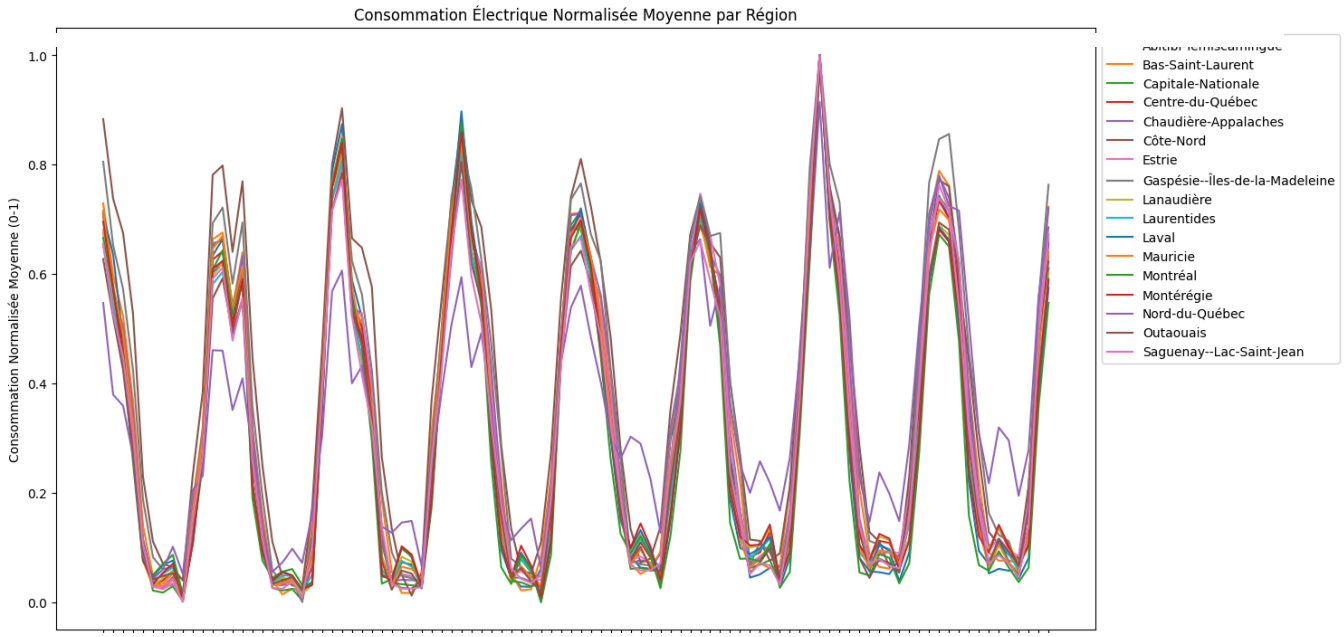
	NOM_MUNICIPALITE	ANNEE_MOIS	REGION_ADM_QC_TXT	Total (kWh)
0	Acton Vale	2016-01	Montérégie	8163358.00
1	Acton Vale	2016-02	Montérégie	7159392.00
2	Acton Vale	2016-03	Montérégie	6484247.00
3	Acton Vale	2016-04	Montérégie	5245742.00
4	Acton Vale	2016-05	Montérégie	3656639.00

```
elec_grouped_normalized.head()
```

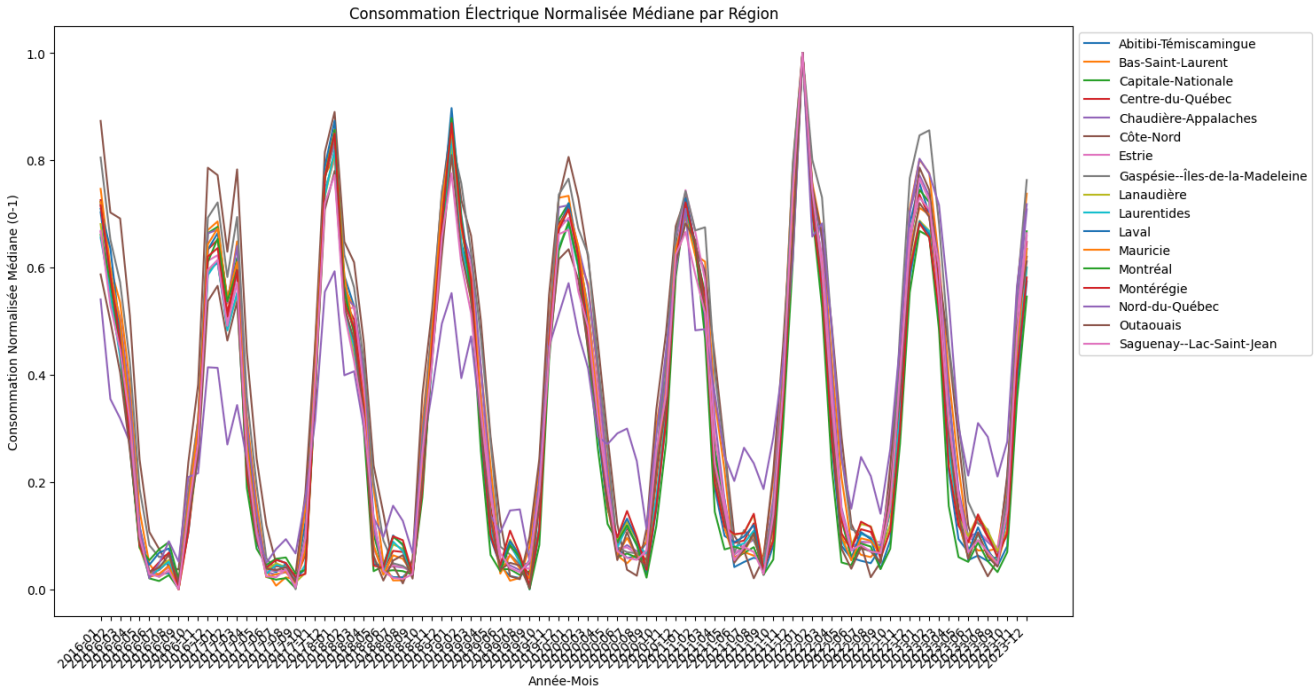
	NOM_MUNICIPALITE	ANNEE_MOIS	REGION_ADM_QC_TXT	Total (kWh)	Consommation_Normalisee
0	Acton Vale	2016-01	Montérégie	8163358.00	0.72
1	Acton Vale	2016-02	Montérégie	7159392.00	0.58
2	Acton Vale	2016-03	Montérégie	6484247.00	0.49
3	Acton Vale	2016-04	Montérégie	5245742.00	0.32
4	Acton Vale	2016-05	Montérégie	3656639.00	0.10

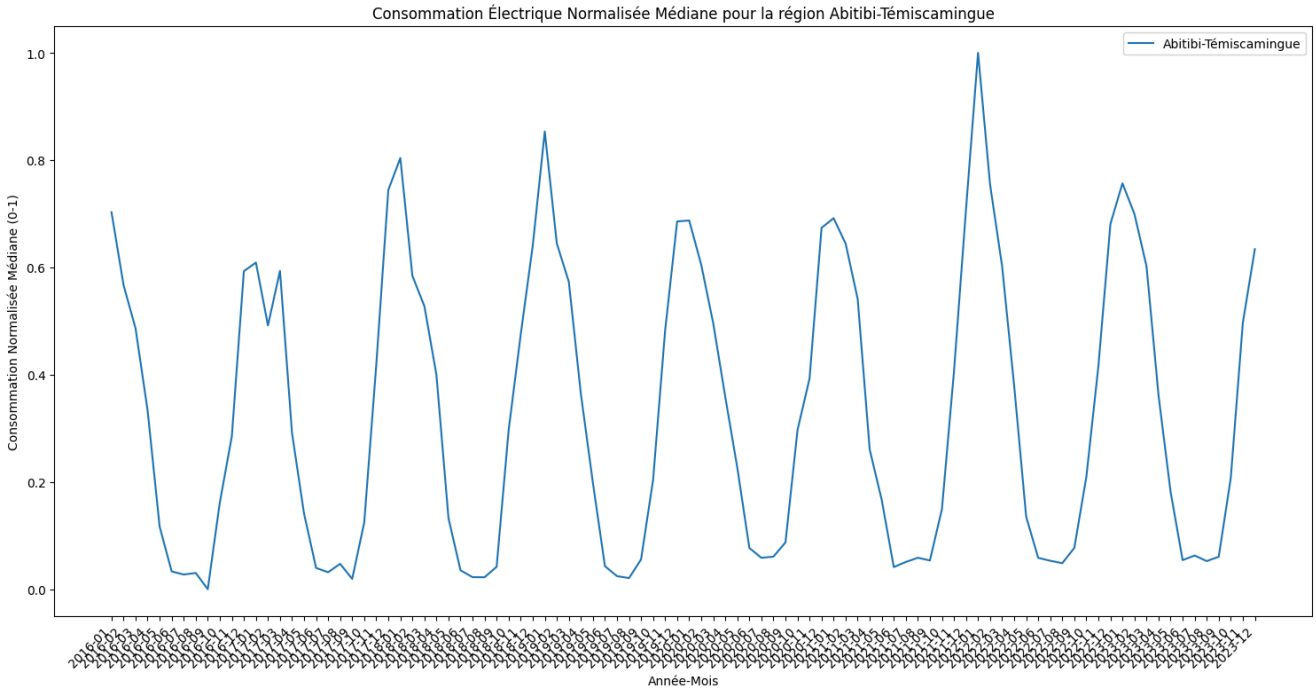
```
# plot de la consommation normalisée moyenne par région
```

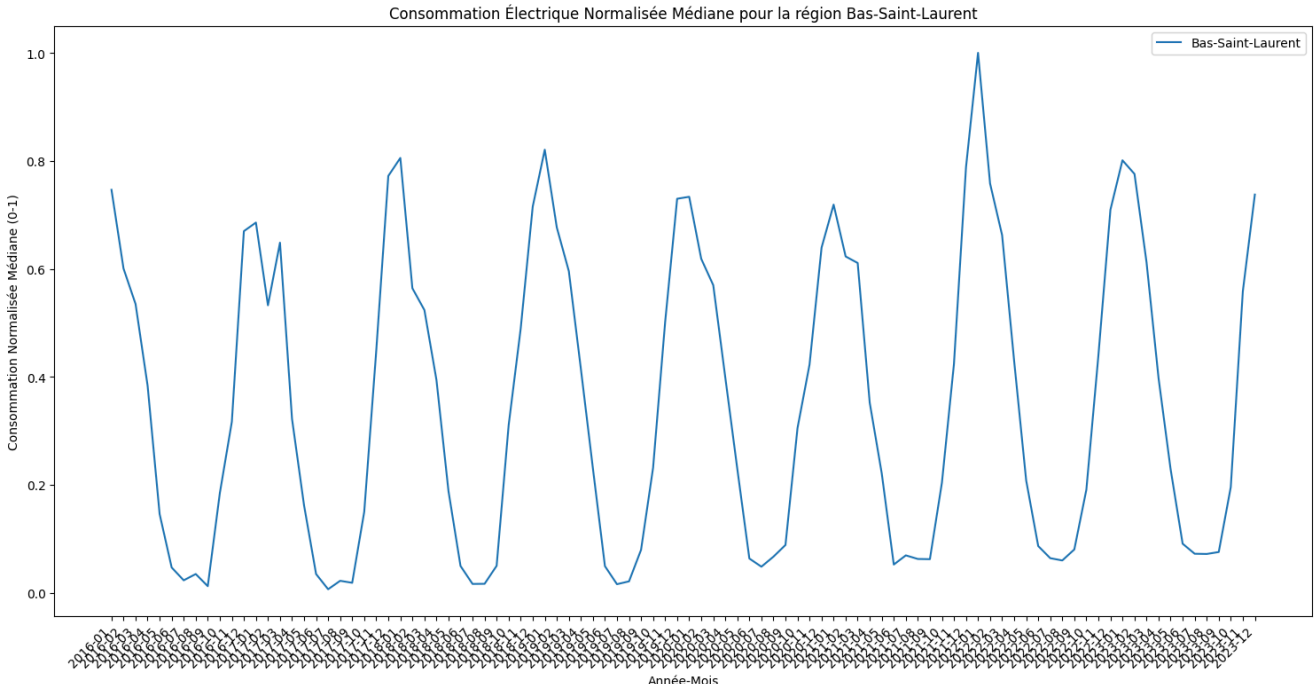
```
import matplotlib.pyplot as plt
region_avg_consumption = elec_grouped_normalized.groupby(['REGION_ADM_QC_TXT', 'ANNEE_MOIS'])['Consommation_Normalisee'].mean().reset_ir

# graphique
plt.figure(figsize=(15, 8))

for region in region_avg_consumption['REGION_ADM_QC_TXT'].unique():
    region_data = region_avg_consumption[region_avg_consumption['REGION_ADM_QC_TXT'] == region]
    plt.plot(region_data['ANNEE_MOIS'], region_data['Consommation_Normalisee'], label=region)

plt.xlabel('Année-Mois')
plt.ylabel('Consommation Normalisée Moyenne (0-1)')
plt.title('Consommation Électrique Normalisée Moyenne par Région')
plt.xticks(rotation=45, ha='right')
plt.legend(loc='upper left', bbox_to_anchor=(1, 1))
plt.tight_layout()
plt.show()
```

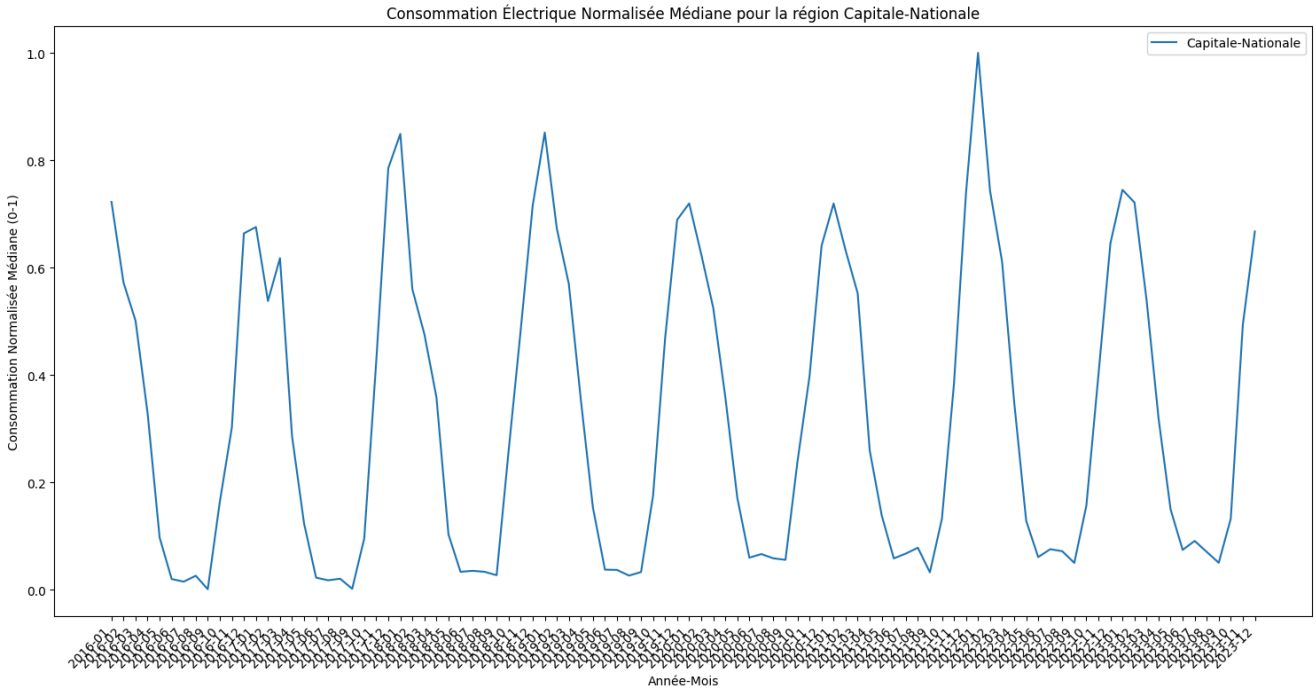


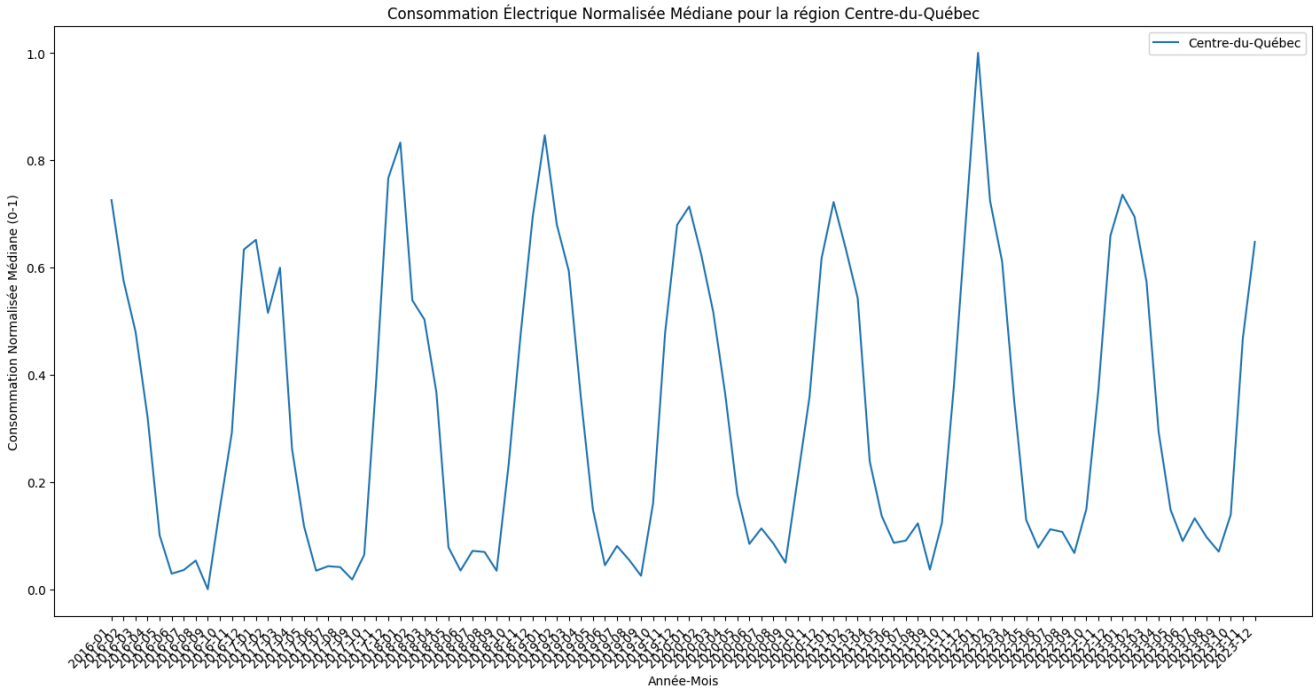

```
# plot de la consommation normalisée médiane par région

import matplotlib.pyplot as plt
region_median_consumption = elec_grouped_normalized.groupby(['REGION_ADM_QC_TXT', 'ANNEE_MOIS'])['Consommation_Normalisee'].median().res

# Create a plot with all regions together
plt.figure(figsize=(15, 8))  # Adjust figure size for better visibility

for region in region_median_consumption['REGION_ADM_QC_TXT'].unique():
    region_data = region_median_consumption[region_median_consumption['REGION_ADM_QC_TXT'] == region]
    plt.plot(region_data['ANNEE_MOIS'], region_data['Consommation_Normalisee'], label=region)

plt.xlabel('Année-Mois')
plt.ylabel('Consommation Normalisée Médiane (0-1)')
plt.title('Consommation Électrique Normalisée Médiane par Région')
plt.xticks(rotation=45, ha='right')
plt.legend(loc='upper left', bbox_to_anchor=(1, 1))
plt.tight_layout()
plt.show()

# Create separate plots for each region
for region in region_median_consumption['REGION_ADM_QC_TXT'].unique():
    region_data = region_median_consumption[region_median_consumption['REGION_ADM_QC_TXT'] == region]
    plt.figure(figsize=(15, 8))  # Adjust figure size for better visibility
    plt.plot(region_data['ANNEE_MOIS'], region_data['Consommation_Normalisee'], label=region)
    plt.xlabel('Année-Mois')
    plt.ylabel('Consommation Normalisée Médiane (0-1)')
    plt.title(f'Consommation Électrique Normalisée Médiane pour la région {region}')
    plt.xticks(rotation=45, ha='right')
    plt.legend()
    plt.tight_layout()
    plt.show()
```

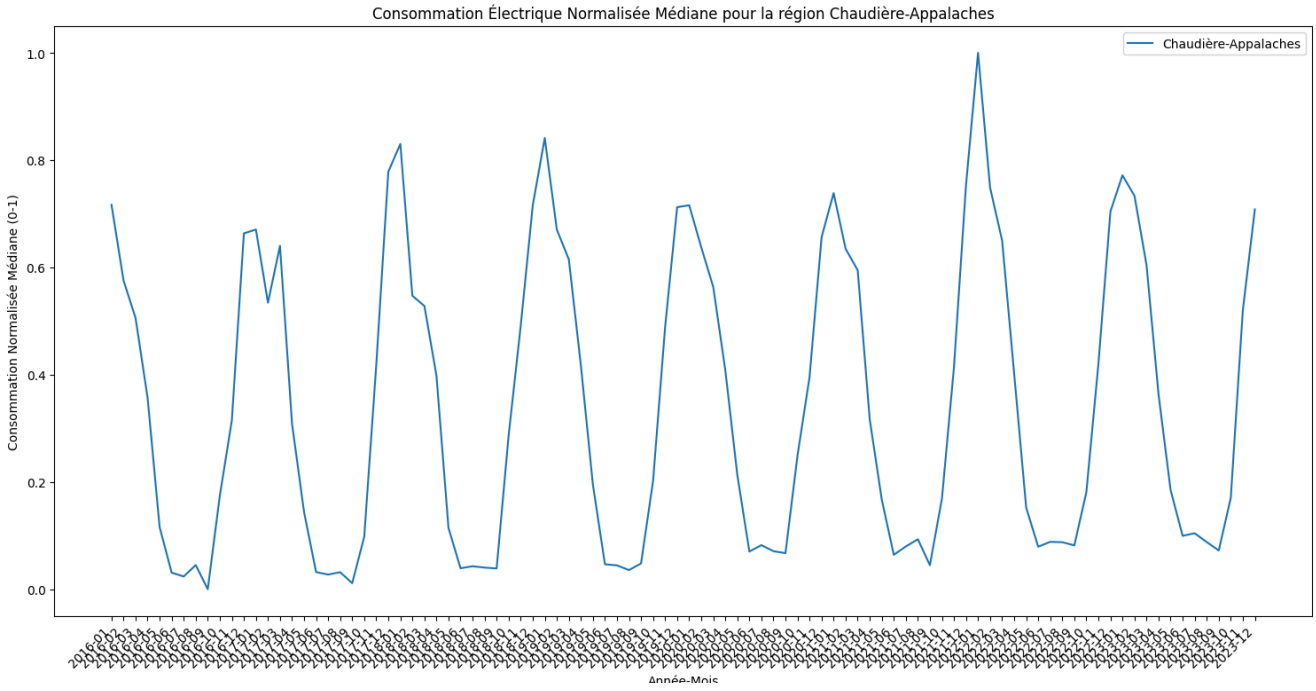

Consommation Électrique Normalisée Médiane par Région
Consommation Normalisée Médiane (0-1)
Année-Mois
Abitibi-Témiscamingue
Bas-Saint-Laurent
Capitale-Nationale
Centre-du-Québec
Chaudière-Appalaches
Côte-Nord
Estrie
Gaspésie--Îles-de-la-Madeleine
Lanaudière
Laurentides
Laval
Mauricie
Montréal
Montérégie
Nord-du-Québec
Outaouais
Saguenay--Lac-Saint-Jean

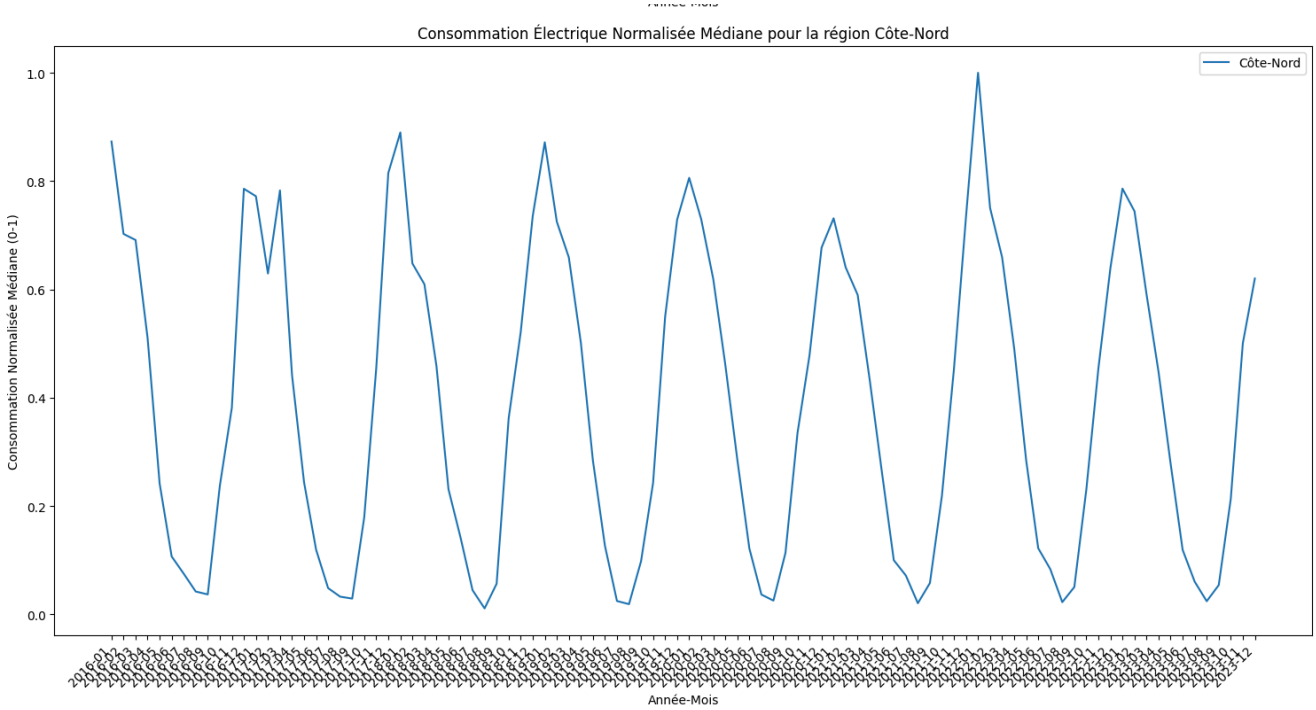

Consommation Électrique Normalisée Médiane pour la région Abitibi-Témiscamingue
Consommation Normalisée Médiane (0-1)
Année-Mois
Abitibi-Témiscamingue

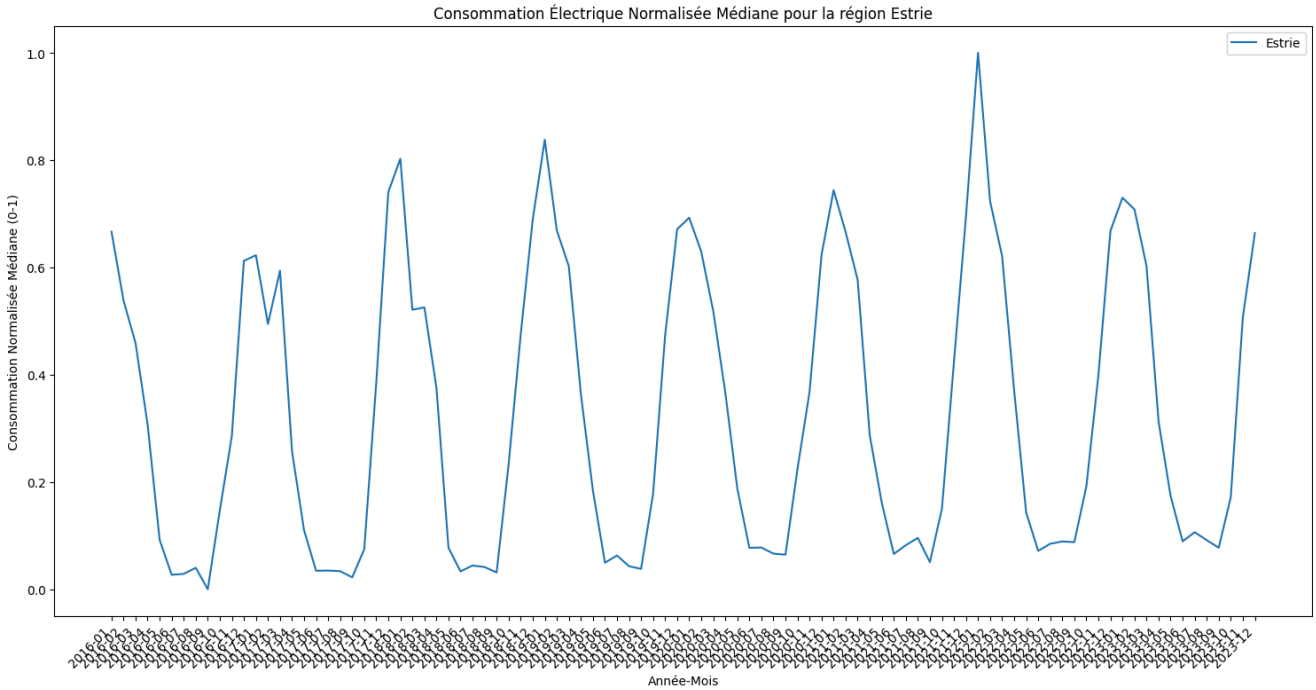

Consommation Électrique Normalisée Médiane pour la région Bas-Saint-Laurent
Consommation Normalisée Médiane (0-1)
Année-Mois
Bas-Saint-Laurent

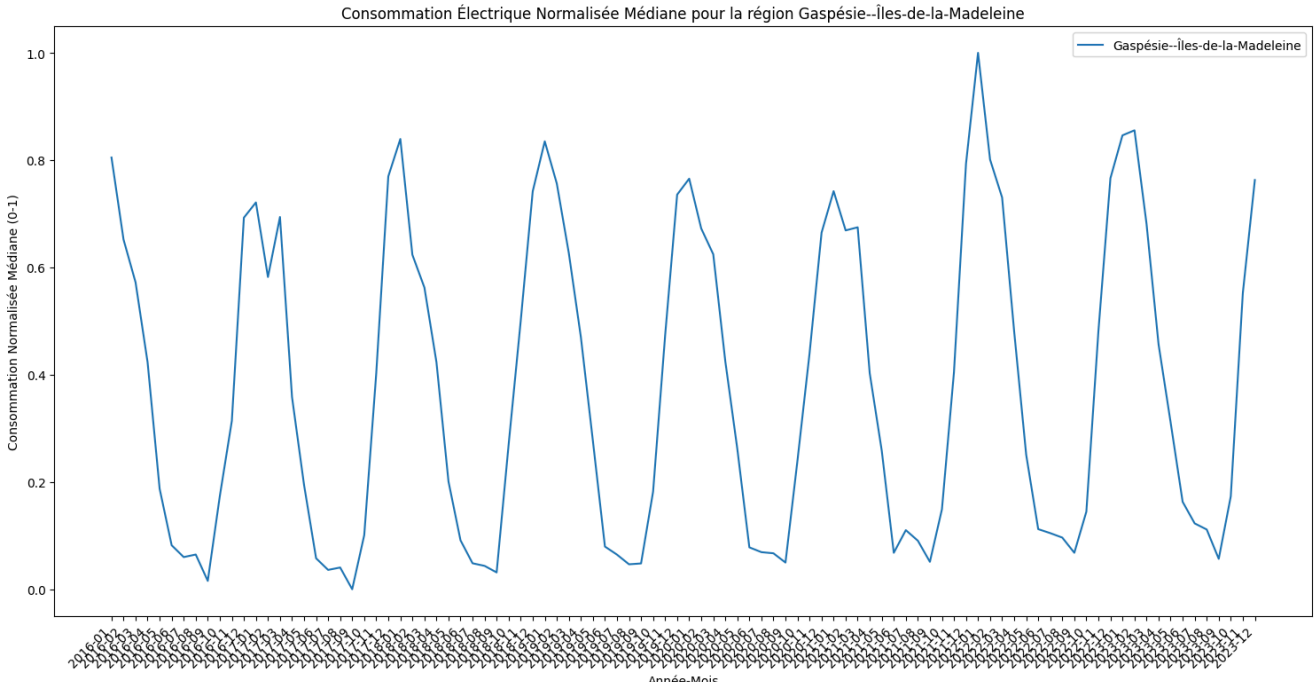

Consommation Électrique Normalisée Médiane pour la région Capitale-Nationale
Capitale-Nationale
Consommation Normalisée Médiane (0-1)
Année-Mois

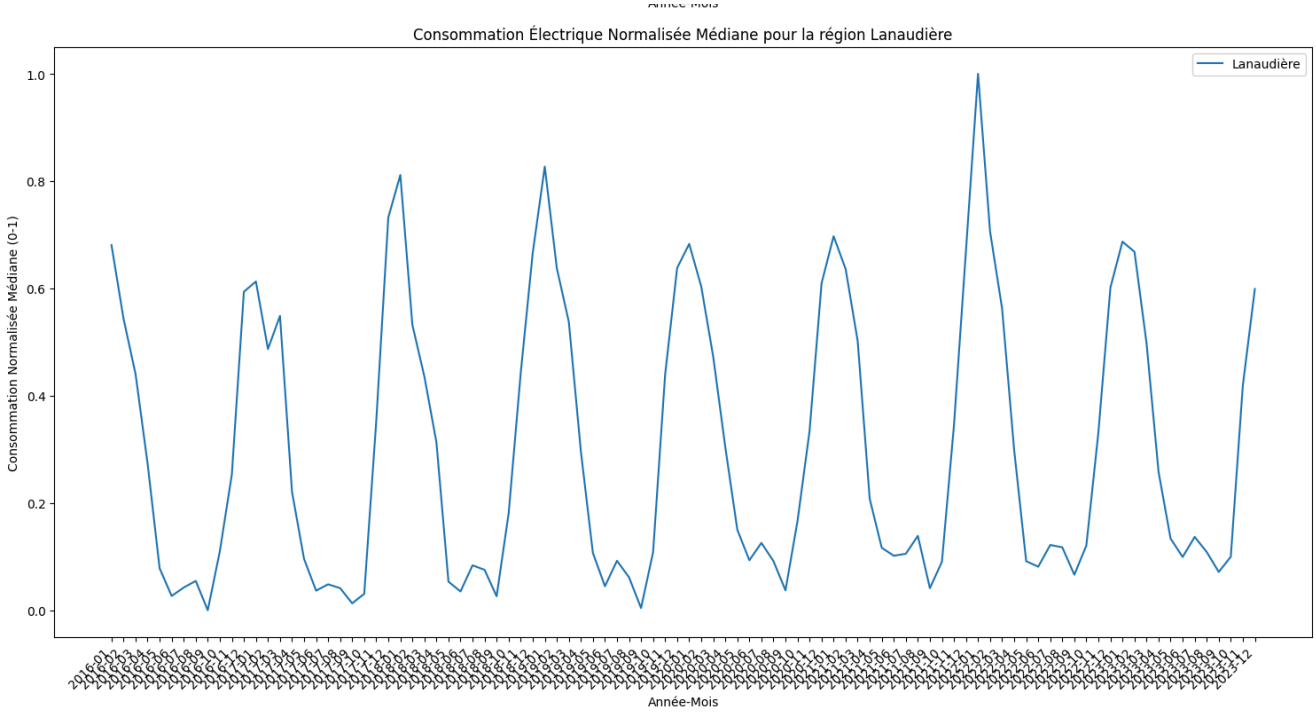

Consommation Électrique Normalisée Médiane pour la région Centre-du-Québec
Centre-du-Québec
Consommation Normalisée Médiane (0-1)
Année-Mois

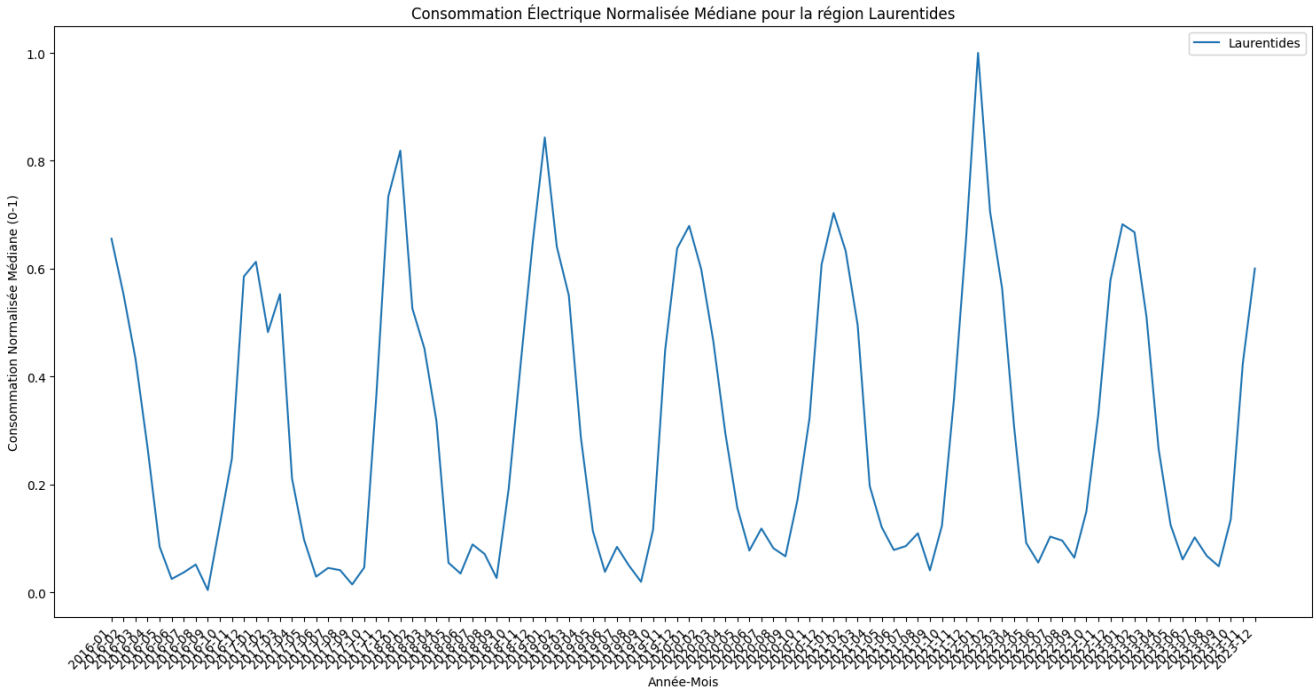

Consommation Électrique Normalisée Médiane pour la région Chaudière-Appalaches
Chaudière-Appalaches
Consommation Normalisée Médiane (0-1)
Année-Mois

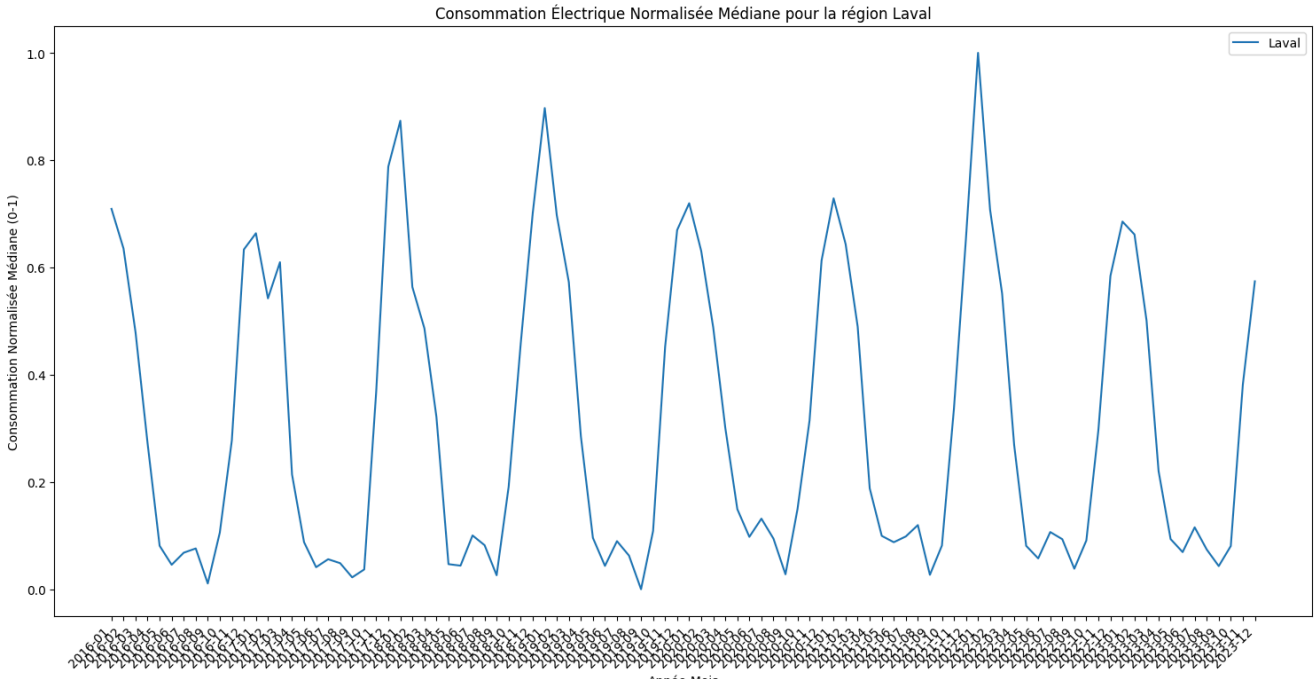

Consommation Électrique Normalisée Médiane pour la région Côte-Nord
Côte-Nord
Consommation Normalisée Médiane (0-1)
Année-Mois

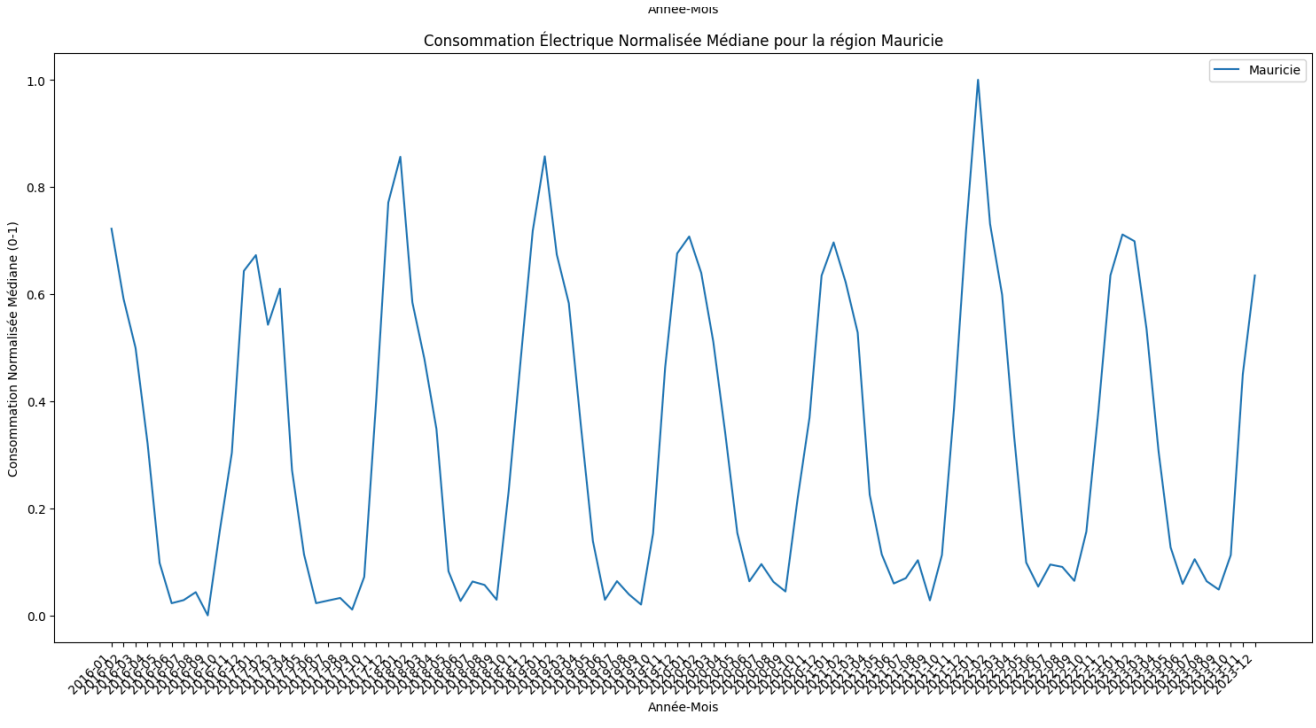

Consommation Électrique Normalisée Médiane pour la région Estrie
Estrie
Consommation Normalisée Médiane (0-1)
Année-Mois

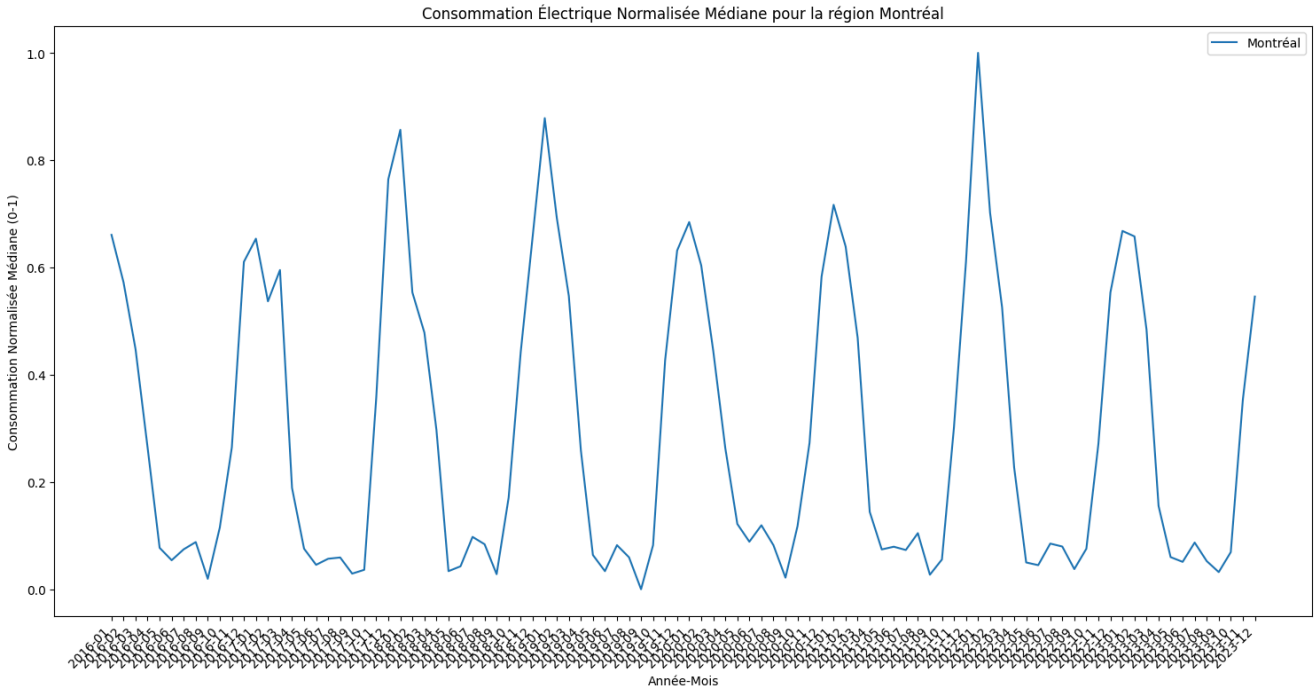

Consommation Électrique Normalisée Médiane pour la région Gaspésie--Îles-de-la-Madeleine
Gaspésie--Îles-de-la-Madeleine
Consommation Normalisée Médiane (0-1)
Année-Mois

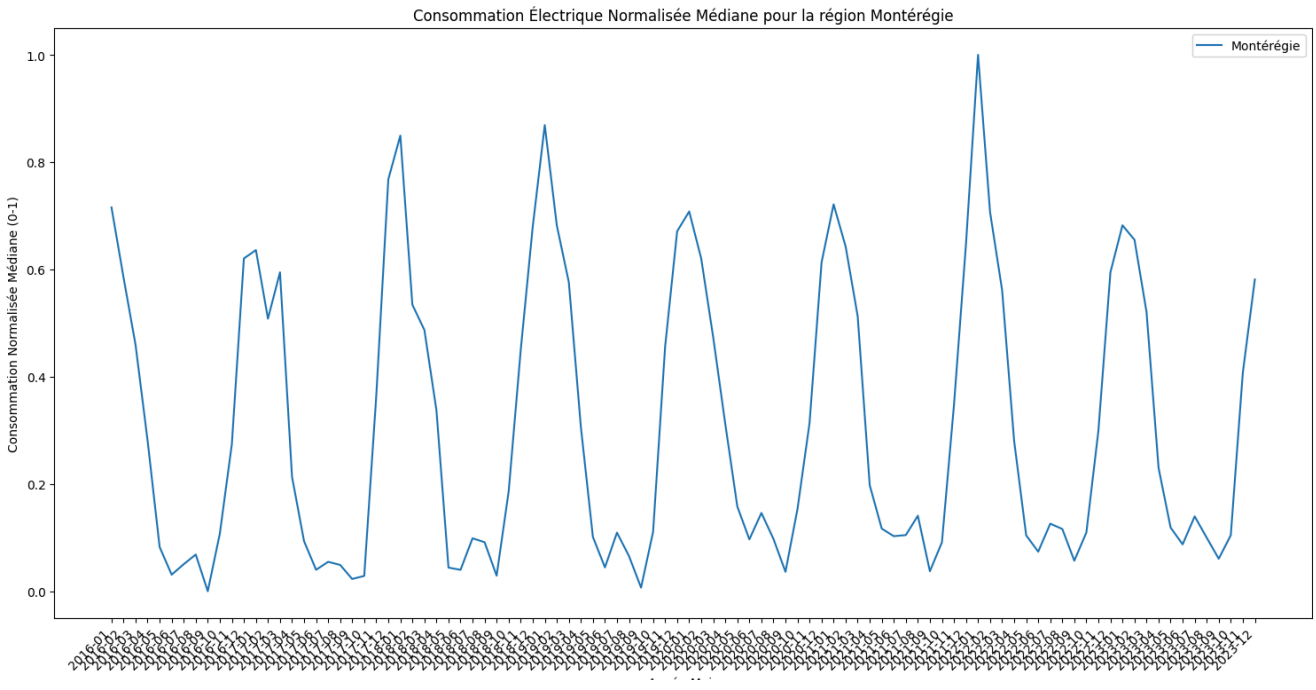

Consommation Électrique Normalisée Médiane pour la région Lanaudière
Lanaudière
Consommation Normalisée Médiane (0-1)
Année-Mois


Consommation Électrique Normalisée Médiane pour la région Laurentides
Laurentides
Consommation Normalisée Médiane (0-1)
Année-Mois

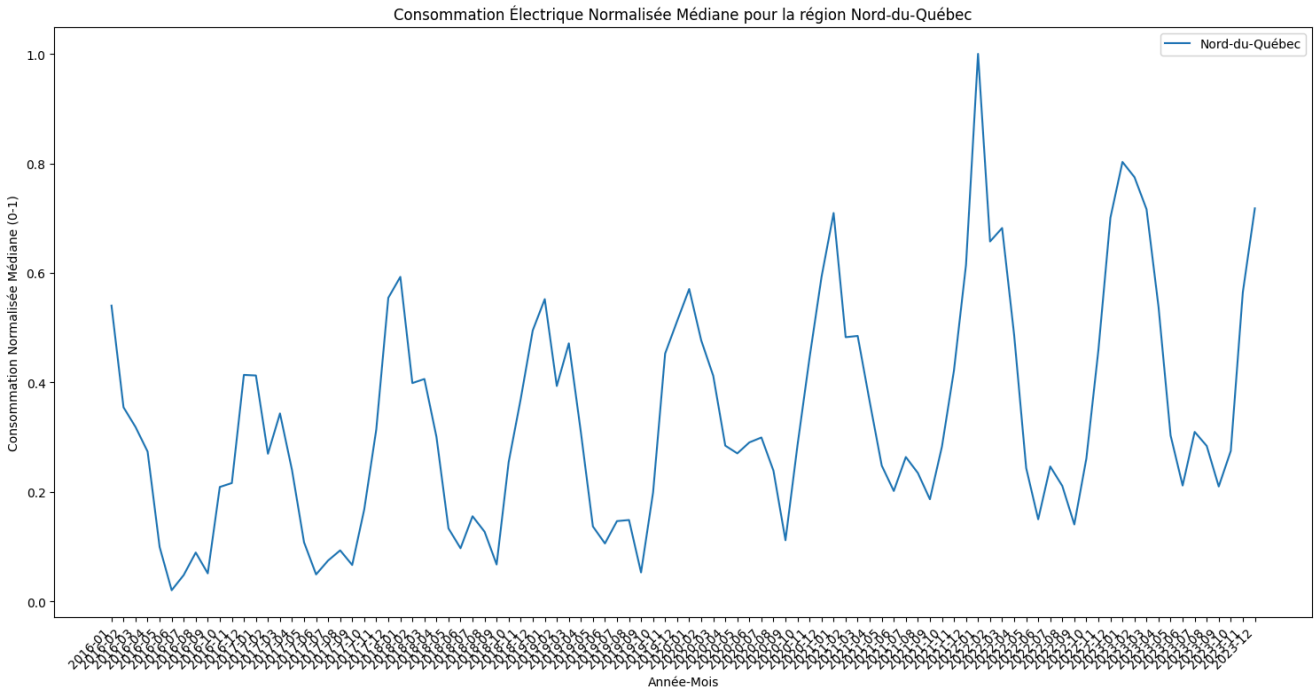

Consommation Électrique Normalisée Médiane pour la région Laval
Laval
Consommation Normalisée Médiane (0-1)

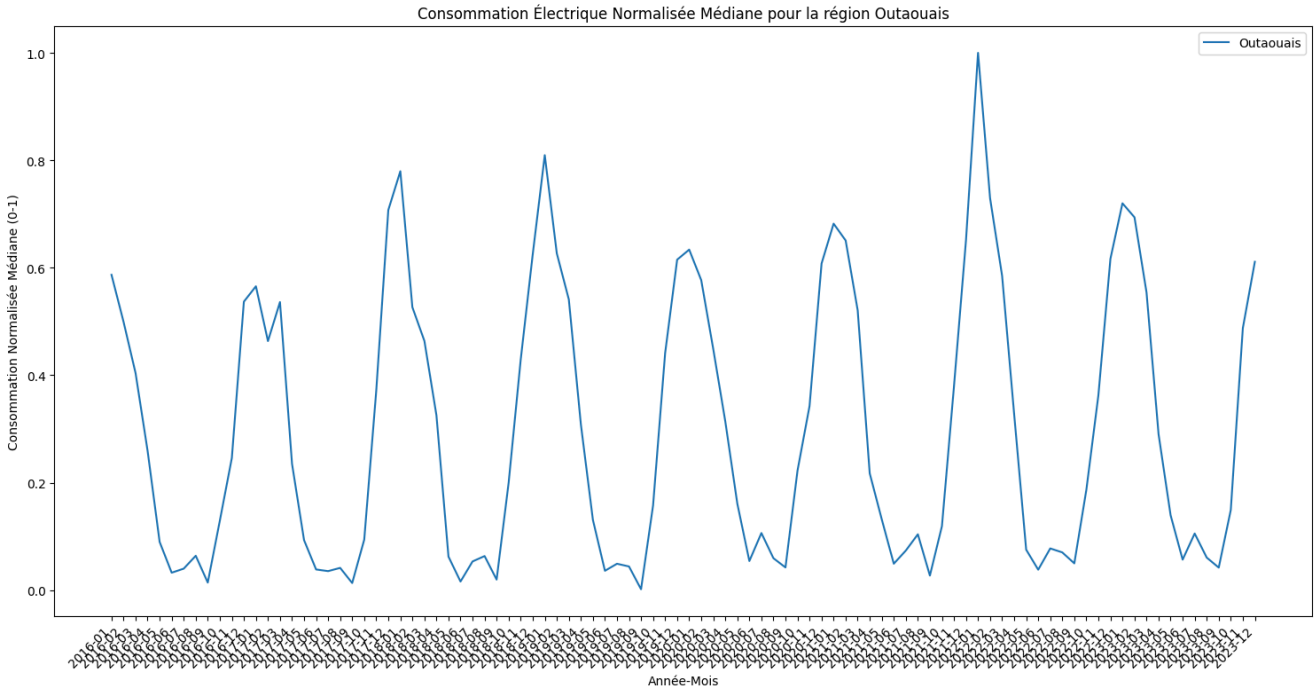

Année-Mois
Consommation Électrique Normalisée Médiane pour la région Mauricie
Mauricie
Consommation Normalisée Médiane (0-1)
Année-Mois

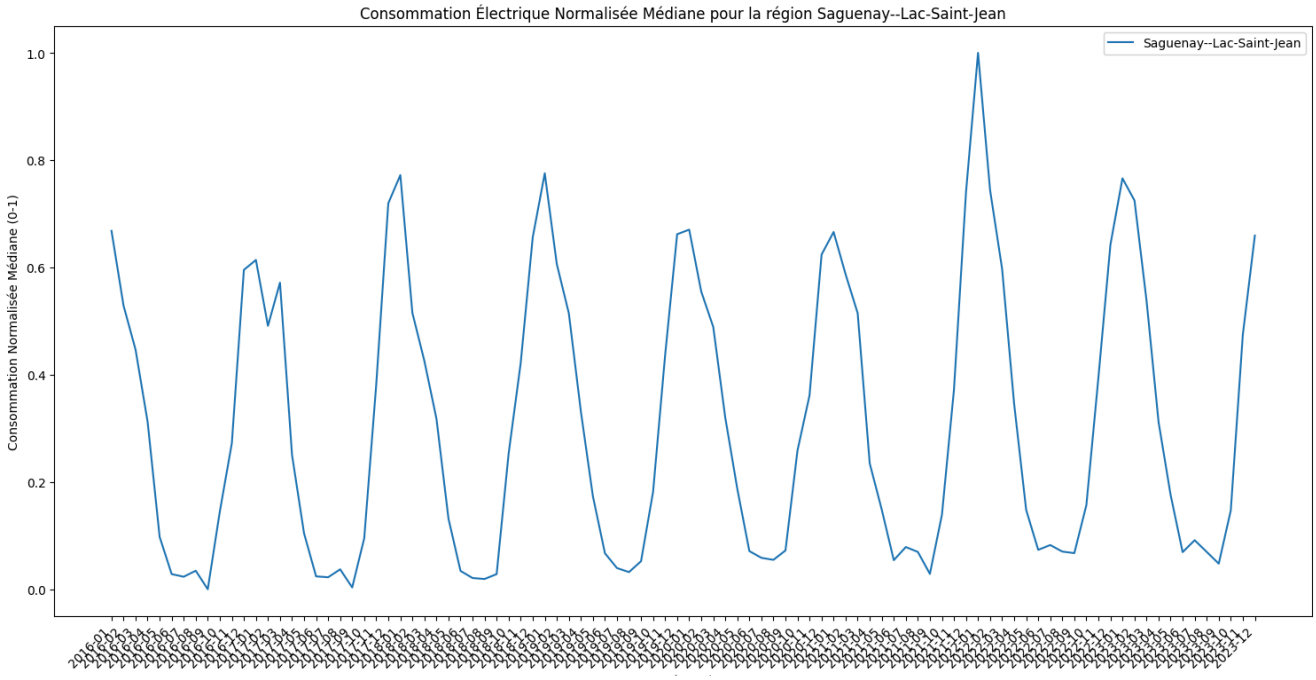

Consommation Électrique Normalisée Médiane pour la région Montréal
Montréal
Consommation Normalisée Médiane (0-1)
Année-Mois


Consommation Électrique Normalisée Médiane pour la région Montérégie
Montérégie
Consommation Normalisée Médiane (0-1)

Année-Mois

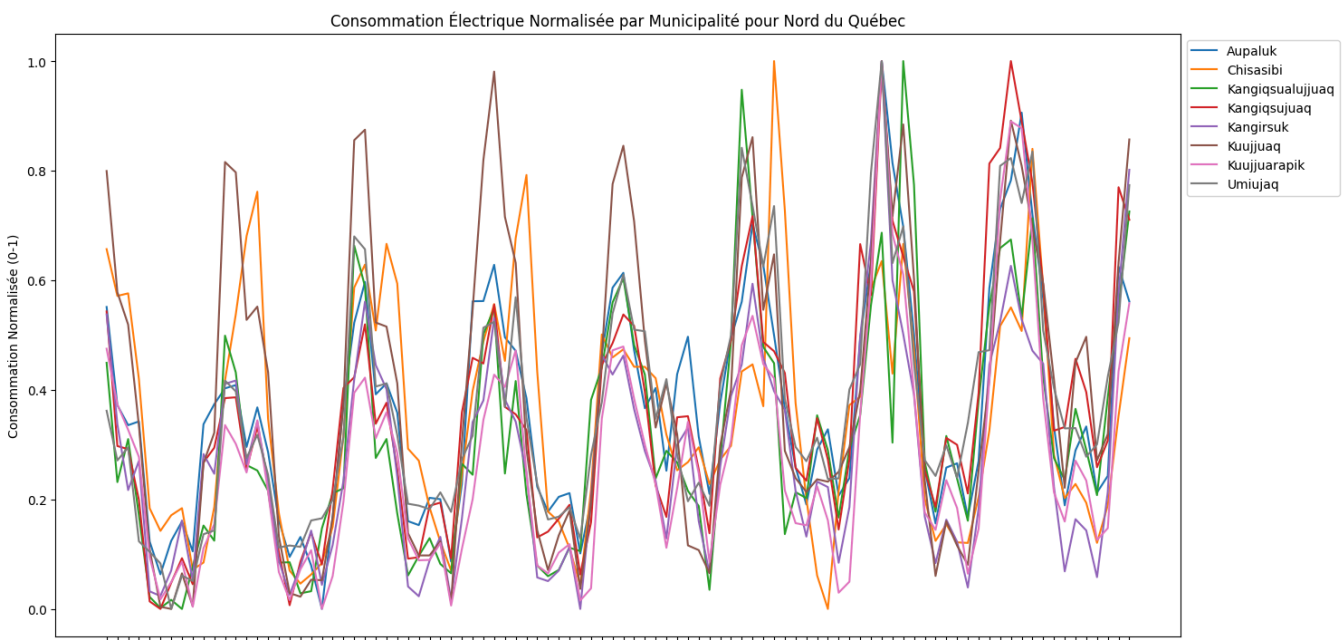

Consommation Électrique Normalisée Médiane pour la région Nord-du-Québec
Nord-du-Québec
Consommation Normalisée Médiane (0-1)
Année-Mois


Consommation Électrique Normalisée Médiane pour la région Outaouais
Outaouais
Consommation Normalisée Médiane (0-1)
Année-Mois


Consommation Électrique Normalisée Médiane pour la région Saguenay--Lac-Saint-Jean
Saguenay--Lac-Saint-Jean
Consommation Normalisée Médiane (0-1)

Annee-Mois

```
# graphique de la consommation des municipalités pour Nord du Québec

import matplotlib.pyplot as plt
# Filter data for Nord du Québec region
nord_quebec_data = elec_grouped_normalized[elec_grouped_normalized['REGION_ADM_QC_TXT'] == 'Nord-du-Québec']

# Create a plot for Nord du Québec
plt.figure(figsize=(15, 8))

for municipality in nord_quebec_data['NOM_MUNICIPALITE'].unique():
  municipality_data = nord_quebec_data[nord_quebec_data['NOM_MUNICIPALITE'] == municipality]
  plt.plot(municipality_data['ANNEE_MOIS'], municipality_data['Consommation_Normalisee'], label=municipality)

plt.xlabel('Année-Mois')
plt.ylabel('Consommation Normalisée (0-1)')
plt.title('Consommation Électrique Normalisée par Municipalité pour Nord du Québec')
plt.xticks(rotation=45, ha='right')
plt.legend(loc='upper left', bbox_to_anchor=(1, 1))
plt.tight_layout()
plt.show()
```

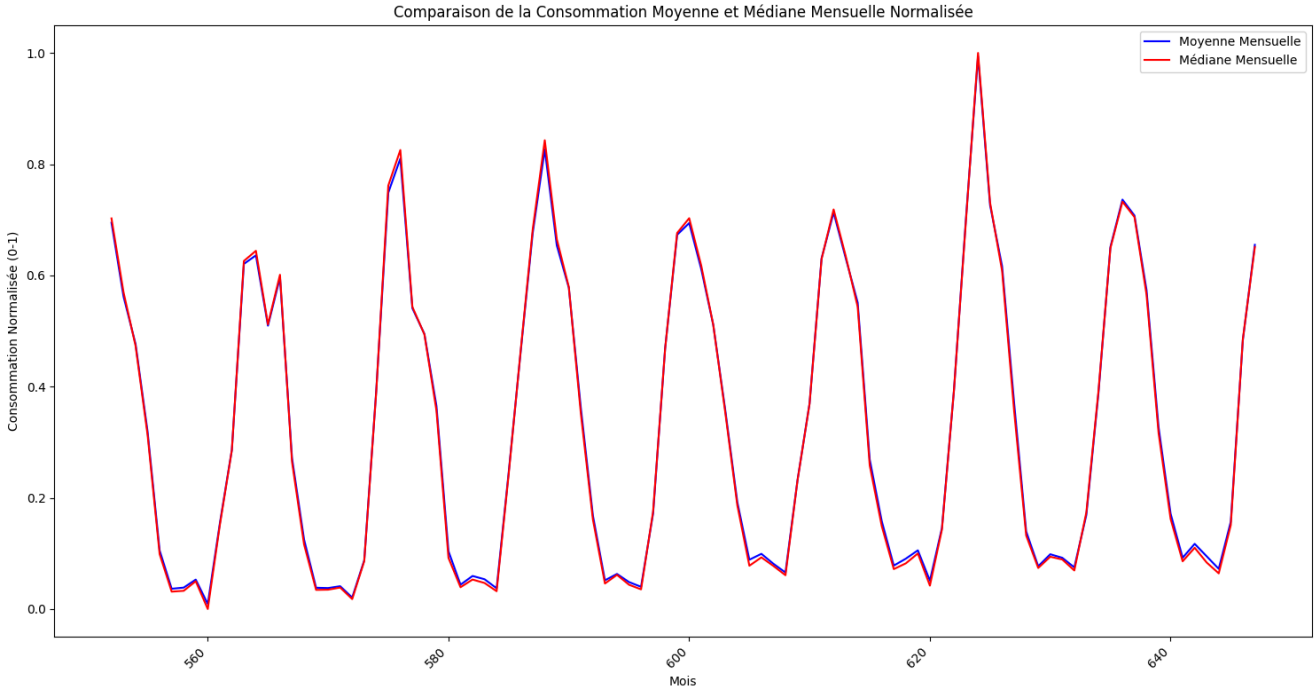


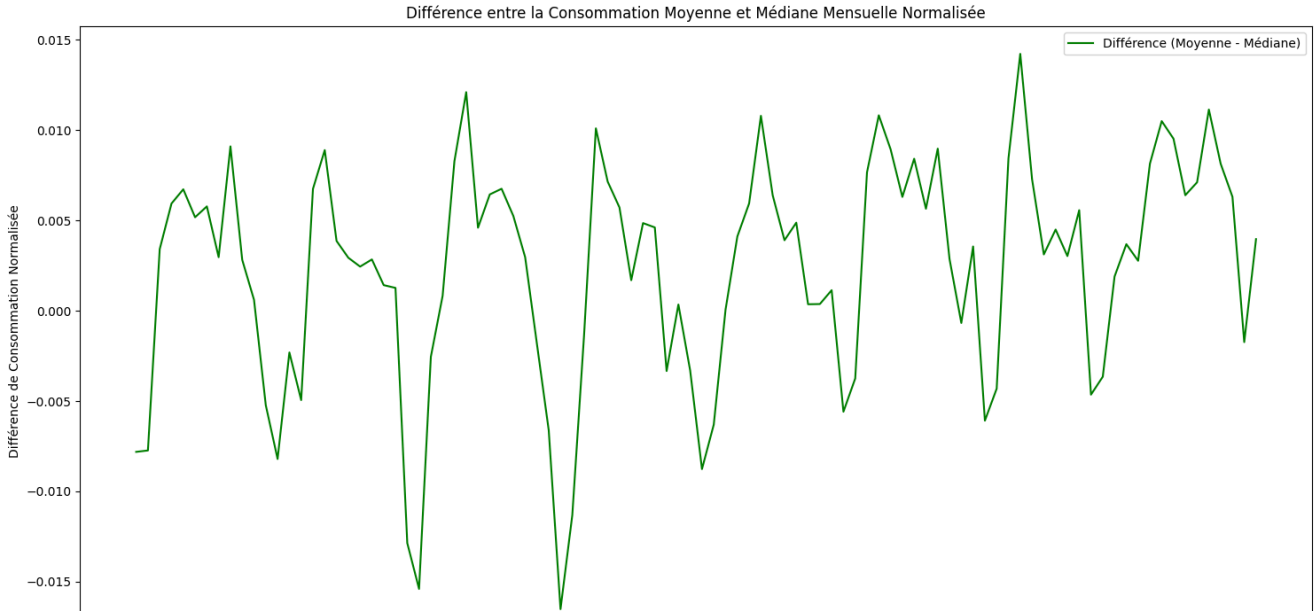

```
# nombre de villes par région

municipalities_by_region = elec_res_no_null.groupby('REGION_ADM_QC_TXT')['NOM_MUNICIPALITE'].nunique()

print(municipalities_by_region)
```

```
REGION_ADM_QC_TXT
Abitibi-Témiscamingue             12
Bas-Saint-Laurent                 25
Capitale-Nationale                 8
Centre-du-Québec                  16
Chaudière-Appalaches              36
Côte-Nord                          6
Estrie                            21
Gaspésie--Îles-de-la-Madeleine     2
Lanaudière                        10
Laurentides                       17
Laval                              1
Mauricie                          10
Montréal                           3
Montérégie                        31
Nord-du-Québec                     8
Outaouais                         11
Saguenay--Lac-Saint-Jean           9
Name: NOM_MUNICIPALITE, dtype: int64
```

```
# Retirer la région Nord-du-Québec de elec_grouped_normalized

elec_grouped_normalized_without_nord = elec_grouped_normalized[
    elec_grouped_normalized['REGION_ADM_QC_TXT'] != 'Nord-du-Québec'
]
```

```
# comparaison de mes courbes médianes et mensuelles
elec_grouped_normalized['ANNEE_MOIS_NUM'] = pd.to_datetime(elec_grouped_normalized['ANNEE_MOIS'], format='%Y-%m').dt.to_period('M').asty


monthly_mean_consumption = elec_grouped_normalized.groupby('ANNEE_MOIS_NUM')['Consommation_Normalisee'].mean().reset_index()
monthly_median_consumption = elec_grouped_normalized.groupby('ANNEE_MOIS_NUM')['Consommation_Normalisee'].median().reset_index()


combined_consumption = pd.merge(monthly_mean_consumption, monthly_median_consumption, on='ANNEE_MOIS_NUM', suffixes=('_mean', '_median'))


plt.figure(figsize=(15, 8))
plt.plot(combined_consumption['ANNEE_MOIS_NUM'], combined_consumption['Consommation_Normalisee_mean'], label='Moyenne Mensuelle', color=
plt.plot(combined_consumption['ANNEE_MOIS_NUM'], combined_consumption['Consommation_Normalisee_median'], label='Médiane Mensuelle', colo

plt.xlabel('Mois')
plt.ylabel('Consommation Normalisée (0-1)')
plt.title('Comparaison de la Consommation Moyenne et Médiane Mensuelle Normalisée')
plt.xticks(rotation=45, ha='right')
plt.legend()
plt.tight_layout()
plt.show()


combined_consumption['Difference'] = combined_consumption['Consommation_Normalisee_mean'] - combined_consumption['Consommation_Normalise


plt.figure(figsize=(15, 8))
plt.plot(combined_consumption['ANNEE_MOIS_NUM'], combined_consumption['Difference'], label='Différence (Moyenne - Médiane)', color='gree
plt.xlabel('Mois')
plt.ylabel('Différence de Consommation Normalisée')
plt.title('Différence entre la Consommation Moyenne et Médiane Mensuelle Normalisée')
plt.xticks(rotation=45, ha='right')
plt.legend()
plt.tight_layout()
plt.show()


print(f"Moyenne de la différence : {combined_consumption['Difference'].mean():.2f}")
print(f"Médiane de la différence : {combined_consumption['Difference'].median():.2f}")
print(f"Écart type de la différence : {combined_consumption['Difference'].std():.2f}")
```






```
monthly_mean_consumption.head()
```

	ANNEE_MOIS_NUM	Consommation_Normalisee
0	552	0.69
1	553	0.56
2	554	0.48
3	555	0.32
4	556	0.11

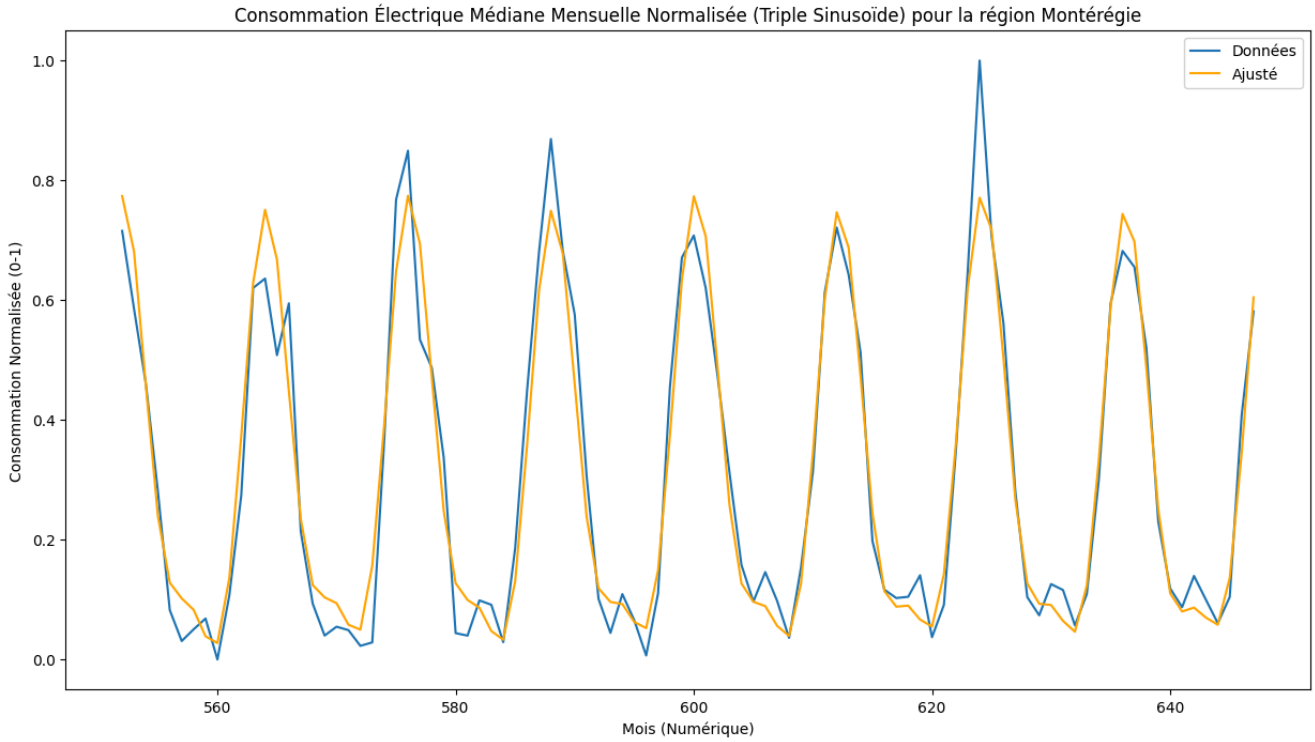
```
elec_grouped_normalized.head()
```

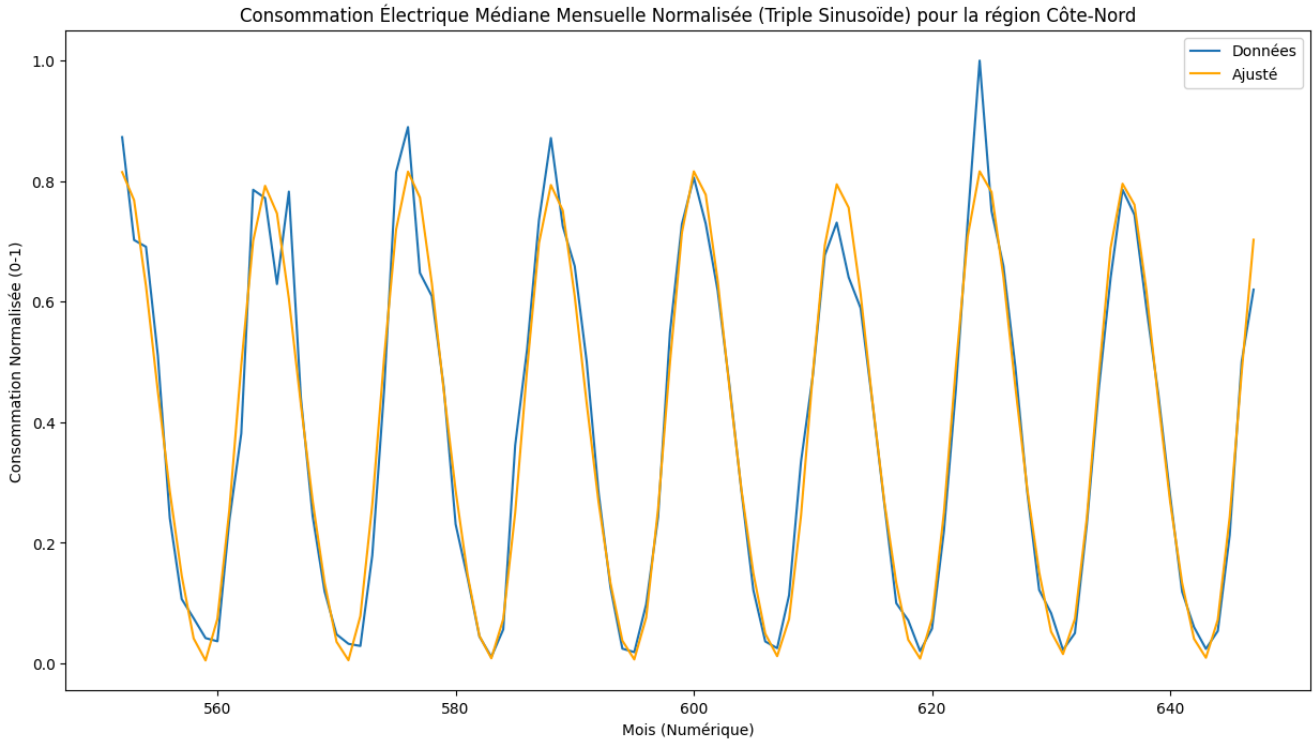
	NOM_MUNICIPALITE	ANNEE_MOIS	REGION_ADM_QC_TXT	Total (kWh)	Consommation_Normalisee	ANNEE_MOIS_NUM
0	Acton Vale	2016-01	Montérégie	8163358.00	0.72	552
1	Acton Vale	2016-02	Montérégie	7159392.00	0.58	553
2	Acton Vale	2016-03	Montérégie	6484247.00	0.49	554
3	Acton Vale	2016-04	Montérégie	5245742.00	0.32	555
4	Acton Vale	2016-05	Montérégie	3656639.00	0.10	556

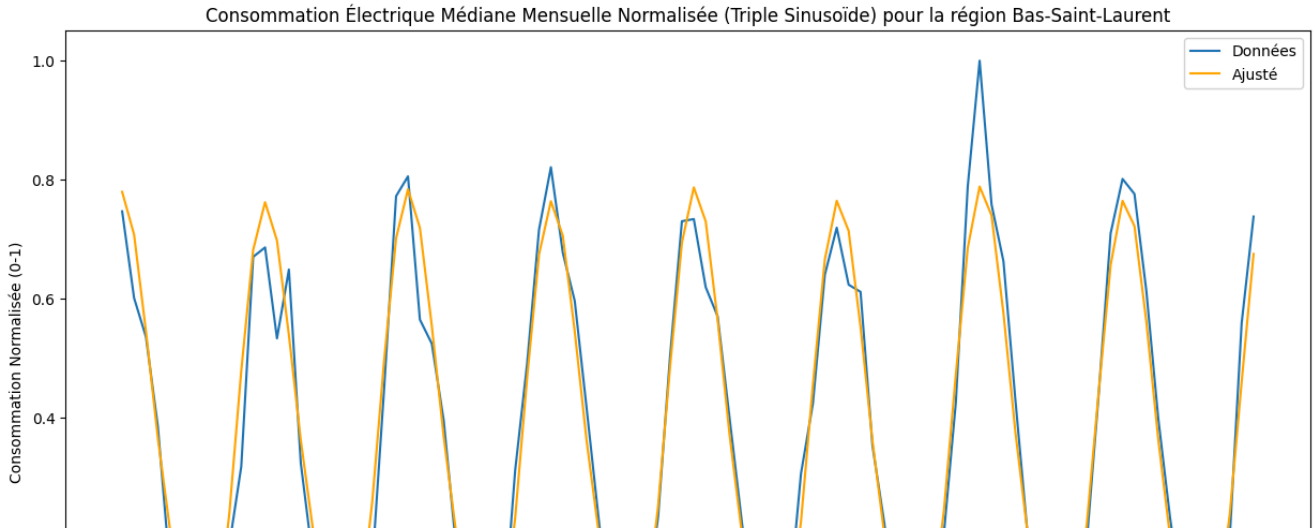
```
#generer la courbe ajutée avec trois sinus pour chaque région en prenant en compte le pic mois
x_data = monthly_mean_consumption['ANNEE_MOIS_NUM']
```

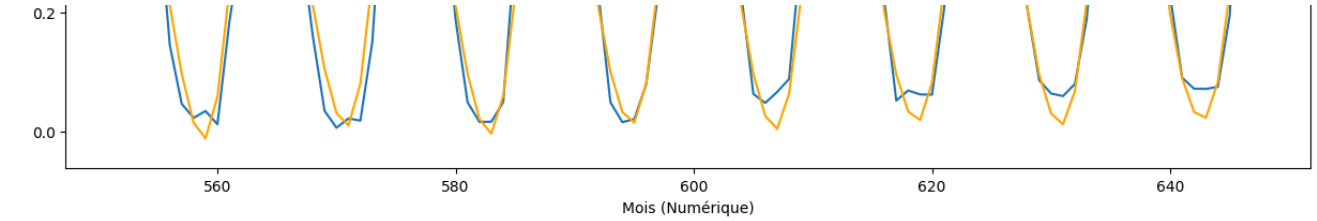
```
y_data = monthly_mean_consumption['Consommation_Normalisee']

from scipy.optimize import curve_fit

#Fonction pour ajuster une courbe sinusoidale
def fit_func_triple(x, amplitude1, frequency1, phase1, amplitude2, frequency2, phase2, amplitude3, frequency3, phase3, offset):
    return (amplitude1 * np.sin(frequency1 * x + phase1) +
            amplitude2 * np.sin(frequency2 * x + phase2) +
            amplitude3 * np.sin(frequency3 * x + phase3) + offset)

# df vide
results_df = pd.DataFrame(columns=['Region', 'Amplitude1', 'Frequency1', 'Phase1', 'Amplitude2', 'Frequency2', 'Phase2', 'Amplitude3',

# boucle pour chaque région
for region in elec_grouped_normalized['REGION_ADM_QC_TXT'].unique():
    region_data = elec_grouped_normalized[elec_grouped_normalized['REGION_ADM_QC_TXT'] == region]
    monthly_median_consumption_region = region_data.groupby(region_data['ANNEE_MOIS_NUM'])['Consommation_Normalisee'].median().reset_inc
    x_data = monthly_median_consumption_region['ANNEE_MOIS_NUM']
    y_data = monthly_median_consumption_region['Consommation_Normalisee']

    # Tentatives initiales pour la courbe
    amplitude1_initial = (y_data.max() - y_data.min()) / 2
    frequency1_initial = 2 * np.pi / 6
    amplitude2_initial = (y_data.max() - y_data.min()) / 4
    frequency2_initial = 2 * np.pi / 12
    amplitude3_initial = (y_data.max() - y_data.min()) / 6
    frequency3_initial = 2 * np.pi / 24  # Adding a third frequency for more flexibility
    phase1_initial = 0
    phase2_initial = 0
    phase3_initial = 0
    offset_initial = y_data.mean()

    initial_guess_triple = [amplitude1_initial, frequency1_initial, phase1_initial,
                            amplitude2_initial, frequency2_initial, phase2_initial,
                            amplitude3_initial, frequency3_initial, phase3_initial,
                            offset_initial]

    try:
        popt, _ = curve_fit(fit_func_triple, x_data, y_data, p0=initial_guess_triple, maxfev=1000)
        amplitude1, frequency1, phase1, amplitude2, frequency2, phase2, amplitude3, frequency3, phase3, offset = popt
        fitted_y = fit_func_triple(x_data, amplitude1, frequency1, phase1, amplitude2, frequency2, phase2, amplitude3, frequency3, phase

        # trouver le mois de pic
        month_peak = monthly_median_consumption_region.loc[monthly_median_consumption_region['Consommation_Normalisee'].idxmax()]['ANNEI

        # Calculer le pic normalisé
        max_consumption = y_data.max()
        month_peak_index = x_data[x_data == month_peak].index[0]
        manual_peak_addition = max_consumption - fitted_y[month_peak_index]
        # Mettre résultats dans df vide
        equation_string = f"y = {amplitude1:.2f} * sin({frequency1:.2f}x + {phase1:.2f}) + {amplitude2:.2f} * sin({frequency2:.2f}x + {p
        results_df = pd.concat([results_df, pd.DataFrame({'Region': [region],
                                                          'Amplitude1': [amplitude1], 'Frequency1': [frequency1], 'Phase1': [phase1],
                                                          'Amplitude2': [amplitude2], 'Frequency2': [frequency2], 'Phase2': [phase2],
                                                          'Amplitude3': [amplitude3], 'Frequency3': [frequency3], 'Phase3': [phase3],
                                                          'Offset': [offset], 'Equation': [equation_string],
                                                          'Month_Peak': [month_peak], 'Manual_Peak_Addition': [manual_peak_addition]})], i

        # graphique
        plt.figure(figsize=(15, 8))
        plt.plot(x_data, y_data, label='Données')
        plt.plot(x_data, fitted_y, label='Ajusté', color='orange')
        plt.xlabel('Mois (Numérique)')
        plt.ylabel('Consommation Normalisée (0-1)')
        plt.title(f'Consommation Électrique Médiane Mensuelle Normalisée (Triple Sinusoïde) pour la région {region}')
        plt.legend()
        plt.show()


    except RuntimeError:
        print(f"Impossible d'ajuster une courbe pour la région {region}")

# afficher les résultats
print(results_df)
```

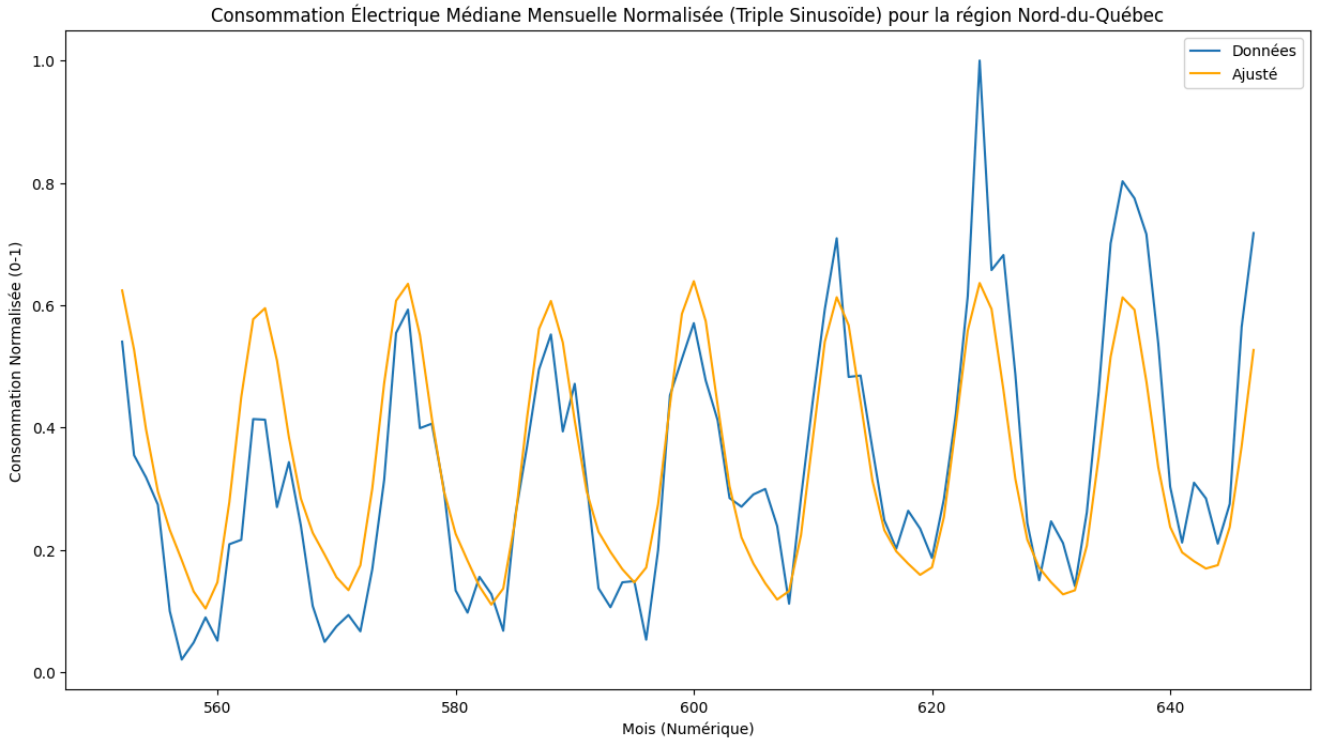
```
<ipython-input-18-bbfb35e3b1a7>:54: FutureWarning: The behavior of DataFrame concatenation with empty or all-NA entries is deprec
  results_df = pd.concat([results_df, pd.DataFrame({'Region': [region],
```

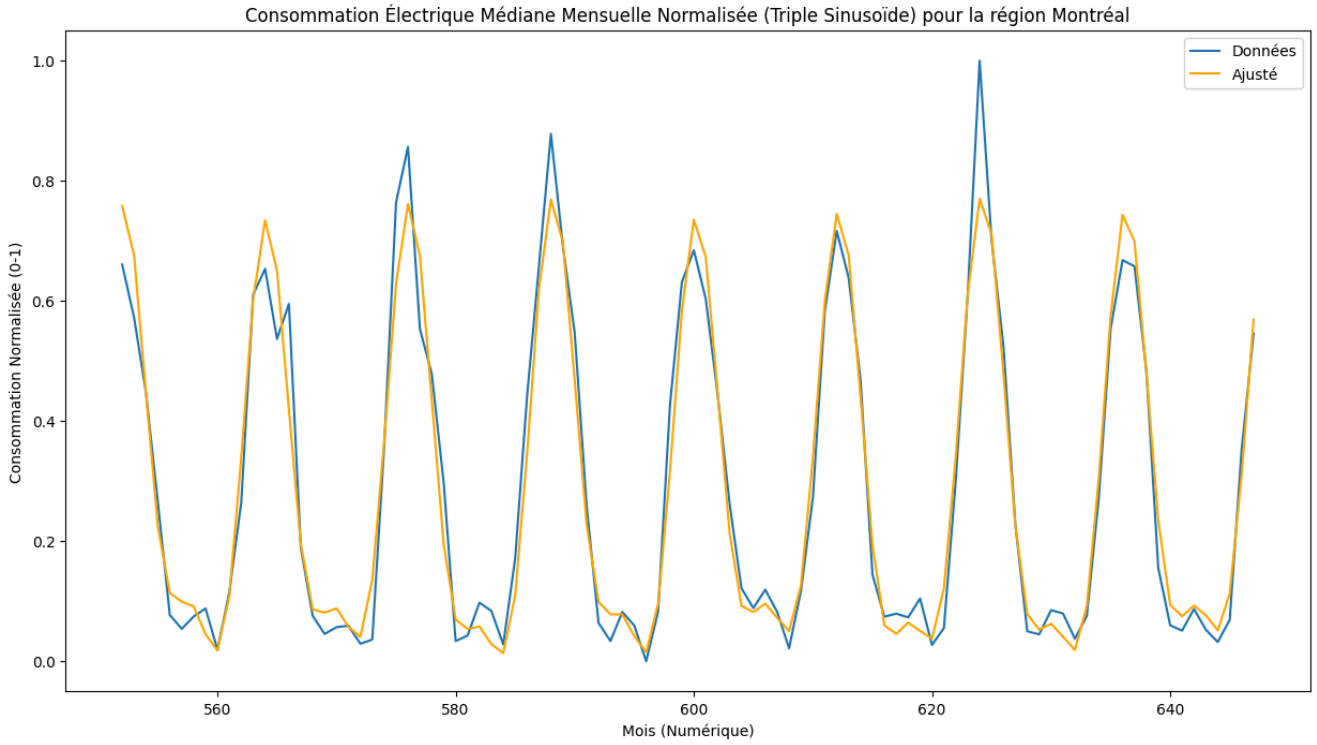


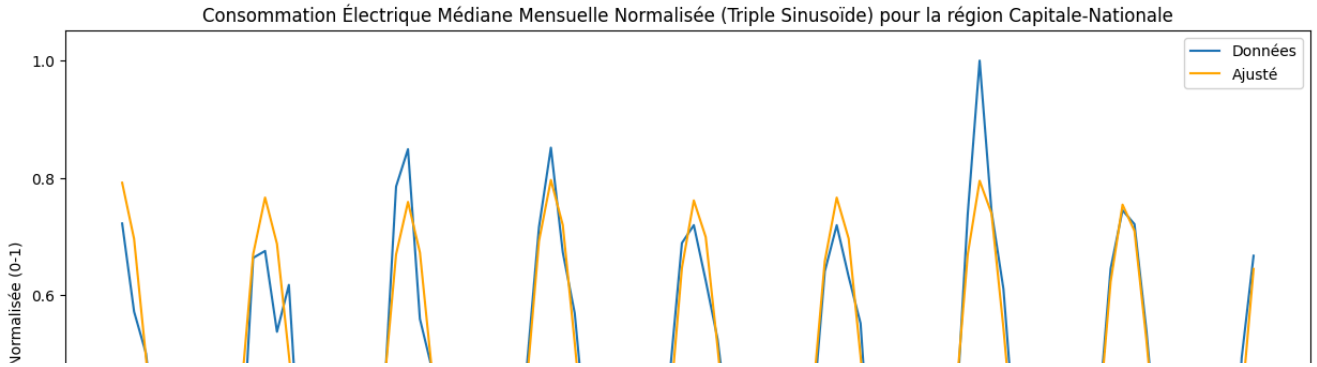


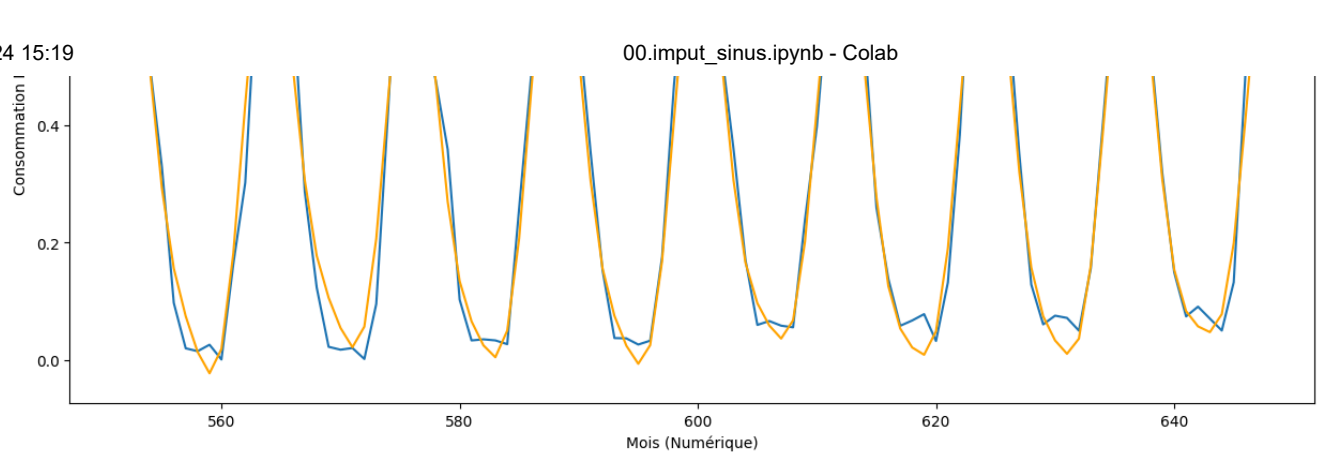


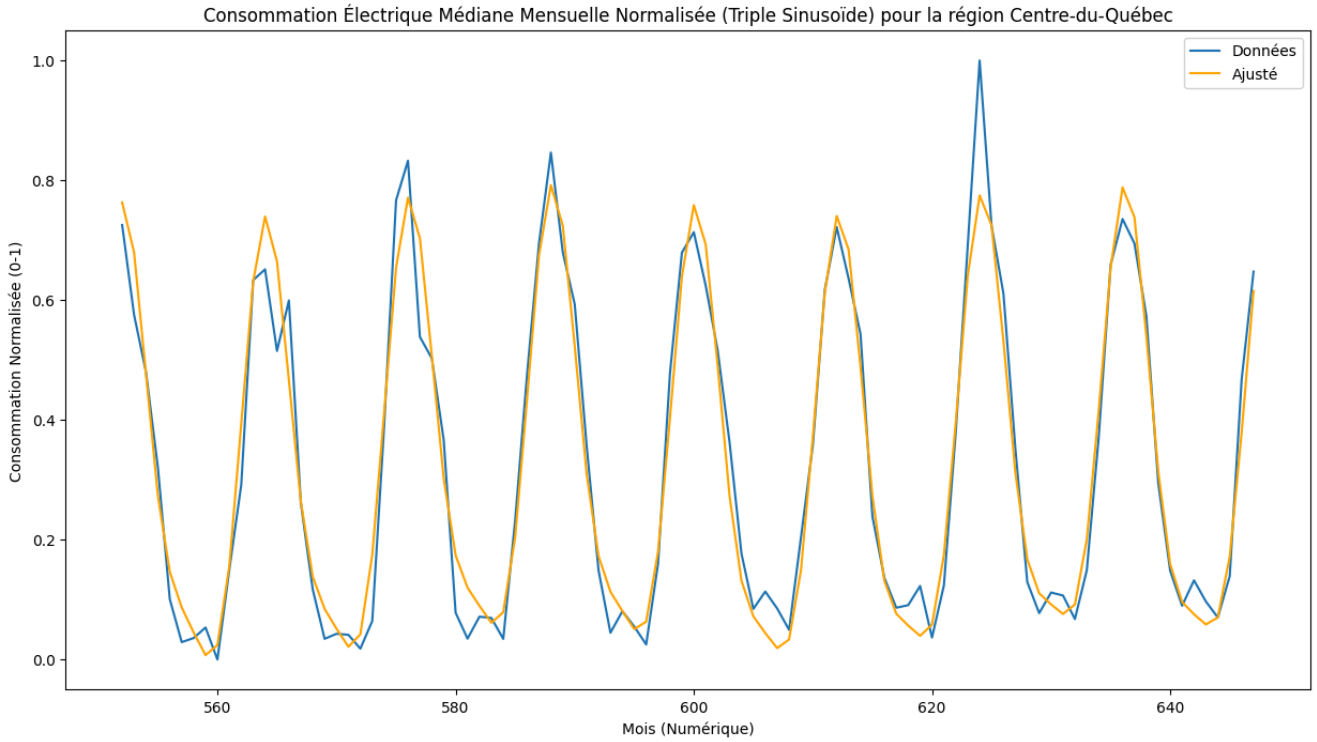

0.2
0.0
560
580
600
620
640
Mois (Numérique)

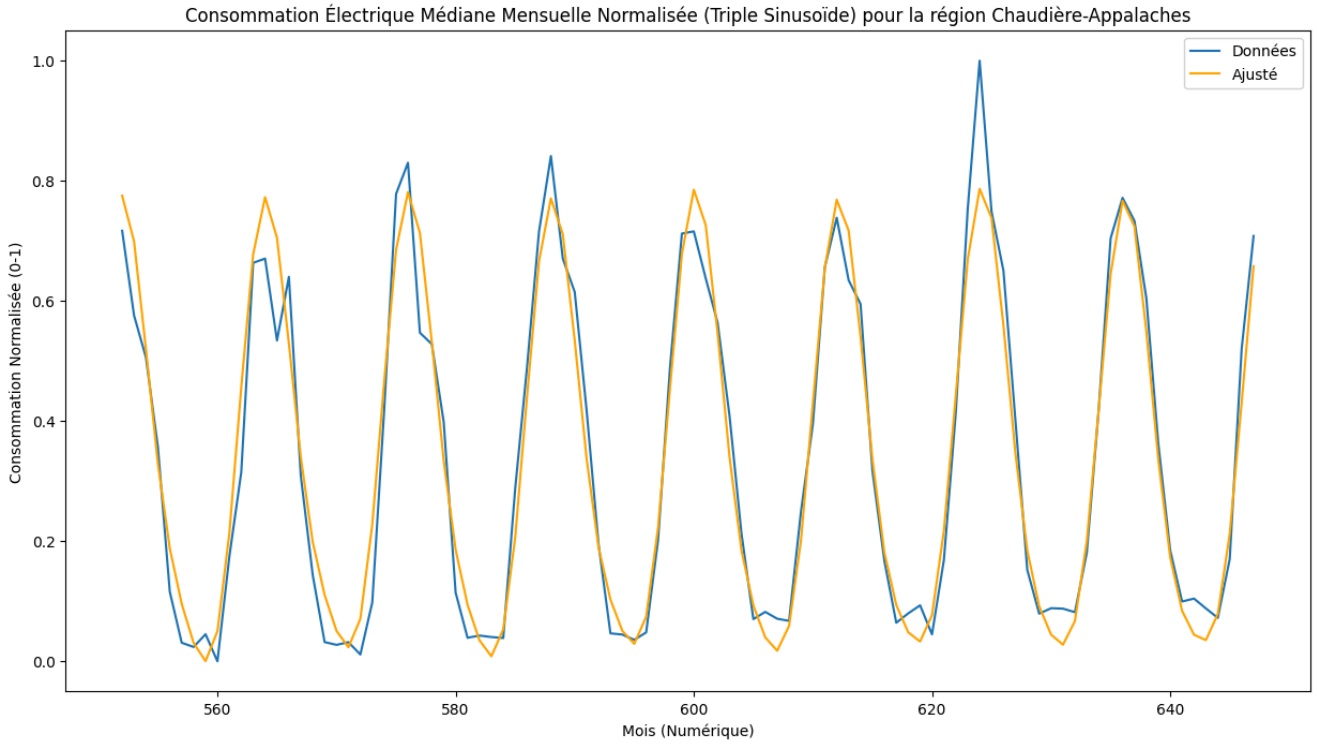

Consommation Électrique Médiane Mensuelle Normalisée (Triple Sinusoïde) pour la région Nord-du-Québec
Données
Ajusté
Consommation Normalisée (0-1)
1.0
0.8
0.6
0.4
0.2
0.0
560
580
600
620
640
Mois (Numérique)

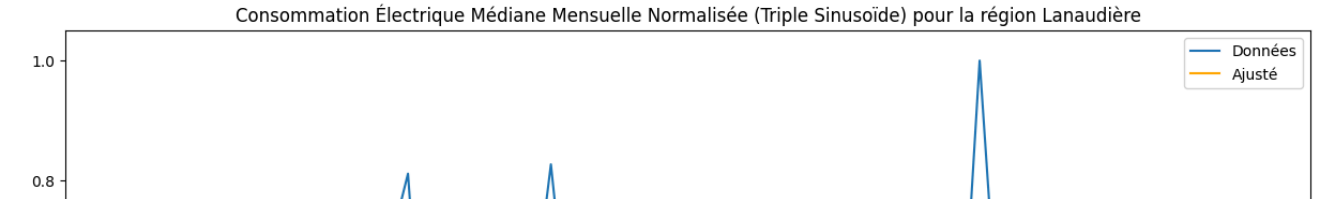

Consommation Électrique Médiane Mensuelle Normalisée (Triple Sinusoïde) pour la région Montréal
Données
Ajusté
Consommation Normalisée (0-1)
1.0
0.8
0.6
0.4
0.2
0.0
560
580
600
620
640
Mois (Numérique)


Consommation Électrique Médiane Mensuelle Normalisée (Triple Sinusoïde) pour la région Capitale-Nationale
Données
Ajusté
1.0
0.8
0.6


Consommation
0.4
0.2
0.0
560
580
600
620
640
Mois (Numérique)


Consommation Électrique Médiane Mensuelle Normalisée (Triple Sinusoïde) pour la région Centre-du-Québec
Données
Ajusté
Consommation Normalisée (0-1)
1.0
0.8
0.6
0.4
0.2
0.0
560
580
600
620
640
Mois (Numérique)


Consommation Électrique Médiane Mensuelle Normalisée (Triple Sinusoïde) pour la région Chaudière-Appalaches
Données
Ajusté
Consommation Normalisée (0-1)
1.0
0.8
0.6
0.4
0.2
0.0
560
580
600
620
640
Mois (Numérique)


Consommation Électrique Médiane Mensuelle Normalisée (Triple Sinusoïde) pour la région Lanaudière
Données
Ajusté
1.0
0.8

PART 0 - IMPORT FILES, CODE_MUN, GEOCODING

drive + librairies

[ ] ↳ *3 cellules masquées*

code_mun

```
code_mun = pd.read_csv(r'/content/gdrive/MyDrive/mémoire/Données/code_geo_mun.csv', encoding= "utf-8")
```

```
code_mun.head()
```

	mcode	munnom	madr1	madr2	madr3	madr4	mcodpos	mcourriel	mweb	mtel	
0	46005	Abercorn	10, rue des Églises O.	Abercorn	10, rue des Églises O.	Abercorn	J0E1B0	info@abercorn.ca	https://municipalites-du-quebec.ca/abercorn/bi...	4505382664	
1	48028	Acton Vale	1025, rue Boulay	Acton Vale	1025, rue Boulay	Acton Vale	J0H1A0	actonvale@ville.actonvale.qc.ca	www.ville.actonvale.qc.ca	4505462703	45
2	31056	Adstock	35, rue Principale O.	Adstock	35, rue Principale O.	Adstock	G0N1S0	dg@adstock.ca	www.adstock.ca	4184222135	41
3	98030	Aguanish	106, route Jacques-Cartier	Aguanish	C.P. 47	Aguanish	G0G1A0	direction@mun.aguanish.org	www.aguanish.org	4185332323	41
4	99125	Akulivik	C.P. 50	Akulivik	C.P. 50	Akulivik	J0M1V0	sec-treasurer@nvakulivik.ca	www.nvakulivik.ca	8194962222	81

```
# ajouter "24" devant tous les mcode (car ils sont dans ce format dans les census)
code_mun['mcode'] = '24' + code_mun['mcode'].astype(str)

#change la colonne regadm : j'ai ça "Estrie (05)" et je veux ça : "Estrie"
code_mun['regadm'] = code_mun['regadm'].str.split(" ").str[0].str.strip()

#seulement garder les colonnes mcode, munnom, mdes, regadm
code_mun = code_mun[['mcode', 'munnom', 'mdes', 'regadm', 'mcodpos']]
```

```
len(code_mun)
```

```
1284
```

```
code_mun.head(15)
```

Afficher la sortie masquée

```
print(code_mun.loc[code_mun["munnom"] == "Les Îles-de-la-Madeleine"])
```

Afficher la sortie masquée

geocoding tableau

```
geocoding = pd.read_csv(r'/content/gdrive/MyDrive/mémoire/Données/geocoding_py.csv', encoding= "latin1", sep = ';' )
```

```
geocoding.head()
```

	Location	Latitude	Longitude
0	Rivière-Bonjour	48.65	-66.88
1	Rivière-Bonjour	48.67	-66.76
2	Rivière-Bonjour	48.65	-66.93
3	Rivière-Bonjour	48.67	-66.76
4	Rivière-Bonjour	48.65	-66.94

```
# garder seulement la premiere ligne pour chaque location

geocoding = geocoding.groupby('Location').first().reset_index()
geocoding.head()
```

	Location	Latitude	Longitude
0	Abercorn	45.03	-72.66
1	Acton Vale	45.65	-72.56
2	Adstock	46.00	-71.08
3	Aguanish	50.22	-62.08
4	Akulivik	60.80	-78.20

```
# ajouter ces infos :
#Lac-Simon  45.899521   -75.097411 Québec   46.813580   -71.238054 Beloeil  45.568881   -73.204193
#Betsiamites    48.940691   -68.642567 Contrecoeur  45.862567   -73.242794 Essipit  48.336988   -69.400486
#Gouvernement régional d'Eeyou Istchee Baie-James   49.757738   -77.621798 Kebaowek 46.787153   -78.984080
#Notre-Dame-du-Sacré-Coeur-d'Issoudun   46.578116   -71.61696

# Create a DataFrame with the new data
new_data = pd.DataFrame({
    'Location': ['Betsiamites', 'Contrecoeur', 'Essipit',
                 'Gouvernement régional d\'Eeyou Istchee Baie-James', 'Kebaowek',
                 'Notre-Dame-du-Sacré-Coeur-d\'Issoudun', 'Saint-Mathieu-de-Beloeil'],
    'Latitude': [48.940691, 45.862567, 48.336988, 49.757738, 46.787153, 46.578116, 45.607257],
    'Longitude': [-68.642567, -73.242794, -69.400486, -77.621798, -78.984080, -71.61696, -73.249765]
})

geocoding = pd.concat([geocoding, new_data], ignore_index=True)
geocoding = geocoding.groupby('Location').first().reset_index()

geocoding.head()
```

	Location	Latitude	Longitude
0	Abercorn	45.03	-72.66
1	Acton Vale	45.65	-72.56
2	Adstock	46.00	-71.08
3	Aguanish	50.22	-62.08
4	Akulivik	60.80	-78.20

```
geocoding[geocoding['Location'] == 'Beloeil']
```

	Location	Latitude	Longitude

```
geocoding.to_csv('geocoding_clean.csv', encoding = 'latin1')
!cp geocoding_clean.csv /content/gdrive/MyDrive/mémoire/Données/
```

PART 1 - HYDRO QC DATA

```
#-------------------------------------------------------------------------------
# Import files and do basic cleaning
#-------------------------------------------------------------------------------
# Electricity data from hydro QC
```

```
elec_df_raw = pd.read_excel(r'/content/gdrive/MyDrive/mémoire/Données/consommation-historique-municipalite-11mars2024.xlsx')
elec_res = elec_df_raw[elec_df_raw['SECTEUR'] == "RÉSIDENTIEL"]
```

EXPLORATION

Général

[ ] ↳ *4 cellules masquées*

Hiver/Autres saisons

```
# Sélectionner les mois d'hiver (novembre à mars) a partir de 2016
elec_df_raw['ANNEE_MOIS'] = pd.to_datetime(elec_df_raw['ANNEE_MOIS'])
elec_df = elec_df_raw[elec_df_raw['SECTEUR'] == "RÉSIDENTIEL"]
elec_df_hiver = elec_df.loc[elec_df_raw['ANNEE_MOIS'].dt.month.isin([11, 12, 1, 2, 3])]

# Calculer le nombre de valeurs manquantes par municipalité
missing_counts = elec_df_hiver.groupby(['NOM_MUNICIPALITE', 'REGION_ADM_QC_TXT',    'MRC_TXT'])['Total (kWh)'].apply(lambda x: x.isnull(

# Créer un tableau de fréquences des nombres de valeurs manquantes
frequency_table = missing_counts.value_counts().sort_index()

# Ajouter une colonne pour les pourcentages
total_municipalities = len(missing_counts)
frequency_table_percent = frequency_table / total_municipalities * 100

# Ajouter une colonne pour les pourcentages cumulés
frequency_table_percent_cum = frequency_table_percent.cumsum()

# Afficher le tableau de fréquences
result_df = pd.DataFrame({
    'Nombre de valeurs manquantes': frequency_table.index,
    'Nombre de municipalités': frequency_table.values,
    'Pourcentage de municipalités': frequency_table_percent.values,
    'Pourcentage de municipalités (cumulé)': frequency_table_percent_cum.values
})

result_df
```

Afficher la sortie masquée

```
# Calculer le nombre de valeurs manquantes par municipalité
missing_counts = elec_df.groupby(['NOM_MUNICIPALITE', 'REGION_ADM_QC_TXT',  'MRC_TXT'])['Total (kWh)'].apply(lambda x: x.isnull().sum()

# Créer un tableau de fréquences des nombres de valeurs manquantes
frequency_table = missing_counts.value_counts().sort_index()

# Ajouter une colonne pour les pourcentages
total_municipalities = len(missing_counts)
frequency_table_percent = frequency_table / total_municipalities * 100

# Ajouter une colonne pour les pourcentages cumulés
frequency_table_percent_cum = frequency_table_percent.cumsum()

# Afficher le tableau de fréquences
result_df = pd.DataFrame({
    'Nombre de valeurs manquantes': frequency_table.index,
    'Nombre de municipalités': frequency_table.values,
    'Pourcentage de municipalités': frequency_table_percent.values,
    'Pourcentage de municipalités (cumulé)': frequency_table_percent_cum.values
})

result_df
```

	Nombre de valeurs manquantes	Nombre de municipalités	Pourcentage de municipalités	Pourcentage de municipalités (cumulé)
0	0	220	20.62	20.62
1	1	13	1.22	21.84
2	2	11	1.03	22.87
3	3	5	0.47	23.34
4	4	7	0.66	23.99
5	5	5	0.47	24.46
6	6	1	0.09	24.55
7	7	3	0.28	24.84
8	8	4	0.37	25.21
9	9	1	0.09	25.30
10	10	2	0.19	25.49
11	11	2	0.19	25.68
12	12	4	0.37	26.05
13	14	6	0.56	26.62
14	15	6	0.56	27.18
15	16	1	0.09	27.27
16	17	1	0.09	27.37
17	18	2	0.19	27.55
18	19	1	0.09	27.65
19	20	1	0.09	27.74
20	21	1	0.09	27.84
21	22	2	0.19	28.02
22	23	2	0.19	28.21
23	24	2	0.19	28.40
24	25	1	0.09	28.49
25	27	1	0.09	28.58
26	28	1	0.09	28.68
27	29	1	0.09	28.77
28	30	1	0.09	28.87
29	31	2	0.19	29.05

```
# parmi les municipalités qui ont au moins une valeur Total (kWh), lesquelles n'en ont aucune en hiver ?
elec_df_one_val = elec_df.dropna(subset=['Total (kWh)'])
elec_df_win = elec_df_one_val[(elec_df_one_val['ANNEE_MOIS'].dt.month.isin([11, 12, 1, 2, 3]))]
elec_df_val_win = elec_df_win.dropna(subset=['Total (kWh)'])

print("Le nombre de villes avec au moins une valeur : ")
print(len(elec_df_one_val['NOM_MUNICIPALITE'].unique()))
print("Le nombre de villes avec au moins une valeur en hiver : ")
print(len(elec_df_val_win['NOM_MUNICIPALITE'].unique()))
```

```
Le nombre de villes avec au moins une valeur :
478
Le nombre de villes avec au moins une valeur en hiver :
459
```

40	48	3	0.28	30.83

```
# comparer la consommation médiane en hiver et celle en été

# Sélectionner les mois d'hiver (novembre à mars)
elec_df_hiver = elec_df[(elec_df['ANNEE_MOIS'].dt.month.isin([11, 12, 1, 2, 3]))]

# Sélectionner les mois d'été (juin à août)
elec_df_ete = elec_df[(elec_df['ANNEE_MOIS'].dt.month.isin([4, 5, 6, 7, 8, 9, 10]))]

# Calculer la consommation médiane en hiver
median_hiver = elec_df_hiver['Total (kWh)'].median()

# Calculer la consommation médiane en été
```

```
median_ete = elec_df_ete['Total (kWh)'].median()

# Afficher les résultats
print("Consommation médiane en hiver :", median_hiver)
print("Consommation médiane en été :", median_ete)

# Comparer les consommations
if median_hiver > median_ete:
  print("La consommation médiane est plus élevée en hiver.")
elif median_hiver < median_ete:
  print("La consommation médiane est plus élevée en été.")
else:
  print("La consommation médiane est la même en hiver et en été.")
```

```
Consommation médiane en hiver : 1909231.5
Consommation médiane en été : 920365.0
La consommation médiane est plus élevée en hiver.
```

59	71	2	0.19	34.58

```
# boxplot de la distribution de ma consommation médiane en hiver puis en été par municipalité

import seaborn as sns
import matplotlib.pyplot as plt

# Calculer la consommation médiane en hiver par municipalité
median_hiver_par_municipalite = elec_df_hiver.groupby('NOM_MUNICIPALITE')['Total (kWh)'].median()

# Calculer la consommation médiane en été par municipalité
median_ete_par_municipalite = elec_df_ete.groupby('NOM_MUNICIPALITE')['Total (kWh)'].median()

# Créer un DataFrame pour les données du boxplot
df_boxplot = pd.DataFrame({
    'Saison': ['Hiver'] * len(median_hiver_par_municipalite) + ['Été'] * len(median_ete_par_municipalite),
    'Consommation médiane (kWh)': list(median_hiver_par_municipalite) + list(median_ete_par_municipalite)
})

# Créer le boxplot
plt.figure(figsize=(10, 6))
sns.boxplot(x='Saison', y='Consommation médiane (kWh)', data=df_boxplot)
plt.title('Distribution de la consommation médiane en hiver et en été par municipalité')
plt.show()
```

Afficher la sortie masquée

75	90	2	0.19	38.33

```
# municipalités dont les consommations médianes sont très différentes de ces valeurs (été et hiver)

# Calculer la consommation médiane en hiver et en été par municipalité
median_consommation = elec_df.groupby(['NOM_MUNICIPALITE', elec_df['ANNEE_MOIS'].dt.month.isin([11, 12, 1, 2, 3])])['Total (kWh)'].medi
median_consommation.columns = ['Ete', 'Hiver']

# Calculer l'écart à la médiane globale pour chaque municipalité
median_consommation['Ecart_Ete'] = abs(median_consommation['Ete'] - median_ete)
median_consommation['Ecart_Hiver'] = abs(median_consommation['Hiver'] - median_hiver)

# Trier les municipalités par écart à la médiane
median_consommation_sorted = median_consommation.sort_values(by=['Ecart_Ete', 'Ecart_Hiver'], ascending=False)

# Afficher les municipalités avec les écarts les plus importants
print(median_consommation_sorted.head(10))

# Lister les mois avec leur consommation hivernale et estivale pour une municipalité donnée
municipalite = "Montréal"  # Remplacer par la municipalité souhaitée

# Filtrer les données pour la municipalité sélectionnée
elec_df_municipalite = elec_df[elec_df['NOM_MUNICIPALITE'] == municipalite]

# Calculer la consommation mensuelle
consommation_mensuelle = elec_df_municipalite.groupby(['ANNEE_MOIS'])['Total (kWh)'].sum()

# Afficher les résultats
print("\nConsommation mensuelle pour", municipalite, ":")
print(consommation_mensuelle)
```

Afficher la sortie masquée

MERGE ET CLEANING

```
elec_res = elec_df_raw[elec_df_raw['SECTEUR'] == "RÉSIDENTIEL"]


# supprimer les municipalités qui ont au moins une valeur nulle "Total kWh"

# Supprimer les municipalités avec au moins une valeur nulle dans 'Total (kWh)'
filtered_elec_df = elec_res.groupby('NOM_MUNICIPALITE').filter(lambda x: x['Total (kWh)'].isnull().sum() == 0)



len(filtered_elec_df['NOM_MUNICIPALITE'].unique())
```

```
226
```

```
# on s'intéresse aux noms de municipalités présents dans plusieurs mrc et/ou régions

# Grouper les lignes par NOM_MUNICIPALITE et agrège les valeurs de MRC_TXT dans une liste
municipality_mrc = filtered_elec_df.groupby('NOM_MUNICIPALITE')['MRC_TXT'].agg(set).reset_index()

# Filtrer les municipalités avec plus d'un MRC unique
municipality_mrc_multiple = municipality_mrc[municipality_mrc['MRC_TXT'].apply(len) > 1]

# Afficher le résultat
print(municipality_mrc_multiple)
```

Afficher la sortie masquée

```
# Grouper les lignes par NOM_MUNICIPALITE et agrège les valeurs uniques de REGION_ADM_QC_TXT dans une liste
municipality_region = filtered_elec_df.groupby('NOM_MUNICIPALITE')['REGION_ADM_QC_TXT'].agg(set).reset_index()

# Filtrer les municipalités avec plus d'une région unique
municipality_region_multiple = municipality_region[municipality_region['REGION_ADM_QC_TXT'].apply(len) > 1]

# Afficher le résultat
print(municipality_region_multiple)
```

Afficher la sortie masquée

```
#voir les lignes dupliquées (= aucune)

duplicate_rows = filtered_elec_df[filtered_elec_df.duplicated()]
print(duplicate_rows)
```

```
Empty DataFrame
Columns: [REGION_ADM_QC_TXT, MRC_TXT, NOM_MUNICIPALITE, ANNEE_MOIS, SECTEUR, Total (kWh)]
Index: []
```

```
# modifier Pessamit dans code_num en Betsiamites et Courcelles--Saint-Évariste en Courcelles --> deux municipalités qui ont changé de nc
#et qui n'ont pas été actualisées par HQ

code_mun.loc[code_mun['munnom'] == 'Pessamit', 'munnom'] = 'Betsiamites'
code_mun.loc[code_mun['munnom'] == 'Courcelles--Saint-Évariste', 'munnom'] = 'Courcelles'



# Merge elec_df et code_mun sur municipalité et region
merged_df = pd.merge(filtered_elec_df, code_mun, left_on=['NOM_MUNICIPALITE', 'REGION_ADM_QC_TXT'], right_on=['munnom', 'regadm'], how=


merged_df.head()
```

	REGION_ADM_QC_TXT	MRC_TXT	NOM_MUNICIPALITE	ANNEE_MOIS	SECTEUR	Total (kWh)	mcode	munnom	mdes	regadm	mcodpo
0	Abitibi-Témiscamingue	NaN	Rouyn-Noranda	2016-01	RÉSIDENTIEL	51418994.00	2486042	Rouyn-Noranda	Ville	Abitibi-Témiscamingue	J9X5C
1	Abitibi-Témiscamingue	NaN	Rouyn-Noranda	2016-02	RÉSIDENTIEL	44733665.00	2486042	Rouyn-Noranda	Ville	Abitibi-Témiscamingue	J9X5C
2	Abitibi-Témiscamingue	NaN	Rouyn-Noranda	2016-03	RÉSIDENTIEL	39834665.00	2486042	Rouyn-Noranda	Ville	Abitibi-Témiscamingue	J9X5C
3	Abitibi-Témiscamingue	NaN	Rouyn-Noranda	2016-04	RÉSIDENTIEL	31622756.00	2486042	Rouyn-Noranda	Ville	Abitibi-Témiscamingue	J9X5C
4	Abitibi-Témiscamingue	NaN	Rouyn-Noranda	2016-05	RÉSIDENTIEL	20922463.00	2486042	Rouyn-Noranda	Ville	Abitibi-Témiscamingue	J9X5C

```
# del munnom_x  mdes_x  regadm_x mcode_y    munnom_y    mdes_y  regadm_y    mcodpos_y
```

```
merged_df = merged_df.drop(['munnom', 'mdes', 'regadm', 'SECTEUR'], axis=1)



# Afficher toutes les municipalités de merged_df qui n'ont pas de code_geo
missing_codes = merged_df[merged_df['mcode'].isna()]['NOM_MUNICIPALITE'].unique()
print(missing_codes)
```

```
['Courcelles']
```

```
# récupèrer dans df_courcelles et les informations sur courcelles de code_mun puis ajoute les à merged_df pour avoir les colonnes mcode

# Récupérer les informations sur Courcelles dans code_mun
df_courcelles = code_mun[code_mun['munnom'] == 'Courcelles']

# Extraire les valeurs de 'mcode' et 'mcodpos'
mcode_courcelles = df_courcelles['mcode'].iloc[0]
mcodpos_courcelles = df_courcelles['mcodpos'].iloc[0]

# Ajouter les valeurs à merged_df pour Courcelles
merged_df.loc[merged_df['NOM_MUNICIPALITE'] == 'Courcelles', 'mcode'] = mcode_courcelles
merged_df.loc[merged_df['NOM_MUNICIPALITE'] == 'Courcelles', 'mcodpos'] = mcodpos_courcelles



merged_df.loc[merged_df['NOM_MUNICIPALITE'] == 'Courcelles']
```

	REGION_ADM_QC_TXT	MRC_TXT	NOM_MUNICIPALITE	ANNEE_MOIS	Total (kWh)	mcode	mcodpos
11040	Estrie	Le Granit	Courcelles	2016-01	804118.00	2429027	G0M1C0
11041	Estrie	Le Granit	Courcelles	2016-02	706932.00	2429027	G0M1C0
11042	Estrie	Le Granit	Courcelles	2016-03	734365.00	2429027	G0M1C0
11043	Estrie	Le Granit	Courcelles	2016-04	578176.00	2429027	G0M1C0
11044	Estrie	Le Granit	Courcelles	2016-05	393562.00	2429027	G0M1C0
11045	Estrie	Le Granit	Courcelles	2016-06	336693.00	2429027	G0M1C0
11046	Estrie	Le Granit	Courcelles	2016-07	308564.00	2429027	G0M1C0
11047	Estrie	Le Granit	Courcelles	2016-08	324355.00	2429027	G0M1C0
11048	Estrie	Le Granit	Courcelles	2016-09	316731.00	2429027	G0M1C0
11049	Estrie	Le Granit	Courcelles	2016-10	442667.00	2429027	G0M1C0
11050	Estrie	Le Granit	Courcelles	2016-11	516696.00	2429027	G0M1C0
11051	Estrie	Le Granit	Courcelles	2016-12	792423.00	2429027	G0M1C0
11052	Estrie	Le Granit	Courcelles	2017-01	773633.00	2429027	G0M1C0
11053	Estrie	Le Granit	Courcelles	2017-02	676303.00	2429027	G0M1C0
11054	Estrie	Le Granit	Courcelles	2017-03	839013.00	2429027	G0M1C0
11055	Estrie	Le Granit	Courcelles	2017-04	533970.00	2429027	G0M1C0
11056	Estrie	Le Granit	Courcelles	2017-05	399360.00	2429027	G0M1C0
11057	Estrie	Le Granit	Courcelles	2017-06	339685.00	2429027	G0M1C0
11058	Estrie	Le Granit	Courcelles	2017-07	321259.00	2429027	G0M1C0
11059	Estrie	Le Granit	Courcelles	2017-08	330183.00	2429027	G0M1C0
11060	Estrie	Le Granit	Courcelles	2017-09	328823.00	2429027	G0M1C0
11061	Estrie	Le Granit	Courcelles	2017-10	381617.00	2429027	G0M1C0
11062	Estrie	Le Granit	Courcelles	2017-11	590516.00	2429027	G0M1C0
11063	Estrie	Le Granit	Courcelles	2017-12	838365.00	2429027	G0M1C0
11064	Estrie	Le Granit	Courcelles	2018-01	872601.00	2429027	G0M1C0
11065	Estrie	Le Granit	Courcelles	2018-02	701556.00	2429027	G0M1C0
11066	Estrie	Le Granit	Courcelles	2018-03	731734.00	2429027	G0M1C0
11067	Estrie	Le Granit	Courcelles	2018-04	660343.00	2429027	G0M1C0
11068	Estrie	Le Granit	Courcelles	2018-05	393535.00	2429027	G0M1C0
11069	Estrie	Le Granit	Courcelles	2018-06	334486.00	2429027	G0M1C0
11070	Estrie	Le Granit	Courcelles	2018-07	323996.00	2429027	G0M1C0
11071	Estrie	Le Granit	Courcelles	2018-08	324597.00	2429027	G0M1C0
11072	Estrie	Le Granit	Courcelles	2018-09	333198.00	2429027	G0M1C0
11073	Estrie	Le Granit	Courcelles	2018-10	505271.00	2429027	G0M1C0
11074	Estrie	Le Granit	Courcelles	2018-11	655850.00	2429027	G0M1C0
11075	Estrie	Le Granit	Courcelles	2018-12	806906.00	2429027	G0M1C0
11076	Estrie	Le Granit	Courcelles	2019-01	893092.00	2429027	G0M1C0
11077	Estrie	Le Granit	Courcelles	2019-02	790741.00	2429027	G0M1C0
11078	Estrie	Le Granit	Courcelles	2019-03	797964.00	2429027	G0M1C0
11079	Estrie	Le Granit	Courcelles	2019-04	689509.00	2429027	G0M1C0
11080	Estrie	Le Granit	Courcelles	2019-05	487381.00	2429027	G0M1C0
11081	Estrie	Le Granit	Courcelles	2019-06	349926.00	2429027	G0M1C0
11082	Estrie	Le Granit	Courcelles	2019-07	319979.00	2429027	G0M1C0
11083	Estrie	Le Granit	Courcelles	2019-08	310038.00	2429027	G0M1C0
11084	Estrie	Le Granit	Courcelles	2019-09	342667.00	2429027	G0M1C0
11085	Estrie	Le Granit	Courcelles	2019-10	452603.00	2429027	G0M1C0
11086	Estrie	Le Granit	Courcelles	2019-11	667633.00	2429027	G0M1C0
11087	Estrie	Le Granit	Courcelles	2019-12	815449.00	2429027	G0M1C0
11088	Estrie	Le Granit	Courcelles	2020-01	804750.00	2429027	G0M1C0

11089	Estrie	Le Granit	Courcelles	2020-02	754039.00	2429027	G0M1C0
11090	Estrie	Le Granit	Courcelles	2020-03	735086.00	2429027	G0M1C0
11091	Estrie	Le Granit	Courcelles	2020-04	641483.00	2429027	G0M1C0
11092	Estrie	Le Granit	Courcelles	2020-05	453333.00	2429027	G0M1C0
11093	Estrie	Le Granit	Courcelles	2020-06	332507.00	2429027	G0M1C0
11094	Estrie	Le Granit	Courcelles	2020-07	323321.00	2429027	G0M1C0
11095	Estrie	Le Granit	Courcelles	2020-08	324512.00	2429027	G0M1C0
11096	Estrie	Le Granit	Courcelles	2020-09	341390.00	2429027	G0M1C0
11097	Estrie	Le Granit	Courcelles	2020-10	466051.00	2429027	G0M1C0
11098	Estrie	Le Granit	Courcelles	2020-11	568979.00	2429027	G0M1C0
11099	Estrie	Le Granit	Courcelles	2020-12	749193.00	2429027	G0M1C0
11100	Estrie	Le Granit	Courcelles	2021-01	825367.00	2429027	G0M1C0
11101	Estrie	Le Granit	Courcelles	2021-02	751246.00	2429027	G0M1C0
11102	Estrie	Le Granit	Courcelles	2021-03	759648.00	2429027	G0M1C0
11103	Estrie	Le Granit	Courcelles	2021-04	546353.00	2429027	G0M1C0
11104	Estrie	Le Granit	Courcelles	2021-05	416528.00	2429027	G0M1C0
11105	Estrie	Le Granit	Courcelles	2021-06	322738.00	2429027	G0M1C0
11106	Estrie	Le Granit	Courcelles	2021-07	333222.00	2429027	G0M1C0
11107	Estrie	Le Granit	Courcelles	2021-08	341290.00	2429027	G0M1C0
11108	Estrie	Le Granit	Courcelles	2021-09	329480.00	2429027	G0M1C0
11109	Estrie	Le Granit	Courcelles	2021-10	427595.00	2429027	G0M1C0
11110	Estrie	Le Granit	Courcelles	2021-11	599486.00	2429027	G0M1C0
11111	Estrie	Le Granit	Courcelles	2021-12	803997.00	2429027	G0M1C0
11112	Estrie	Le Granit	Courcelles	2022-01	989590.00	2429027	G0M1C0
11113	Estrie	Le Granit	Courcelles	2022-02	818148.00	2429027	G0M1C0
11114	Estrie	Le Granit	Courcelles	2022-03	816225.00	2429027	G0M1C0
11115	Estrie	Le Granit	Courcelles	2022-04	651129.00	2429027	G0M1C0
11116	Estrie	Le Granit	Courcelles	2022-05	404881.00	2429027	G0M1C0
11117	Estrie	Le Granit	Courcelles	2022-06	333283.00	2429027	G0M1C0
11118	Estrie	Le Granit	Courcelles	2022-07	317307.00	2429027	G0M1C0
11119	Estrie	Le Granit	Courcelles	2022-08	320979.00	2429027	G0M1C0
11120	Estrie	Le Granit	Courcelles	2022-09	355115.00	2429027	G0M1C0
11121	Estrie	Le Granit	Courcelles	2022-10	455632.00	2429027	G0M1C0
11122	Estrie	Le Granit	Courcelles	2022-11	595842.00	2429027	G0M1C0
11123	Estrie	Le Granit	Courcelles	2022-12	786763.00	2429027	G0M1C0
11124	Estrie	Le Granit	Courcelles	2023-01	825155.00	2429027	G0M1C0
11125	Estrie	Le Granit	Courcelles	2023-02	812337.00	2429027	G0M1C0
11126	Estrie	Le Granit	Courcelles	2023-03	763313.00	2429027	G0M1C0
11127	Estrie	Le Granit	Courcelles	2023-04	584036.00	2429027	G0M1C0
11128	Estrie	Le Granit	Courcelles	2023-05	426413.00	2429027	G0M1C0
11129	Estrie	Le Granit	Courcelles	2023-06	349787.00	2429027	G0M1C0
11130	Estrie	Le Granit	Courcelles	2023-07	343703.00	2429027	G0M1C0
11131	Estrie	Le Granit	Courcelles	2023-08	334212.00	2429027	G0M1C0
11132	Estrie	Le Granit	Courcelles	2023-09	344842.00	2429027	G0M1C0
11133	Estrie	Le Granit	Courcelles	2023-10	426641.00	2429027	G0M1C0
11134	Estrie	Le Granit	Courcelles	2023-11	676831.00	2429027	G0M1C0
11135	Estrie	Le Granit	Courcelles	2023-12	801253.00	2429027	G0M1C0

```
#Definir une fonction pour calculer la médiane et la moyenne des mois hivernaux et non-hivernaux
def calculate_stats(df):
  winter_months = [11, 12, 1, 2, 3]
  is_winter = df['ANNEE_MOIS'].dt.month.isin(winter_months)
  winter_df = df[is_winter]
  summer_df = df[~is_winter]
  # Include 'REGION_ADM_QC_TXT' and 'mcode' in the result
  return pd.Series({
      'median_total_kwh_winter': winter_df['Total (kWh)'].median(),
      'mean_total_kwh_winter': winter_df['Total (kWh)'].mean(),
      'median_total_kwh_summer': summer_df['Total (kWh)'].median(),
      'mean_total_kwh_summer': summer_df['Total (kWh)'].mean(),
      'median_total_kwh_all': df['Total (kWh)'].median(),
      'mean_total_kwh_all': df['Total (kWh)'].mean(),
      'REGION_ADM_QC_TXT': df['REGION_ADM_QC_TXT'].iloc[0],  # Take the first value, assuming it's the same for the group
      'mcode': df['mcode'].iloc[0],  # Take the first value, assuming it's the same for the group
      'mcodpos' : df['mcodpos'].iloc[0]
  })

#appliquer la fonction
merged_df['ANNEE_MOIS'] = pd.to_datetime(merged_df['ANNEE_MOIS'])
merged_df_cleaned = merged_df.groupby(['NOM_MUNICIPALITE', 'REGION_ADM_QC_TXT']).apply(calculate_stats).reset_index()
```

```
<ipython-input-32-833ecd00a184>:22: DeprecationWarning: DataFrameGroupBy.apply operated on the grouping columns. This behavior is de
  merged_df_cleaned = merged_df.groupby('NOM_MUNICIPALITE').apply(calculate_stats).reset_index()
```

```
# renommer les colonnes 'REGION_ADM_QC_TXT', 'NOM_MUNICIPALITE', 'mcode' par    "region" "nom_geo" "code_geo"

grouped_df = merged_df_cleaned.rename(columns={'REGION_ADM_QC_TXT': 'region', 'NOM_MUNICIPALITE': 'nom_geo', 'mcode': 'code_geo', 'mcod


grouped_df.head(100)
```

	nom_geo	median_total_kwh_winter	mean_total_kwh_winter	median_total_kwh_summer	mean_total_kwh_summer	median_total_
0	Acton Vale	7471105.50	7342004.72	3520431.50	3774089.86	46
1	Aguanish	317612.00	308225.33	138171.50	156157.62	2
2	Auclair	447692.50	444469.03	234309.50	262890.45	3
3	Aupaluk	70664.50	70376.93	49022.50	50473.59	5
4	Baie-D'Urfé	5503273.50	5507386.97	2764132.50	2826615.64	31
5	Baie-Saint-Paul	8416780.00	8339247.85	3589910.50	4035432.70	53
6	Baie-des-Sables	617461.50	618386.03	288508.50	326723.04	4
7	Baie-du-Febvre	1035131.50	1018802.15	436368.50	475602.30	6
8	Beauceville	5854756.50	5740464.05	2818868.00	3067680.89	38
9	Beaumont	3101225.00	3054816.05	1633573.50	1702996.16	20
10	Beloeil	21852200.00	21488264.30	11533970.00	11745012.05	131
11	Berthierville	4449149.00	4418030.70	1973401.00	2069267.77	25
12	Betsiamites	2219117.00	2185111.98	1198561.00	1246442.05	16
13	Blainville	59200809.50	58893108.05	33333495.50	33646212.05	369
14	Blanc-Sablon	1690489.00	1670062.25	871046.50	899859.27	11
15	Boileau	711279.50	713385.75	270538.00	302941.95	4
16	Boisbriand	25999147.50	25586547.07	13026717.00	13396551.43	150
17	Boucherville	44558872.00	43300627.35	23392481.50	23938444.73	262
18	Bromont	14236069.50	14080221.70	6704538.00	6769939.23	79
19	Bryson	744010.50	732495.47	328003.00	351988.91	4
20	Brébeuf	1396546.50	1385771.30	563888.50	623983.07	8
21	Bury	1150617.00	1121192.32	551930.50	609873.75	7
22	Carleton-sur-Mer	4562149.00	4515891.53	1998456.50	2295396.98	30
23	Chisasibi	4362215.50	4458756.00	2535328.00	2756124.62	32
24	Châteauguay	47474280.50	46670450.67	23015713.50	23680468.45	271
25	Cookshire-Eaton	5097608.00	4977061.85	2478297.00	2696562.54	33
26	Coteau-du-Lac	7653693.50	7522339.75	3854350.50	3946315.64	44
27	Courcelles	780198.00	756078.45	344272.50	400195.30	5
28	Daveluyville	2321385.50	2307643.42	1126555.00	1207995.86	14
29	Deux-Montagnes	18355316.50	17884042.90	8064543.00	8467813.84	102
30	Dixville	532925.00	523286.45	280796.00	300412.46	3
31	Drummondville	76536225.50	75040193.47	35729824.50	37602636.93	455
32	Dundee	565186.50	559695.53	300912.50	308796.21	3
33	Dunham	4360974.00	4322064.90	1936633.00	2136599.70	26
34	Dupuy	847941.00	851723.85	397806.50	455197.61	5
35	Fossambault-sur-le-Lac	3511294.00	3472294.92	1474193.00	1603886.71	21
36	Franquelin	433898.00	427233.80	163747.50	191069.04	2
37	Granby	64830026.50	63902717.95	30641300.00	32110408.61	388
38	Grand-Métis	234771.00	238878.33	123868.00	134095.46	1
39	Hampstead	10491135.00	10376253.70	4897502.50	5072579.96	58
40	Hatley	3704520.00	3654674.40	1633779.00	1754772.91	21

41	Hemmingford	3141429.00	3082601.88	1443617.50	1505463.32	18
42	Hébertville	2753744.00	2742903.80	1214524.00	1367008.61	178
43	Kangiqsualujjuaq	200560.00	200762.90	154221.50	155754.64	16
44	Kangiqsujuaq	244558.50	254755.25	190851.00	192055.27	2
45	Kangirsuk	151921.50	153706.17	109927.00	111747.36	12
46	Kiamika	934453.50	934232.35	414434.00	458189.52	58
47	Kuujjuaq	894365.50	888107.82	637771.00	648609.46	7
48	Kuujjuarapik	242540.00	249036.10	182985.00	186961.30	20
49	L'Ascension	1496209.50	1491023.25	591903.00	659452.41	87
50	L'Assomption	23218078.00	23073554.52	11550421.00	11776775.05	135
51	L'Isle-Verte	1213797.50	1203753.30	571843.50	643187.11	84
52	L'Islet	3370303.00	3346565.27	1542417.50	1740983.54	228
53	L'Île-du-Grand-Calumet	848567.50	846625.65	370523.50	397589.12	49
54	La Malbaie	10148080.50	10131253.82	4131686.00	4813861.62	667
55	La Pocatière	4069522.00	4073481.55	1622983.00	1923362.96	268
56	La Prairie	24522589.00	24337274.05	12181899.50	12538979.36	1410
57	La Sarre	8131545.00	8072991.55	3337558.50	3902281.57	548
58	Labrecque	1568882.00	1557657.82	755216.00	810530.25	10
59	Lac-Bouchette	1330329.50	1338861.38	626598.50	693637.27	87
60	Lac-Casault	23423.50	23500.17	10252.00	11027.07	
61	Lac-Etchemin	4564986.00	4516459.25	2025749.50	2300481.95	303
62	Lac-Pythonga	26541.00	25927.72	11279.00	12379.30	
63	Lac-Saint-Paul	709740.00	704399.40	327815.00	345775.61	43
64	Lac-Sainte-Marie	2201710.00	2161548.23	646992.50	743158.70	109
65	Latulipe-et-Gaboury	323960.50	323672.80	133980.50	154642.66	20
66	Launay	222098.00	218164.20	102724.50	113752.07	14
67	Laurier-Station	2265299.00	2264964.75	1160987.00	1225423.29	149
68	Laurierville	1056157.00	1045259.62	604524.00	633212.50	74
69	Laval	418257340.00	409326896.85	188451917.00	195014689.95	23318
70	Lavaltrie	15118641.00	14824219.25	7339229.50	7590380.82	878
71	Leclercville	549475.00	537707.90	230364.00	249736.21	32
72	Les Hauteurs	418727.00	422017.78	212294.50	238015.46	30
73	Les Îles-de-la-Madeleine	11623910.00	11336195.78	5981359.50	6439575.96	791
74	Longueuil	219380752.00	213086243.88	100539723.50	103753469.52	12267
75	Lorraine	9093364.00	8928601.47	6008058.00	6038741.89	664
76	Louiseville	7514901.00	7453062.00	3288032.50	3517224.62	440

77	Marston	802012.00	787193.88	384803.50	422155.57	52
78	Mascouche	46830682.00	45712677.95	24490077.00	24967712.00	280
79	Maskinongé	2562120.50	2527943.05	1116022.50	1208331.80	14
80	McMasterville	5446226.50	5339624.40	2687550.00	2751729.71	31
81	Mercier	13807374.00	13705739.88	7343913.00	7487704.82	83
82	Mirabel	55516530.50	55263661.77	29762408.00	29670686.04	334
83	Moffet	236677.50	235275.00	112037.50	121819.32	1
84	Mont-Royal	24299572.50	23939962.65	11234661.50	11629138.11	130
85	Mont-Tremblant	24743911.00	24372364.62	8898184.50	9998331.88	134
86	Montcerf-Lytton	739888.00	738888.07	394032.50	412638.59	4
87	Montmagny	11378289.50	11211150.93	4748895.00	5402916.09	73
88	Montréal	1498221974.50	1462671289.33	578251505.50	626078839.77	8103
89	Natashquan	519965.50	560032.10	299217.00	331195.07	4
90	Neuville	5133534.00	5082352.12	2350740.50	2506702.54	32
91	Nicolet	8501720.50	8435936.53	3827177.50	4080533.79	50
92	North Hatley	1405025.00	1381779.45	556698.00	611481.52	7
93	Notre-Dame-de-Bonsecours	446020.50	448347.92	195054.00	206629.57	2
94	Notre-Dame-des-Bois	1170995.00	1188313.98	507071.00	574038.16	7
95	Notre-Dame-du-Laus	3236576.50	3286068.35	1411039.50	1528939.41	19
96	Notre-Dame-du-Portage	1842151.00	1811648.02	756594.50	861639.73	12
97	Notre-Dame-du-Rosaire	384250.00	385333.20	180789.00	199215.55	2
98	Nédélec	329775.00	324342.25	135281.50	157750.25	2
99	Orford	8183840.50	8054237.72	3145278.00	3500256.21	45

csv

```
grouped_df.to_csv('grouped_df.csv', encoding = 'latin1')
!cp grouped_df.csv /content/gdrive/MyDrive/mémoire/Données/
```

PART 2 - CENSUS DATA

a. Census data 2021

```
# ------------------------------------------------------------------------
# CENSUS DATA 2021
df = pd.read_csv(r'/content/gdrive/MyDrive/mémoire/Données/Census 2021/clean_stat_can_21_utf.csv', encoding= "utf_8")

# Renommer colonnes
df = df.rename(columns={'CODE_GÉO_ALT':'code_geo', 'NOM_GÉO' : 'nom_geo', 'NOM_CARACTÉRISTIQUE' : 'nom_car', 'C1_CHIFFRE_TOTAL' : 'total
                        })
```

```
<ipython-input-4-93bcc83db9d7>:3: DtypeWarning: Columns (1) have mixed types. Specify dtype option on import or set low_memory=False
  df = pd.read_csv(r'/content/gdrive/MyDrive/mémoire/Données/Census 2021/clean_stat_can_21_utf.csv', encoding= "utf_8")
```

```
len(df)
```

```
39959628
```

```
df.head()
```

	code_geo	nom_geo	nom_car	total
0	1	Canada	Population, 2021	36991981.00
1	1	Canada	Population, 2016	35151728.00
2	1	Canada	Variation en pourcentage de la population, 201...	5.20
3	1	Canada	Total des logements privés	16284235.00
4	1	Canada	Logements privés occupés par des résidents hab...	14978941.00

```
##garder seulement les lignes où le code géographique est composé de 7 chiffres (les municipalités)

df_filtered = df[df['code_geo'].astype(str).str.len() == 7]


#j'ai ça dans ma colonne NOM_GEO : "Les Îles-de-la-Madeleine, Municipalité (MÉ) " du census 2021 et je veux juste avoir ça : "Les Îles-d

df_filtered['nom_geo'] = df_filtered['nom_geo'].str.split(',').str[0]
```

Afficher la sortie masquée

```
df_filtered.head(250)
```

Afficher la sortie masquée

```
len(df_filtered)
```

```
3372942
```

```
# Cette fonction stocke les changements réalisés
changes = []

def log_change(old_value, new_value):
    changes.append({'Ancien_nom_car': old_value, 'Nouveau_nom_car': new_value})

# Code modifié pour enregistrer les changements de nom_car

### Type de logement
start_indices = df_filtered[df_filtered["nom_car"] == "Total - Logements privés occupés par type de construction résidentielle - Données
row_names = ["Type de construction résidentielle - Maison individuelle non attenante",
             "Type de construction résidentielle - Maison jumelée",
             "Type de construction résidentielle - Maison en rangée",
```

```
             "Type de construction résidentielle - Appartement ou plain-pied dans un duplex",
             "Type de construction résidentielle - Appartement dans un immeuble de moins de cinq étages",
             "Type de construction résidentielle - Appartement dans un immeuble de cinq étages ou plus"
             ]
for index in start_indices:
    for i in range(6):
        old_value = df_filtered.at[index + i + 1, "nom_car"]
        new_value = row_names[i]
        if old_value != new_value:  # To ensure that we only log actual changes
            df_filtered.at[index + i + 1, "nom_car"] = new_value
            log_change(old_value, new_value)

### Revenu des ménages
start_indices = df_filtered[df_filtered["nom_car"] == "Total - Statistiques du revenu pour les ménages privés - Données intégrales (100
row_names = ["Revenu total médian des ménages en 2020($) avant impôts",
             "Revenu total médian des ménages en 2020($) après impôts"]

for index in start_indices:
    for i in range(2):
        old_value = df_filtered.at[index + i + 1, "nom_car"]
        new_value = row_names[i]
        if old_value != new_value:
            df_filtered.at[index + i + 1, "nom_car"] = new_value
            log_change(old_value, new_value)

### Période de construction
start_indices = df_filtered[df_filtered["nom_car"] == "Total - Logements privés occupés selon la période de construction - Données-échar
row_names = [

             "Logement occupé selon période de construction - 1960 ou avant",
             "Logement occupé selon période de construction - 1961 à 1980",
             "Logement occupé selon période de construction - 1981 à 1990",
             "Logement occupé selon période de construction - 1991 à 2000",
             "Logement occupé selon période de construction - 2001 à 2005",
             "Logement occupé selon période de construction - 2006 à 2010",
             "Logement occupé selon période de construction - 2011 à 2015",
             "Logement occupé selon période de construction - 2016 à 2021"
             ]
for index in start_indices:
    for i in range(8):
        old_value = df_filtered.at[index + i + 1, "nom_car"]
        new_value = row_names[i]
        if old_value != new_value:
            df_filtered.at[index + i + 1, "nom_car"] = new_value
            log_change(old_value, new_value)

### Réparations majeures requises
start_indices = df_filtered[df_filtered["nom_car"] == "Total - Logements privés occupés selon l'état du logement - Données-échantillon (
row_names = ["Logement occupé selon l'état du logement - Réparations majeures requises"
             ]
for index in start_indices:
    for i in range(1):
        old_value = df_filtered.at[index + i + 2, "nom_car"]
        new_value = row_names[i]
        if old_value != new_value:
            df_filtered.at[index + i + 2, "nom_car"] = new_value
            log_change(old_value, new_value)

### Mode d'occupation
start_indices = df_filtered[df_filtered["nom_car"] == "Total - Ménages privés selon le mode d'occupation - Données-échantillon (25 %)"]
row_names = ["Ménages privés selon le mode d'occupation - Propriétaire",
             "Ménages privés selon le mode d'occupation - Locataire",
             "Ménages privés selon le mode d'occupation - Logement fourni par le gouvernement local"
             ]
for index in start_indices:
    for i in range(3):
        old_value = df_filtered.at[index + i + 1, "nom_car"]
        new_value = row_names[i]
        if old_value != new_value:
            df_filtered.at[index + i + 1, "nom_car"] = new_value
            log_change(old_value, new_value)

### Education
start_indices = df_filtered[df_filtered["nom_car"] == "Total - Plus haut certificat, diplôme ou grade pour la population âgée de 25 à 64
row_names = ["Plus haut diplôme - Aucun certificat, diplôme ou grade",
             "Plus haut diplôme - Diplôme d'études secondaires ou attestation d'équivalence",
             "Plus haut diplôme - Certificat, diplôme ou grade d’études postsecondaires"
             ]
for index in start_indices:
    for i in range(3):
        old_value = df_filtered.at[index + i + 1, "nom_car"]
        new_value = row_names[i]
```

```
            if old_value != new_value:
                df_filtered.at[index + i + 1, "nom_car"] = new_value
                log_change(old_value, new_value)

    ### Groupes d'âge
    start_indices = df_filtered[df_filtered["nom_car"] == "Total - Groupes d'âge de la population - Données intégrales (100 %)"].index
    row_names = ["Groupes d'âge de la population - 65 ans et plus"]

    for index in start_indices:
        for i in range(1):
            old_value = df_filtered.at[index + i + 16, "nom_car"]
            new_value = row_names[i]
            if old_value != new_value:
                df_filtered.at[index + i + 16, "nom_car"] = new_value
                log_change(old_value, new_value)

    ### Frais de logements
    start_indices = df_filtered[df_filtered["nom_car"] == "Total - Ménages propriétaires et locataires dont le revenu total du ménage est su
    row_names = ["Ménages dont 30 % ou plus du revenu est consacré aux frais de logement seulement"]

    for index in start_indices:
        for i in range(1):
            old_value = df_filtered.at[index + i + 2, "nom_car"]
            new_value = row_names[i]
            if old_value != new_value:
                df_filtered.at[index + i + 2, "nom_car"] = new_value
                log_change(old_value, new_value)

     ### Revenus médians
    start_indices = df_filtered[df_filtered["nom_car"] == "Total - Statistiques du revenu en 2020 pour la population âgée de 15 ans et plus
    row_names = ["Revenu total médian avant impôts en 2020 parmi les bénéficiaires d'un revenu total ($)"]

    for index in start_indices:
        for i in range(1):
            old_value = df_filtered.at[index + i + 2, "nom_car"]
            new_value = row_names[i]
            if old_value != new_value:
                df_filtered.at[index + i + 2, "nom_car"] = new_value
                log_change(old_value, new_value)

    start_indices = df_filtered[df_filtered["nom_car"] == "Total - Statistiques du revenu en 2020 pour la population âgée de 15 ans et plus
    row_names = ["Revenu total médian après impôts en 2020 parmi les bénéficiaires d'un revenu total ($)"]

    for index in start_indices:
        for i in range(1):
            old_value = df_filtered.at[index + i + 4, "nom_car"]
            new_value = row_names[i]
            if old_value != new_value:
                df_filtered.at[index + i + 4, "nom_car"] = new_value
                log_change(old_value, new_value)

    # Crée une DataFrame pour les changements et afficher les valeurs uniques
    change_df = pd.DataFrame(changes)
    unique_changes = change_df.drop_duplicates()
    print(unique_changes)
```

Afficher la sortie masquée

```
    df_filtered.head(500)
```

Afficher la sortie masquée

```
    #Créer toutes les variables et les mettre dans un df

    vars = [
        "Revenu total médian après impôts en 2020 parmi les bénéficiaires d'un revenu total ($)",
        "Revenu total médian avant impôts en 2020 parmi les bénéficiaires d'un revenu total ($)",
        "Revenu total médian des ménages en 2020($) avant impôts",
        "Revenu total médian des ménages en 2020($) après impôts",
        "Population, 2021",
        "Population, 2016",

        "Superficie des terres en kilomètres carrés",
        "Taille moyenne des ménages privés",
        "Nombre moyen de pièces par logement",
        "Logements privés occupés par des résidents habituels",
        "Total des logements privés",
        "Taux de chômage",

        #variables pour calculter mes taux
```

```
    "Total - Groupes d'âge de la population - Données intégrales (100 %)",

    "Total - Logements privés occupés selon l'état du logement - Données-échantillon (25 %)",

    "Total - Logements privés occupés par type de construction résidentielle - Données intégrales (100 %)",

    "Total - Logements privés occupés selon la période de construction - Données-échantillon (25 %)",

    "Total - Ménages privés selon le mode d'occupation - Données-échantillon (25 %)",

    "Total - Plus haut certificat, diplôme ou grade pour la population âgée de 25 à 64 ans dans les ménages privés - Données-échantillor

    "Total - Ménages propriétaires et locataires dont le revenu total du ménage est supérieur à zéro, dans les logements privés non agri
]


rate_vars = [
    "Type de construction résidentielle - Maison individuelle non attenante",
    "Type de construction résidentielle - Maison jumelée",
    "Type de construction résidentielle - Maison en rangée",
    "Type de construction résidentielle - Appartement ou plain-pied dans un duplex",
    "Type de construction résidentielle - Appartement dans un immeuble de moins de cinq étages",
    "Type de construction résidentielle - Appartement dans un immeuble de cinq étages ou plus",

    "Logement occupé selon période de construction - 1960 ou avant",
    "Logement occupé selon période de construction - 1961 à 1980",
    "Logement occupé selon période de construction - 1981 à 1990",
    "Logement occupé selon période de construction - 1991 à 2000",
    "Logement occupé selon période de construction - 2001 à 2005",
    "Logement occupé selon période de construction - 2006 à 2010",
    "Logement occupé selon période de construction - 2011 à 2015",
    "Logement occupé selon période de construction - 2016 à 2021",

    "Logement occupé selon l'état du logement - Réparations majeures requises",

    "Ménages privés selon le mode d'occupation - Propriétaire",
    "Ménages privés selon le mode d'occupation - Locataire",
    "Ménages privés selon le mode d'occupation - Logement fourni par le gouvernement local",

    "Plus haut diplôme - Aucun certificat, diplôme ou grade",
    "Plus haut diplôme - Diplôme d'études secondaires ou attestation d'équivalence",
    "Plus haut diplôme - Certificat, diplôme ou grade d’études postsecondaires",

    "Groupes d'âge de la population - 65 ans et plus",

    "Ménages dont 30 % ou plus du revenu est consacré aux frais de logement seulement"

]


# garder seulement les lignes ou nom_car = var + rate_var

all_vars = vars + rate_vars
df_filtered = df_filtered[df_filtered['nom_car'].isin(all_vars)]



all_vars
```

Afficher la sortie masquée

```
weird_var = set(all_vars) - set(df_filtered['nom_car'].unique())


weird_var
```

```
set()
```

```
df.iloc[1450:1500]
```

Afficher la sortie masquée

```
df_filtered.head(100)
```

Afficher la sortie masquée

```
len(df_filtered['code_geo'].unique())
```

```
1282
```

```
len(df_filtered['nom_geo'].unique())
```

```
1227
```

```
# nom_géo qui ont différents code_géo

# Groupe les lignes par 'nom_geo' et agrège les valeurs uniques de 'code_geo' dans une liste
grouped = df_filtered.groupby('nom_geo')['code_geo'].agg(set).reset_index()

# Filtre les noms_geo avec plus d'un code_geo unique
multiple_codes = grouped[grouped['code_geo'].apply(len) > 1]

# Affiche le résultat
print(multiple_codes)
```

Afficher la sortie masquée

```
# Créer une liste de toutes les municipalités uniques avec leur code géographique
unique_municipalities = df_filtered[['code_geo', 'nom_geo']].drop_duplicates()

# Transformer le DataFrame pour le fusionner avec les variables cibles
# Ici, nous utilisons le pivot pour réorganiser les données
df_pivoted = df_filtered.pivot_table(index=['code_geo', 'nom_geo'],
                                     columns='nom_car',
                                     values='total',
                                     fill_value=np.nan).reset_index()

# Maintenant, fusionner les municipalités uniques avec le DataFrame pivoté
df_final = unique_municipalities.merge(df_pivoted, on=['code_geo', 'nom_geo'], how='left')
```

```
df_final.head()
```

	code_geo	nom_geo	Groupes d'âge de la population - 65 ans et plus	Logement occupé selon l'état du logement - Réparations majeures requises	Logement occupé selon période de construction - 1960 ou avant	Logement occupé selon période de construction - 1961 à 1980	Logement occupé selon période de construction - 1981 à 1990	Logement occupé selon période de construction - 1991 à 2000	Logement occupé selon période de construction - 2001 à 2005	Logement occupé selon période de construction - 2006 à 2010
0	2401023	Les Îles-de-la-Madeleine	3485.00	520.00	1280.00	1760.00	1070.00	525.00	215.00	305.00
1	2401042	Grosse-Île	105.00	20.00	85.00	70.00	30.00	20.00	15.00	0.00
2	2402005	Percé	1065.00	160.00	675.00	470.00	175.00	85.00	30.00	30.00
3	2402010	Sainte-Thérèse-de-Gaspé	300.00	55.00	135.00	160.00	100.00	15.00	20.00	25.00
4	2402015	Grande-Rivière	1010.00	160.00	515.00	595.00	225.00	90.00	50.00	60.00

```
print(df_final.columns)
```

Afficher la sortie masquée

```
len(all_vars)
```

```
42
```

```
len(df_final.columns)
```

```
44
```

```
# quelles sont les colonnes qu'il y a dans all_vars et pas dans df_final

set(all_vars) - set(df_final.columns)
```

```
set()
```

```
len(df_final.columns)
```

```
44
```

```
# combien de valeurs manquantes et non manquantes il y a dans chacune de mes colonnes de df_final

missing_counts = df_final.isnull().sum()
non_missing_counts = df_final.notnull().sum()

result_df = pd.DataFrame({
    'Missing Values': missing_counts,
    'Non-Missing Values': non_missing_counts,
    'Pourcentage de valeurs manquantes totales' : (missing_counts / len(df_final))*100
})

print(result_df)
```

Afficher la sortie masquée

```
# pourcentages de lignes qui ont 40 valeurs nulles, puis 39, puis 38, etc. jusqu'à 0

# Calculer le nombre de valeurs nulles par ligne
null_counts = df_final.isnull().sum(axis=1)

# Compter le nombre de lignes avec chaque nombre de valeurs nulles
null_counts_frequency = null_counts.value_counts().sort_index(ascending=False)

# Calculer le pourcentage de lignes pour chaque nombre de valeurs nulles
null_counts_percentage = (null_counts_frequency / len(df_final)) * 100

# Afficher les pourcentages
print(null_counts_percentage)
```

```
41    0.31
40    0.31
37    8.27
28    0.08
27    0.39
6     5.69
2     0.47
0    84.48
Name: count, dtype: float64
```

```
null_counts_frequency
```

	count
41	4
40	4
37	106
28	1
27	5
6	73
2	6
0	1083

dtype: int64

```
# Calcul des pourcentages pour chaque colonne de `rate_vars`

# 1. Type de construction résidentielle - Maison individuelle non attenante (%)
df_final["Type de construction résidentielle - Maison individuelle non attenante (%)"] = (
    df_final["Type de construction résidentielle - Maison individuelle non attenante"] /
    df_final["Total - Logements privés occupés par type de construction résidentielle - Données intégrales (100 %)"]
) * 100
```

```
del df_final["Type de construction résidentielle - Maison individuelle non attenante"]

# 2. Type de construction résidentielle - Maison jumelée (%)
df_final["Type de construction résidentielle - Maison jumelée (%)"] = (
    df_final["Type de construction résidentielle - Maison jumelée"] /
    df_final["Total - Logements privés occupés par type de construction résidentielle - Données intégrales (100 %)"]
) * 100

del df_final["Type de construction résidentielle - Maison jumelée"]

# 3. Type de construction résidentielle - Maison en rangée (%)
df_final["Type de construction résidentielle - Maison en rangée (%)"] = (
    df_final["Type de construction résidentielle - Maison en rangée"] /
    df_final["Total - Logements privés occupés par type de construction résidentielle - Données intégrales (100 %)"]
) * 100

del df_final["Type de construction résidentielle - Maison en rangée"]

# 4. Type de construction résidentielle - Appartement ou plain-pied dans un duplex (%)
df_final["Type de construction résidentielle - Appartement ou plain-pied dans un duplex (%)"] = (
    df_final["Type de construction résidentielle - Appartement ou plain-pied dans un duplex"] /
    df_final["Total - Logements privés occupés par type de construction résidentielle - Données intégrales (100 %)"]
) * 100

del df_final["Type de construction résidentielle - Appartement ou plain-pied dans un duplex"]

# 5. Type de construction résidentielle - Appartement dans un immeuble de moins de cinq étages (%)
df_final["Type de construction résidentielle - Appartement dans un immeuble de moins de cinq étages (%)"] = (
    df_final["Type de construction résidentielle - Appartement dans un immeuble de moins de cinq étages"] /
    df_final["Total - Logements privés occupés par type de construction résidentielle - Données intégrales (100 %)"]
) * 100

del df_final["Type de construction résidentielle - Appartement dans un immeuble de moins de cinq étages"]

# 6. Type de construction résidentielle - Appartement dans un immeuble de cinq étages ou plus (%)
df_final["Type de construction résidentielle - Appartement dans un immeuble de cinq étages ou plus (%)"] = (
    df_final["Type de construction résidentielle - Appartement dans un immeuble de cinq étages ou plus"] /
    df_final["Total - Logements privés occupés par type de construction résidentielle - Données intégrales (100 %)"]
) * 100

del df_final["Type de construction résidentielle - Appartement dans un immeuble de cinq étages ou plus"]
del df_final["Total - Logements privés occupés par type de construction résidentielle - Données intégrales (100 %)"]

# 7. Logement occupé selon période de construction - 1960 ou avant (%)
df_final["Logement occupé selon période de construction - 1960 ou avant (%)"] = (
    df_final["Logement occupé selon période de construction - 1960 ou avant"] /
    df_final["Total - Logements privés occupés selon la période de construction - Données-échantillon (25 %)"]
) * 100

del df_final["Logement occupé selon période de construction - 1960 ou avant"]

# 8. Logement occupé selon période de construction - 1961 à 1980 (%)
df_final["Logement occupé selon période de construction - 1961 à 1980 (%)"] = (
    df_final["Logement occupé selon période de construction - 1961 à 1980"] /
    df_final["Total - Logements privés occupés selon la période de construction - Données-échantillon (25 %)"]
) * 100

del df_final["Logement occupé selon période de construction - 1961 à 1980"]

# 9. Logement occupé selon période de construction - 1981 à 1990 (%)
df_final["Logement occupé selon période de construction - 1981 à 1990 (%)"] = (
    df_final["Logement occupé selon période de construction - 1981 à 1990"] /
    df_final["Total - Logements privés occupés selon la période de construction - Données-échantillon (25 %)"]
) * 100

del df_final["Logement occupé selon période de construction - 1981 à 1990"]

# 10. Logement occupé selon période de construction - 1991 à 2000 (%)
df_final["Logement occupé selon période de construction - 1991 à 2000 (%)"] = (
    df_final["Logement occupé selon période de construction - 1991 à 2000"] /
    df_final["Total - Logements privés occupés selon la période de construction - Données-échantillon (25 %)"]
) * 100

del df_final["Logement occupé selon période de construction - 1991 à 2000"]

# 11. Logement occupé selon période de construction - 2001 à 2005 (%)
df_final["Logement occupé selon période de construction - 2001 à 2005 (%)"] = (
    df_final["Logement occupé selon période de construction - 2001 à 2005"] /
    df_final["Total - Logements privés occupés selon la période de construction - Données-échantillon (25 %)"]
) * 100

del df_final["Logement occupé selon période de construction - 2001 à 2005"]
```

```
# 12. Logement occupé selon période de construction - 2006 à 2010 (%)
df_final["Logement occupé selon période de construction - 2006 à 2010 (%)"] = (
    df_final["Logement occupé selon période de construction - 2006 à 2010"] /
    df_final["Total - Logements privés occupés selon la période de construction - Données-échantillon (25 %)"]
) * 100

del df_final["Logement occupé selon période de construction - 2006 à 2010"]

# 13. Logement occupé selon période de construction - 2011
df_final["Logement occupé selon période de construction - 2011 à 2015 (%)"] = (
    df_final["Logement occupé selon période de construction - 2011 à 2015"] /
    df_final["Total - Logements privés occupés selon la période de construction - Données-échantillon (25 %)"]
) * 100

del df_final["Logement occupé selon période de construction - 2011 à 2015"]

# 14. Logement occupé selon période de construction - 2016 à 2021 (%)
df_final["Logement occupé selon période de construction - 2016 à 2021 (%)"] = (
    df_final["Logement occupé selon période de construction - 2016 à 2021"] /
    df_final["Total - Logements privés occupés selon la période de construction - Données-échantillon (25 %)"]
) * 100

del df_final["Logement occupé selon période de construction - 2016 à 2021"]
del df_final["Total - Logements privés occupés selon la période de construction - Données-échantillon (25 %)"]

# 15. Logement occupé selon l'état du logement - Réparations majeures requises (%)
df_final["Logement occupé selon l'état du logement - Réparations majeures requises (%)"] = (
    df_final["Logement occupé selon l'état du logement - Réparations majeures requises"] /
    df_final["Total - Logements privés occupés selon l'état du logement - Données-échantillon (25 %)"]
) * 100

del df_final["Logement occupé selon l'état du logement - Réparations majeures requises"]
del df_final["Total - Logements privés occupés selon l'état du logement - Données-échantillon (25 %)"]

# 16. Ménages privés selon le mode d'occupation - Propriétaire (%)
df_final["Ménages privés selon le mode d'occupation - Propriétaire (%)"] = (
    df_final["Ménages privés selon le mode d'occupation - Propriétaire"] /
    df_final["Total - Ménages privés selon le mode d'occupation - Données-échantillon (25 %)"]
) * 100

del df_final["Ménages privés selon le mode d'occupation - Propriétaire"]

# 17. Ménages privés selon le mode d'occupation - Locataire (%)
df_final["Ménages privés selon le mode d'occupation - Locataire (%)"] = (
    df_final["Ménages privés selon le mode d'occupation - Locataire"] /
    df_final["Total - Ménages privés selon le mode d'occupation - Données-échantillon (25 %)"]
) * 100

del df_final["Ménages privés selon le mode d'occupation - Locataire"]

# 18. Ménages privés selon le mode d'occupation - Logement fourni par le gouvernement local (%)
df_final["Ménages privés selon le mode d'occupation - Logement fourni par le gouvernement local (%)"] = (
    df_final["Ménages privés selon le mode d'occupation - Logement fourni par le gouvernement local"] /
    df_final["Total - Ménages privés selon le mode d'occupation - Données-échantillon (25 %)"]
) * 100

del df_final["Ménages privés selon le mode d'occupation - Logement fourni par le gouvernement local"]
del df_final["Total - Ménages privés selon le mode d'occupation - Données-échantillon (25 %)"]

# 19. Plus haut diplôme - Aucun certificat, diplôme ou grade (%)
df_final["Plus haut diplôme - Aucun certificat, diplôme ou grade (%)"] = (
    df_final["Plus haut diplôme - Aucun certificat, diplôme ou grade"] /
    df_final["Total - Plus haut certificat, diplôme ou grade pour la population âgée de 25 à 64 ans dans les ménages privés - Données-éc
) * 100

del df_final["Plus haut diplôme - Aucun certificat, diplôme ou grade"]

# 20. Plus haut diplôme - Diplôme d'études secondaires ou attestation d'équivalence (%)
df_final["Plus haut diplôme - Diplôme d'études secondaires ou attestation d'équivalence (%)"] = (
    df_final["Plus haut diplôme - Diplôme d'études secondaires ou attestation d'équivalence"] /
    df_final["Total - Plus haut certificat, diplôme ou grade pour la population âgée de 25 à 64 ans dans les ménages privés - Données-éc
) * 100

del df_final["Plus haut diplôme - Diplôme d'études secondaires ou attestation d'équivalence"]

# 21. Plus haut diplôme - Certificat, diplôme ou grade d’études postsecondaires (%)
df_final["Plus haut diplôme - Certificat, diplôme ou grade d’études postsecondaires (%)"] = (
    df_final["Plus haut diplôme - Certificat, diplôme ou grade d’études postsecondaires"] /
    df_final["Total - Plus haut certificat, diplôme ou grade pour la population âgée de 25 à 64 ans dans les ménages privés - Données-éc
) * 100
```

```
del df_final["Plus haut diplôme - Certificat, diplôme ou grade d'études postsecondaires"]
del df_final["Total - Plus haut certificat, diplôme ou grade pour la population âgée de 25 à 64 ans dans les ménages privés - Données-éc

# 22. Groupes d'âge de la population - 65 ans et plus (%)
df_final["Groupes d'âge de la population - 65 ans et plus (%)"] = (
    df_final["Groupes d'âge de la population - 65 ans et plus"] /
    df_final["Total - Groupes d'âge de la population - Données intégrales (100 %)"]
) * 100

del df_final["Groupes d'âge de la population - 65 ans et plus"]
del df_final["Total - Groupes d'âge de la population - Données intégrales (100 %)"]

# 23. Ménages dont 30 % ou plus du revenu est consacré aux frais de logement seulement (%)
df_final["Ménages dont 30 % ou plus du revenu est consacré aux frais de logement seulement (%)"] = (
    df_final["Ménages dont 30 % ou plus du revenu est consacré aux frais de logement seulement"] /
    df_final["Total - Ménages propriétaires et locataires dont le revenu total du ménage est supérieur à zéro, dans les logements privés
) * 100

del df_final["Ménages dont 30 % ou plus du revenu est consacré aux frais de logement seulement"]
del df_final["Total - Ménages propriétaires et locataires dont le revenu total du ménage est supérieur à zéro, dans les logements privés

# Affichage du DataFrame mis à jour
df_final.head()
```

	code_geo	nom_geo	Logements privés occupés par des résidents habituels	Nombre moyen de pièces par logement	Population, 2016	Population, 2021	Revenu total médian après impôts en 2020 parmi les bénéficiaires d'un revenu total ($)	Revenu total médian avant impôts en 2020 parmi les bénéficiaires d'un revenu total ($)	Revenu total médian des ménages en 2020($) après impôts	Revenu total médian des ménages en 2020($) avant impôts	Superficie des terres en kilomètres carrés	m
0	2401023	Les Îles-de-la-Madeleine	5610.00	6.20	12010.00	12190.00	35200.00	39600.00	65500.00	75000.00	155.06	
1	2401042	Grosse-Île	220.00	6.40	465.00	464.00	34400.00	39600.00	75000.00	86000.00	32.28	
2	2402005	Percé	1550.00	6.00	3103.00	3095.00	31000.00	34400.00	51200.00	56400.00	432.81	
3	2402010	Sainte-Thérèse-de-Gaspé	475.00	6.00	1015.00	979.00	33200.00	37200.00	58400.00	65000.00	34.45	
4	2402015	Grande-Rivière	1643.00	5.80	3408.00	3384.00	31400.00	35600.00	54000.00	60400.00	87.92	

```
# Affichage du DataFrame mis à jour
df_final.head(20)
```

	code_geo	nom_geo	Logements privés occupés par des résidents habituels	Nombre moyen de pièces par logement	Population, 2016	Population, 2021	Revenu total médian après impôts en 2020 parmi les bénéficiaires d'un revenu total ($)	Revenu total médian avant impôts en 2020 parmi les bénéficiaires d'un revenu total ($)	Revenu total médian des ménages en 2020($) après impôts	Revenu total médian des ménages en 2020($) avant impôts	Superfi des ter kilomèt car
0	2401023	Les Îles-de-la-Madeleine	5610.00	6.20	12010.00	12190.00	35200.00	39600.00	65500.00	75000.00	155
1	2401042	Grosse-Île	220.00	6.40	465.00	464.00	34400.00	39600.00	75000.00	86000.00	32
2	2402005	Percé	1550.00	6.00	3103.00	3095.00	31000.00	34400.00	51200.00	56400.00	432
3	2402010	Sainte-Thérèse-de-Gaspé	475.00	6.00	1015.00	979.00	33200.00	37200.00	58400.00	65000.00	34
4	2402015	Grande-Rivière	1643.00	5.80	3408.00	3384.00	31400.00	35600.00	54000.00	60400.00	87
5	2402028	Chandler	3572.00	5.90	7546.00	7490.00	31600.00	35200.00	54400.00	60000.00	418
6	2402047	Port-Daniel--Gascons	1079.00	6.20	2210.00	2271.00	30200.00	33200.00	53200.00	59200.00	300
7	2402902	Mont-Alexandre	0.00	NaN	0.00	0.00	NaN	NaN	NaN	NaN	1800
8	2403005	Gaspé	6780.00	6.30	14568.00	15063.00	36000.00	41200.00	63200.00	72000.00	1118
9	2403010	Cloridorme	315.00	6.50	671.00	607.00	29200.00	32000.00	50000.00	55200.00	158
10	2403015	Petite-Vallée	82.00	7.50	170.00	157.00	NaN	NaN	NaN	NaN	39
11	2403020	Grande-Vallée	530.00	6.10	1057.00	1077.00	31800.00	34800.00	55600.00	60400.00	144
12	2403025	Murdochville	327.00	6.70	651.00	643.00	33200.00	37200.00	52000.00	57600.00	60
13	2403902	Rivière-Saint-Jean	0.00	NaN	0.00	0.00	NaN	NaN	NaN	NaN	1748
14	2403904	Collines-du-Basque	0.00	NaN	0.00	0.00	NaN	NaN	NaN	NaN	817
15	2404005	Sainte-Madeleine-de-la-Rivière-Madeleine	161.00	6.50	289.00	289.00	23400.00	23800.00	39600.00	40800.00	262
16	2404010	Saint-Maxime-du-Mont-Louis	501.00	6.00	1134.00	1047.00	28200.00	30600.00	48400.00	53200.00	232
17	2404015	Mont-Saint-Pierre	98.00	6.30	155.00	186.00	NaN	NaN	NaN	NaN	52
18	2404020	Rivière-à-Claude	76.00	6.30	128.00	141.00	NaN	NaN	NaN	NaN	155
19	2404025	Marsoui	149.00	6.20	275.00	289.00	27400.00	28800.00	46000.00	50400.00	181

b.CENSUS 2016

```
# ------------------------------------------------------------------------------
# CENSUS DATA 2016 : importation
df2 = pd.read_csv(r'/content/gdrive/MyDrive/mémoire/Données/Census 2016/clean_stat_can_2016_clean_utf.csv',
                  encoding = "utf-8" )

#Renommer les colonnes pour  correpsondre à celles de 2021
df2 = df2.rename(columns={'CODE_GÉO_ALT':'code_geo', 'NIVEAU_GÉO' : 'type_geo', 'NOM_GÉO' : 'nom_geo', 'DIM: Profil des aires de diffusi
})
```

```
<ipython-input-35-ca695eb64185>:5: DtypeWarning: Columns (0) have mixed types. Specify dtype option on import or set low_memory=Fals
  df2 = pd.read_csv(r'/content/gdrive/MyDrive/mémoire/Données/Census 2016/clean_stat_can_2016_clean_utf.csv',
```

```
len(df2)
```

```
33801621
```

```
df2.head()
```

	nom_geo	code_geo	nom_car	total
0	Canada	1	Population, 2016	35151728
1	Canada	1	Population, 2011	33476688
2	Canada	1	Variation en pourcentage de la population, 201...	5.0
3	Canada	1	Total des logements privés	15412443
4	Canada	1	Logements privés occupés par des résidents hab...	14072079

```
##garder seulement les lignes où le code géographique est composé de 7 chiffres (les municipalités)

df2_filtered = df2[df2['code_geo'].astype(str).str.len() == 7]


df2_filtered.head(250)
```

Afficher la sortie masquée

```
# nom_géo qui ont différents code_géo

# Grouper les lignes par 'nom_geo' et agrège les valeurs uniques de 'code_geo' dans une liste
grouped = df2_filtered.groupby('nom_geo')['code_geo'].agg(set).reset_index()

# Filtrer les noms_geo avec plus d'un code_geo unique
multiple_codes = grouped[grouped['code_geo'].apply(len) > 1]

# Afficher le résultat
print(multiple_codes)
print(len(multiple_codes))
```

Afficher la sortie masquée

```
# Cette fonction stocke les changements réalisés
changes = []

def log_change(old_value, new_value):
    changes.append({'Ancien_nom_car': old_value, 'Nouveau_nom_car': new_value})

# Code modifié pour enregistrer les changements de nom_car

### Type de logement
start_indices = df2_filtered[df2_filtered["nom_car"] == "Total - Logements privés occupés par type de construction résidentielle - Donne
row_names = ["Type de construction résidentielle - Maison individuelle non attenante",
             "Type de construction résidentielle - Appartement dans un immeuble de cinq étages ou plus"
            ]

for index in start_indices:
    for i in range(2):
        old_value = df2_filtered.at[index + i + 1, "nom_car"]
        new_value = row_names[i]
        if old_value != new_value:  # To ensure that we only log actual changes
            df2_filtered.at[index + i + 1, "nom_car"] = new_value
            log_change(old_value, new_value)

start_indices = df2_filtered[df2_filtered["nom_car"] == "Total - Logements privés occupés par type de construction résidentielle - Donne
row_names = [ "Type de construction résidentielle - Maison jumelée",
              "Type de construction résidentielle - Maison en rangée",
              "Type de construction résidentielle - Appartement ou plain-pied dans un duplex",
              "Type de construction résidentielle - Appartement dans un immeuble de moins de cinq étages",
              ]
for index in start_indices:
    for i in range(4):
        old_value = df2_filtered.at[index + i + 4, "nom_car"]
        new_value = row_names[i]
        if old_value != new_value:  # To ensure that we only log actual changes
            df2_filtered.at[index + i + 4, "nom_car"] = new_value
            log_change(old_value, new_value)

### Revenu des ménages

### Revenu des ménages
start_indices = df2_filtered[df2_filtered["nom_car"] == "Total - Statistiques du revenu en 2015 pour les ménages privés selon la taille
row_names = ["Revenu total médian des ménages en 2015($) avant impôts",
             "Revenu total médian des ménages en 2015($) après impôts"]
```

```
for index in start_indices:
    for i in range(2):
        old_value = df2_filtered.at[index + i + 1, "nom_car"]
        new_value = row_names[i]
        if old_value != new_value:
            df2_filtered.at[index + i + 1, "nom_car"] = new_value
            log_change(old_value, new_value)

start_indices = df2_filtered[df2_filtered["nom_car"] == "Total - Statistiques du revenu en 2015 pour la population âgée de 15 ans et plu
row_names = ["Revenu total médian avant impôts en 2015 parmi les bénéficiaires d'un revenu total ($)"]

for index in start_indices:
    for i in range(1):
        old_value = df2_filtered.at[index + i + 2, "nom_car"]
        new_value = row_names[i]
        if old_value != new_value:
            df2_filtered.at[index + i + 2, "nom_car"] = new_value
            log_change(old_value, new_value)

start_indices = df2_filtered[df2_filtered["nom_car"] == "Total - Statistiques du revenu en 2015 pour la population âgée de 15 ans et plu
row_names = [
             "Revenu total médian après impôts en 2015 parmi les bénéficiaires d'un revenu total ($)"]

for index in start_indices:
    for i in range(1):
        old_value = df2_filtered.at[index + i + 4, "nom_car"]
        new_value = row_names[i]
        if old_value != new_value:
            df2_filtered.at[index + i + 4, "nom_car"] = new_value
            log_change(old_value, new_value)

### Période de construction
start_indices = df2_filtered[df2_filtered["nom_car"] == "Total - Logements privés occupés selon la période de construction - Données-éch
row_names = [

             "Logement occupé selon période de construction - 1960 ou avant",
             "Logement occupé selon période de construction - 1961 à 1980",
             "Logement occupé selon période de construction - 1981 à 1990",
             "Logement occupé selon période de construction - 1991 à 2000",
             "Logement occupé selon période de construction - 2001 à 2005",
             "Logement occupé selon période de construction - 2006 à 2010",
             "Logement occupé selon période de construction - 2011 à 2016"
             ]
for index in start_indices:
    for i in range(7):
        old_value = df2_filtered.at[index + i + 1, "nom_car"]
        new_value = row_names[i]
        if old_value != new_value:
            df2_filtered.at[index + i + 1, "nom_car"] = new_value
            log_change(old_value, new_value)

### Réparations majeures requises
start_indices = df2_filtered[df2_filtered["nom_car"] == "Total - Logements privés occupés selon l'état du logement - Données-échantillor
row_names = ["Logement occupé selon l'état du logement - Réparations majeures requises"
             ]
for index in start_indices:
    for i in range(1):
        old_value = df2_filtered.at[index + i + 2, "nom_car"]
        new_value = row_names[i]
        if old_value != new_value:
            df2_filtered.at[index + i + 2, "nom_car"] = new_value
            log_change(old_value, new_value)

### Mode d'occupation
start_indices = df2_filtered[df2_filtered["nom_car"] == "Total - Ménages privés selon le mode d'occupation - Données-échantillons (25 %)
row_names = ["Ménages privés selon le mode d'occupation - Propriétaire",
             "Ménages privés selon le mode d'occupation - Locataire",
             "Ménages privés selon le mode d'occupation - Logement fourni par le gouvernement local"
             ]
for index in start_indices:
    for i in range(3):
        old_value = df2_filtered.at[index + i + 1, "nom_car"]
        new_value = row_names[i]
        if old_value != new_value:
            df2_filtered.at[index + i + 1, "nom_car"] = new_value
            log_change(old_value, new_value)

### Education
start_indices = df2_filtered[df2_filtered["nom_car"] == "Total - Plus haut certificat, diplôme ou grade pour la population âgée de 25 à
row_names = ["Plus haut diplôme - Aucun certificat, diplôme ou grade",
             "Plus haut diplôme - Diplôme d'études secondaires ou attestation d'équivalence",
```

```
            "Plus haut diplôme - Certificat, diplôme ou grade d'études postsecondaires"
            ]
for index in start_indices:
    for i in range(3):
        old_value = df2_filtered.at[index + i + 1, "nom_car"]
        new_value = row_names[i]
        if old_value != new_value:
            df2_filtered.at[index + i + 1, "nom_car"] = new_value
            log_change(old_value, new_value)

### Groupes d'âge
start_indices = df2_filtered[df2_filtered["nom_car"] == "Total - Répartition (%) de la population par grands groupes d'âge - Données int
row_names = ["Groupes d'âge de la population - 65 ans et plus"]

for index in start_indices:
    for i in range(1):
        old_value = df2_filtered.at[index + i + 3, "nom_car"]
        new_value = row_names[i]
        if old_value != new_value:
            df2_filtered.at[index + i + 3, "nom_car"] = new_value
            log_change(old_value, new_value)

### Frais de logements
start_indices = df2_filtered[df2_filtered["nom_car"] == "Total - Ménages propriétaires et locataires dont le revenu total du ménage est
row_names = ["Ménages dont 30 % ou plus du revenu est consacré aux frais de logement seulement"]

for index in start_indices:
    for i in range(1):
        old_value = df2_filtered.at[index + i + 2, "nom_car"]
        new_value = row_names[i]
        if old_value != new_value:
            df2_filtered.at[index + i + 2, "nom_car"] = new_value
            log_change(old_value, new_value)

# Crée une DataFrame pour les changements et afficher les valeurs uniques
change_df = pd.DataFrame(changes)
unique_changes = change_df.drop_duplicates()
print(unique_changes)
```

```
                                        Ancien_nom_car  \
0                       Maison individuelle non attenante
1      Appartement dans un immeuble de cinq étages ou...
2570                                      Maison jumelée
2571                                    Maison en rangée
2572           Appartement ou plain-pied dans un duplex
2573   Appartement dans un immeuble de moins de cinq ...
7710         Revenu total médian des ménages en 2015 ($)
7711   Revenu après impôt médian des ménages en 2015 ($)
10280  Revenu total médian en 2015 parmi les bénéfici...
11565  Revenu après impôt médian en 2015 parmi les bé...
12850                                     1960 ou avant
12851                                       1961 à 1980
12852                                       1981 à 1990
12853                                       1991 à 2000
12854                                       2001 à 2005
12855                                       2006 à 2010
12856                                       2011 à 2016
21845                      Réparations majeures requises
23130                                     Propriétaire
23131                                        Locataire
23132                                  Logement de bande
26985                 Aucun certificat, diplôme ou grade
26986  Diplôme d'études secondaires ou attestation d'...
26987  Certificat, diplôme ou grade d'études postseco...
30840                                     65 ans et plus
32125  30 % ou plus du revenu est consacré aux frais ...

                                       Nouveau_nom_car
0      Type de construction résidentielle - Maison in...
1      Type de construction résidentielle - Apparteme...
2570   Type de construction résidentielle - Maison ju...
2571   Type de construction résidentielle - Maison en...
2572   Type de construction résidentielle - Apparteme...
2573   Type de construction résidentielle - Apparteme...
7710   Revenu total médian des ménages en 2015($) ava...
7711   Revenu total médian des ménages en 2015($) apr...
10280  Revenu total médian avant impôts en 2015 parmi...
11565  Revenu total médian après impôts en 2015 parmi...
12850  Logement occupé selon période de construction ...
12851  Logement occupé selon période de construction ...
12852  Logement occupé selon période de construction ...
12853  Logement occupé selon période de construction ...
12854  Logement occupé selon période de construction ...
12855  Logement occupé selon période de construction ...
12856  Logement occupé selon période de construction ...
21845  Logement occupé selon l'état du logement - Rép...
```

```
23130  Ménages privés selon le mode d'occupation - Pr...
23131  Ménages privés selon le mode d'occupation - Lo...
23132  Ménages privés selon le mode d'occupation - Lo...
26985  Plus haut diplôme - Aucun certificat, diplôme ...
26986  Plus haut diplôme - Diplôme d'études secondair...
26987  Plus haut diplôme - Certificat, diplôme ou gra...
30840    Groupes d'âge de la population - 65 ans et plus
32125  Ménages dont 30 % ou plus du revenu est consac...
```

```
vars = [
    "Revenu total médian après impôts en 2015 parmi les bénéficiaires d'un revenu total ($)",
    "Revenu total médian avant impôts en 2015 parmi les bénéficiaires d'un revenu total ($)",
    "Population, 2011",
    "Population, 2016",
    "Superficie des terres en kilomètres carrés",
    "Taille moyenne des ménages privés",
    "Nombre moyen de pièces par logement",
    "Logements privés occupés par des résidents habituels",
    "Total des logements privés",
    "Taux de chômage",
    "Revenu total médian des ménages en 2015($) avant impôts",
    "Revenu total médian des ménages en 2015($) après impôts",

    #variables pour calculer mes taux

    "Total - Logements privés occupés par type de construction résidentielle - Données intégrales (100 %)",

    "Total - Logements privés occupés selon la période de construction - Données-échantillons (25 %)",

    "Total - Logements privés occupés selon l'état du logement - Données-échantillons (25 %)",

    "Total - Ménages privés selon le mode d'occupation - Données-échantillons (25 %)",

    "Total - Plus haut certificat, diplôme ou grade pour la population âgée de 25 à 64 ans dans les ménages privés - Données-échantillor

    "Total - Groupes d'âge et âge moyen de la population - Données intégrales (100 %)",

    "Total - Ménages propriétaires et locataires dont le revenu total du ménage est supérieur à zéro, dans les logements privés non agri

]

rate_vars = rate_vars = [
    "Type de construction résidentielle - Maison individuelle non attenante",
    "Type de construction résidentielle - Maison jumelée",
    "Type de construction résidentielle - Maison en rangée",
    "Type de construction résidentielle - Appartement ou plain-pied dans un duplex",
    "Type de construction résidentielle - Appartement dans un immeuble de moins de cinq étages",
    "Type de construction résidentielle - Appartement dans un immeuble de cinq étages ou plus",

    "Logement occupé selon période de construction - 1960 ou avant",
    "Logement occupé selon période de construction - 1961 à 1980",
    "Logement occupé selon période de construction - 1981 à 1990",
    "Logement occupé selon période de construction - 1991 à 2000",
    "Logement occupé selon période de construction - 2001 à 2005",
    "Logement occupé selon période de construction - 2006 à 2010",
    "Logement occupé selon période de construction - 2011 à 2016",

    "Logement occupé selon l'état du logement - Réparations majeures requises",

    "Ménages privés selon le mode d'occupation - Propriétaire",
    "Ménages privés selon le mode d'occupation - Locataire",
    "Ménages privés selon le mode d'occupation - Logement fourni par le gouvernement local",

    "Plus haut diplôme - Aucun certificat, diplôme ou grade",
    "Plus haut diplôme - Diplôme d'études secondaires ou attestation d'équivalence",
    "Plus haut diplôme - Certificat, diplôme ou grade d’études postsecondaires",

    "Groupes d'âge de la population - 65 ans et plus",

    "Ménages dont 30 % ou plus du revenu est consacré aux frais de logement seulement"

]
```

```
# garder seulement les lignes ou nom_car = var + rate_var

all_vars = vars + rate_vars
df2_filtered = df2_filtered[df2_filtered['nom_car'].isin(all_vars)]
```

```
df2_filtered['total'] = pd.to_numeric(df2_filtered['total'], errors='coerce')

# Créer une liste de toutes les municipalités uniques avec leur code géographique
unique_municipalities = df2_filtered[['code_geo', 'nom_geo']].drop_duplicates()

# Transformer le DataFrame pour le fusionner avec les variables cibles
# Ici, nous utilisons le pivot pour réorganiser les données
df2_pivoted = df2_filtered.pivot_table(index=['code_geo', 'nom_geo'],
                                       columns='nom_car',
                                       values='total',
                                       fill_value=np.nan).reset_index()

# Maintenant, fusionner les municipalités uniques avec le DataFrame pivoté
df2_final = unique_municipalities.merge(df2_pivoted, on=['code_geo', 'nom_geo'], how='left')
```

```
<ipython-input-44-a28ebde5e847>:2: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus
  df2_filtered['total'] = pd.to_numeric(df2_filtered['total'], errors='coerce')
```

```
df2_final.head()
```

	code_geo	nom_geo	Groupes d'âge de la population - 65 ans et plus	Logement occupé selon l'état du logement - Réparations majeures requises	Logement occupé selon période de construction - 1960 ou avant	Logement occupé selon période de construction - 1961 à 1980	Logement occupé selon période de construction - 1981 à 1990	Logement occupé selon période de construction - 1991 à 2000	Logement occupé selon période de construction - 2001 à 2005	Logement occupé selon période de construction - 2006 à 2010
0	2401023	Les Îles-de-la-Madeleine	24.40	355.00	1285.00	1830.00	965.00	485.00	220.00	310.00
1	2401042	Grosse-Île	18.30	40.00	75.00	95.00	20.00	15.00	10.00	0.00
2	2402005	Percé	28.30	210.00	650.00	455.00	170.00	135.00	40.00	45.00
3	2402010	Sainte-Thérèse-de-Gaspé	23.60	45.00	135.00	210.00	60.00	20.00	10.00	20.00
4	2402015	Grande-Rivière	23.60	145.00	500.00	620.00	220.00	105.00	35.00	75.00

```
# je me demande combien de valeurs manquantes et non manquantes il y a dans chacune de mes colonnes

missing_counts = df2_filtered.isnull().sum()
non_missing_counts = df2_filtered.notnull().sum()

result_df = pd.DataFrame({
    'Missing Values': missing_counts,
    'Non-Missing Values': non_missing_counts
})

print(result_df)
```

```
           Missing Values  Non-Missing Values
nom_geo                 0               52685
code_geo                0               52685
nom_car                 0               52685
total                4490               48195
```

```
df2_final.head()
```

	code_geo	nom_geo	Groupes d'âge de la population - 65 ans et plus	Logement occupé selon l'état du logement - Réparations majeures requises	Logement occupé selon période de construction - 1960 ou avant	Logement occupé selon période de construction - 1961 à 1980	Logement occupé selon période de construction - 1981 à 1990	Logement occupé selon période de construction - 1991 à 2000	Logement occupé selon période de construction - 2001 à 2005	Logement occupé selon période de construction - 2006 à 2010
0	2401023	Les Îles-de-la-Madeleine	24.40	355.00	1285.00	1830.00	965.00	485.00	220.00	310.00
1	2401042	Grosse-Île	18.30	40.00	75.00	95.00	20.00	15.00	10.00	0.00
2	2402005	Percé	28.30	210.00	650.00	455.00	170.00	135.00	40.00	45.00
3	2402010	Sainte-Thérèse-de-Gaspé	23.60	45.00	135.00	210.00	60.00	20.00	10.00	20.00
4	2402015	Grande-Rivière	23.60	145.00	500.00	620.00	220.00	105.00	35.00	75.00

```
weird_var = set(all_vars) - set(df2_filtered['nom_car'].unique())

weird_var
```

```
set()
```

```
len(df2_final.columns)
```

```
43
```

```
# tableau de fréquence des valeurs manquantes

null_counts = df2_final.isnull().sum(axis=1)
null_counts_frequency = null_counts.value_counts().sort_index(ascending=False)
null_counts_percentage = (null_counts_frequency / len(df2_final)) * 100
print(null_counts_percentage)
```

```
40    0.31
36    8.48
26    0.08
24    0.16
16    0.08
6     0.08
4     5.76
2     0.47
1     0.16
0    84.44
Name: count, dtype: float64
```

```
# Calcul des pourcentages pour chaque colonne de `rate_vars`

# 1. Type de construction résidentielle - Maison individuelle non attenante (%)
df2_final["Type de construction résidentielle - Maison individuelle non attenante (%)"] = (
    df2_final["Type de construction résidentielle - Maison individuelle non attenante"] /
    df2_final["Total - Logements privés occupés par type de construction résidentielle - Données intégrales (100 %)"]
) * 100

del df2_final["Type de construction résidentielle - Maison individuelle non attenante"]

# 2. Type de construction résidentielle - Maison jumelée (%)
df2_final["Type de construction résidentielle - Maison jumelée (%)"] = (
    df2_final["Type de construction résidentielle - Maison jumelée"] /
    df2_final["Total - Logements privés occupés par type de construction résidentielle - Données intégrales (100 %)"]
) * 100

del df2_final["Type de construction résidentielle - Maison jumelée"]

# 3. Type de construction résidentielle - Maison en rangée (%)
df2_final["Type de construction résidentielle - Maison en rangée (%)"] = (
    df2_final["Type de construction résidentielle - Maison en rangée"] /
```

```
    df2_final["Total - Logements privés occupés par type de construction résidentielle - Données intégrales (100 %)"]
) * 100

del df2_final["Type de construction résidentielle - Maison en rangée"]

# 4. Type de construction résidentielle - Appartement ou plain-pied dans un duplex (%)
df2_final["Type de construction résidentielle - Appartement ou plain-pied dans un duplex (%)"] = (
    df2_final["Type de construction résidentielle - Appartement ou plain-pied dans un duplex"] /
    df2_final["Total - Logements privés occupés par type de construction résidentielle - Données intégrales (100 %)"]
) * 100

del df2_final["Type de construction résidentielle - Appartement ou plain-pied dans un duplex"]

# 5. Type de construction résidentielle - Appartement dans un immeuble de moins de cinq étages (%)
df2_final["Type de construction résidentielle - Appartement dans un immeuble de moins de cinq étages (%)"] = (
    df2_final["Type de construction résidentielle - Appartement dans un immeuble de moins de cinq étages"] /
    df2_final["Total - Logements privés occupés par type de construction résidentielle - Données intégrales (100 %)"]
) * 100

del df2_final["Type de construction résidentielle - Appartement dans un immeuble de moins de cinq étages"]

# 6. Type de construction résidentielle - Appartement dans un immeuble de cinq étages ou plus (%)
df2_final["Type de construction résidentielle - Appartement dans un immeuble de cinq étages ou plus (%)"] = (
    df2_final["Type de construction résidentielle - Appartement dans un immeuble de cinq étages ou plus"] /
    df2_final["Total - Logements privés occupés par type de construction résidentielle - Données intégrales (100 %)"]
) * 100

del df2_final["Type de construction résidentielle - Appartement dans un immeuble de cinq étages ou plus"]
del df2_final["Total - Logements privés occupés par type de construction résidentielle - Données intégrales (100 %)"]

# 7. Logement occupé selon période de construction - 1960 ou avant (%)
df2_final["Logement occupé selon période de construction - 1960 ou avant (%)"] = (
    df2_final["Logement occupé selon période de construction - 1960 ou avant"] /
    df2_final["Total - Logements privés occupés selon la période de construction - Données-échantillons (25 %)"]
) * 100

del df2_final["Logement occupé selon période de construction - 1960 ou avant"]

# 8. Logement occupé selon période de construction - 1961 à 1980 (%)
df2_final["Logement occupé selon période de construction - 1961 à 1980 (%)"] = (
    df2_final["Logement occupé selon période de construction - 1961 à 1980"] /
    df2_final["Total - Logements privés occupés selon la période de construction - Données-échantillons (25 %)"]
) * 100

del df2_final["Logement occupé selon période de construction - 1961 à 1980"]

# 9. Logement occupé selon période de construction - 1981 à 1990 (%)
df2_final["Logement occupé selon période de construction - 1981 à 1990 (%)"] = (
    df2_final["Logement occupé selon période de construction - 1981 à 1990"] /
    df2_final["Total - Logements privés occupés selon la période de construction - Données-échantillons (25 %)"]
) * 100

del df2_final["Logement occupé selon période de construction - 1981 à 1990"]

# 10. Logement occupé selon période de construction - 1991 à 2000 (%)
df2_final["Logement occupé selon période de construction - 1991 à 2000 (%)"] = (
    df2_final["Logement occupé selon période de construction - 1991 à 2000"] /
    df2_final["Total - Logements privés occupés selon la période de construction - Données-échantillons (25 %)"]
) * 100


del df2_final["Logement occupé selon période de construction - 1991 à 2000"]

# 11. Logement occupé selon période de construction - 2001 à 2005 (%)
df2_final["Logement occupé selon période de construction - 2001 à 2005 (%)"] = (
    df2_final["Logement occupé selon période de construction - 2001 à 2005"] /
    df2_final["Total - Logements privés occupés selon la période de construction - Données-échantillons (25 %)"]
) * 100

del df2_final["Logement occupé selon période de construction - 2001 à 2005"]

# 12. Logement occupé selon période de construction - 2006 à 2010 (%)
df2_final["Logement occupé selon période de construction - 2006 à 2010 (%)"] = (
    df2_final["Logement occupé selon période de construction - 2006 à 2010"] /
    df2_final["Total - Logements privés occupés selon la période de construction - Données-échantillons (25 %)"]
) * 100

del df2_final["Logement occupé selon période de construction - 2006 à 2010"]

# 13. Logement occupé selon période de construction - 2011 --> je choisis par souci de nomenclature de passer cette colonne à entre 2011
df2_final["Logement occupé selon période de construction - 2011 à 2015 (%)"] = (
    df2_final["Logement occupé selon période de construction - 2011 à 2016"] /
```

```
    df2_final["Total - Logements privés occupés selon la période de construction - Données-échantillons (25 %)"]
) * 100

del df2_final["Logement occupé selon période de construction - 2011 à 2016"]
del df2_final["Total - Logements privés occupés selon la période de construction - Données-échantillons (25 %)"]

# 15. Logement occupé selon l'état du logement - Réparations majeures requises (%)
df2_final["Logement occupé selon l'état du logement - Réparations majeures requises (%)"] = (
    df2_final["Logement occupé selon l'état du logement - Réparations majeures requises"] /
    df2_final["Total - Logements privés occupés selon l'état du logement - Données-échantillons (25 %)"]
) * 100

del df2_final["Logement occupé selon l'état du logement - Réparations majeures requises"]
del df2_final["Total - Logements privés occupés selon l'état du logement - Données-échantillons (25 %)"]

# 16. Ménages privés selon le mode d'occupation - Propriétaire (%)
df2_final["Ménages privés selon le mode d'occupation - Propriétaire (%)"] = (
    df2_final["Ménages privés selon le mode d'occupation - Propriétaire"] /
    df2_final["Total - Ménages privés selon le mode d'occupation - Données-échantillons (25 %)"]
) * 100

del df2_final["Ménages privés selon le mode d'occupation - Propriétaire"]

# 17. Ménages privés selon le mode d'occupation - Locataire (%)
df2_final["Ménages privés selon le mode d'occupation - Locataire (%)"] = (
    df2_final["Ménages privés selon le mode d'occupation - Locataire"] /
    df2_final["Total - Ménages privés selon le mode d'occupation - Données-échantillons (25 %)"]
) * 100

del df2_final["Ménages privés selon le mode d'occupation - Locataire"]

# 18. Ménages privés selon le mode d'occupation - Logement fourni par le gouvernement local (%)
df2_final["Ménages privés selon le mode d'occupation - Logement fourni par le gouvernement local (%)"] = (
    df2_final["Ménages privés selon le mode d'occupation - Logement fourni par le gouvernement local"] /
    df2_final["Total - Ménages privés selon le mode d'occupation - Données-échantillons (25 %)"]
) * 100

del df2_final["Ménages privés selon le mode d'occupation - Logement fourni par le gouvernement local"]
del df2_final["Total - Ménages privés selon le mode d'occupation - Données-échantillons (25 %)"]

# 19. Plus haut diplôme - Aucun certificat, diplôme ou grade (%)
df2_final["Plus haut diplôme - Aucun certificat, diplôme ou grade (%)"] = (
    df2_final["Plus haut diplôme - Aucun certificat, diplôme ou grade"] /
    df2_final["Total - Plus haut certificat, diplôme ou grade pour la population âgée de 25 à 64 ans dans les ménages privés - Données-é
) * 100

del df2_final["Plus haut diplôme - Aucun certificat, diplôme ou grade"]

# 20. Plus haut diplôme - Diplôme d'études secondaires ou attestation d'équivalence (%)
df2_final["Plus haut diplôme - Diplôme d'études secondaires ou attestation d'équivalence (%)"] = (
    df2_final["Plus haut diplôme - Diplôme d'études secondaires ou attestation d'équivalence"] /
    df2_final["Total - Plus haut certificat, diplôme ou grade pour la population âgée de 25 à 64 ans dans les ménages privés - Données-é
) * 100

del df2_final["Plus haut diplôme - Diplôme d'études secondaires ou attestation d'équivalence"]

# 21. Plus haut diplôme - Certificat, diplôme ou grade d’études postsecondaires (%)
df2_final["Plus haut diplôme - Certificat, diplôme ou grade d’études postsecondaires (%)"] = (
    df2_final["Plus haut diplôme - Certificat, diplôme ou grade d’études postsecondaires"] /
    df2_final["Total - Plus haut certificat, diplôme ou grade pour la population âgée de 25 à 64 ans dans les ménages privés - Données-é
) * 100

del df2_final["Plus haut diplôme - Certificat, diplôme ou grade d’études postsecondaires"]
del df2_final["Total - Plus haut certificat, diplôme ou grade pour la population âgée de 25 à 64 ans dans les ménages privés - Données-é

# 22. Groupes d'âge de la population - 65 ans et plus (%)
df2_final["Groupes d'âge de la population - 65 ans et plus (%)"] = (
    df2_final["Groupes d'âge de la population - 65 ans et plus"] /
    df2_final["Total - Groupes d'âge et âge moyen de la population - Données intégrales (100 %)"]
) * 100

del df2_final["Groupes d'âge de la population - 65 ans et plus"]
del df2_final["Total - Groupes d'âge et âge moyen de la population - Données intégrales (100 %)"]

# 23. Ménages dont 30 % ou plus du revenu est consacré aux frais de logement seulement (%)
df2_final["Ménages dont 30 % ou plus du revenu est consacré aux frais de logement seulement (%)"] = (
    df2_final["Ménages dont 30 % ou plus du revenu est consacré aux frais de logement seulement"] /
    df2_final["Total - Ménages propriétaires et locataires dont le revenu total du ménage est supérieur à zéro, dans les logements prive
) * 100

del df2_final["Ménages dont 30 % ou plus du revenu est consacré aux frais de logement seulement"]
del df2_final["Total - Ménages propriétaires et locataires dont le revenu total du ménage est supérieur à zéro, dans les logements prive
```

```
# Affichage du DataFrame mis à jour
df2_final.head()
```

	code_geo	nom_geo	Logements privés occupés par des résidents habituels	Nombre moyen de pièces par logement	Population, 2011	Population, 2016	Revenu total médian après impôts en 2015 parmi les bénéficiaires d'un revenu total ($)	Revenu total médian avant impôts en 2015 parmi les bénéficiaires d'un revenu total ($)	Revenu total médian des ménages en 2015($) après impôts	Revenu total médian des ménages en 2015($) avant impôts	Superficie des terres en kilomètres carrés	m
0	2401023	Les Îles-de-la-Madeleine	5321.00	6.20	12291.00	12010.00	28023.00	31541.00	53794.00	61029.00	172.71	
1	2401042	Grosse-Île	213.00	6.30	490.00	465.00	29024.00	33306.00	68352.00	81152.00	37.59	
2	2402005	Percé	1523.00	6.10	3312.00	3103.00	24073.00	26027.00	40804.00	44826.00	431.37	
3	2402010	Sainte-Thérèse-de-Gaspé	462.00	6.10	1055.00	1015.00	25472.00	27776.00	48640.00	54144.00	34.41	
4	2402015	Grande-Rivière	1610.00	5.70	3456.00	3408.00	23086.00	24709.00	41956.00	45751.00	87.86	

```
len(df2_final['nom_geo'].unique())
```

```
1226
```

c. Merge census 2021 et 2016

```
# noms des colonnes qui sont dans df_final et pas df2_final et inversement

# Colonnes dans df_final mais pas dans df2_final
cols_in_df_final_not_in_df2_final = set(df_final.columns) - set(df2_final.columns)
print("Colonnes dans df_final mais pas dans df2_final:\n", cols_in_df_final_not_in_df2_final)

# Colonnes dans df2_final mais pas dans df_final
cols_in_df2_final_not_in_df_final = set(df2_final.columns) - set(df_final.columns)
print("\nColonnes dans df2_final mais pas dans df_final:\n", cols_in_df2_final_not_in_df_final)
```

```
Colonnes dans df_final mais pas dans df2_final:
 {'Population, 2021', 'Logement occupé selon période de construction - 2016 à 2021 (%)', "Revenu total médian après impôts en 2020 p

Colonnes dans df2_final mais pas dans df_final:
 {'Population, 2011', "Revenu total médian après impôts en 2015 parmi les bénéficiaires d'un revenu total ($)", 'Revenu total médian
```

```
# outer merge df_final et df2_final sur code_géo/nom_géo en faisant la moyenne des colonnes si la même est présente dans les deux df, en

merged_df = pd.merge(df_final, df2_final, on=['code_geo', 'nom_geo'], how='outer', suffixes=('_2021', '_2016'))

#gérerr les potentielles valeurs dupliquées et les valeurs manquantes
for col in merged_df.columns:
    if col.endswith('_2021') and col[:-5] + '_2016' in merged_df.columns:
        # Calculate the mean if both columns have non-null values
        merged_df[col[:-5]] = merged_df[[col, col[:-5] + '_2016']].mean(axis=1, skipna=True)

        # Handle cases where one column is null and the other is not
        merged_df[col[:-5]].fillna(merged_df[col], inplace=True)
        merged_df[col[:-5]].fillna(merged_df[col[:-5] + '_2016'], inplace=True)

        # Drop the original columns
        merged_df.drop([col, col[:-5] + '_2016'], axis=1, inplace=True)
    elif col.endswith('_2021'):
        # Rename columns from 2021 DataFrame to remove suffix
        merged_df.rename(columns={col: col[:-5]}, inplace=True)
    elif col.endswith('_2016'):
        # Rename columns from 2016 DataFrame to remove suffix
        merged_df.rename(columns={col: col[:-5]}, inplace=True)
```

```
merged_df.head()
```

Afficher la sortie masquée

```
merged_df.columns
```

```
Index(['code_geo', 'nom_geo', 'Population, 2021',
       'Revenu total médian après impôts en 2020 parmi les bénéficiaires d'un revenu total ($)',
       'Revenu total médian avant impôts en 2020 parmi les bénéficiaires d'un revenu total ($)',
       'Revenu total médian des ménages en 2020($) après impôts',
       'Revenu total médian des ménages en 2020($) avant impôts',
       'Logement occupé selon période de construction - 2016 à 2021 (%)',
       'Population, 2011',
       'Revenu total médian après impôts en 2015 parmi les bénéficiaires d'un revenu total ($)',
       'Revenu total médian avant impôts en 2015 parmi les bénéficiaires d'un revenu total ($)',
       'Revenu total médian des ménages en 2015($) après impôts',
       'Revenu total médian des ménages en 2015($) avant impôts',
       'Logements privés occupés par des résidents habituels',
       'Nombre moyen de pièces par logement', 'Population, 2016',
       'Superficie des terres en kilomètres carrés',
       'Taille moyenne des ménages privés', 'Taux de chômage',
       'Total des logements privés',
       'Type de construction résidentielle - Maison individuelle non attenante (%)',
       'Type de construction résidentielle - Maison jumelée (%)',
       'Type de construction résidentielle - Maison en rangée (%)',
       'Type de construction résidentielle - Appartement ou plain-pied dans un duplex (%)',
       'Type de construction résidentielle - Appartement dans un immeuble de moins de cinq étages (%)',
       'Type de construction résidentielle - Appartement dans un immeuble de cinq étages ou plus (%)',
       'Logement occupé selon période de construction - 1960 ou avant (%)',
       'Logement occupé selon période de construction - 1961 à 1980 (%)',
       'Logement occupé selon période de construction - 1981 à 1990 (%)',
       'Logement occupé selon période de construction - 1991 à 2000 (%)',
       'Logement occupé selon période de construction - 2001 à 2005 (%)',
       'Logement occupé selon période de construction - 2006 à 2010 (%)',
       'Logement occupé selon période de construction - 2011 à 2015 (%)',
       'Logement occupé selon l'état du logement - Réparations majeures requises (%)',
       'Ménages privés selon le mode d'occupation - Propriétaire (%)',
       'Ménages privés selon le mode d'occupation - Locataire (%)',
       'Ménages privés selon le mode d'occupation - Logement fourni par le gouvernement local (%)',
       'Plus haut diplôme - Aucun certificat, diplôme ou grade (%)',
       'Plus haut diplôme - Diplôme d'études secondaires ou attestation d'équivalence (%)',
       'Plus haut diplôme - Certificat, diplôme ou grade d’études postsecondaires (%)',
       'Groupes d'âge de la population - 65 ans et plus (%)',
       'Ménages dont 30 % ou plus du revenu est consacré aux frais de logement seulement (%)'],
      dtype='object')
```

```
# nouvelle colonne "Revenu médian moyen après impôt" qui est la moyenne des colonnes "Revenu total médian après impôts en 2020 parmi les

merged_df['Revenu médian moyen après impôts'] = merged_df[
    ['Revenu total médian après impôts en 2020 parmi les bénéficiaires d\'un revenu total ($)',
     'Revenu total médian après impôts en 2015 parmi les bénéficiaires d\'un revenu total ($)']
].mean(axis=1)

merged_df['Revenu médian moyen avant impôts'] = merged_df[
    ['Revenu total médian avant impôts en 2020 parmi les bénéficiaires d\'un revenu total ($)',
     'Revenu total médian avant impôts en 2015 parmi les bénéficiaires d\'un revenu total ($)']
].mean(axis=1)

del merged_df['Revenu total médian après impôts en 2020 parmi les bénéficiaires d\'un revenu total ($)']
del merged_df['Revenu total médian après impôts en 2015 parmi les bénéficiaires d\'un revenu total ($)']
del merged_df['Revenu total médian avant impôts en 2020 parmi les bénéficiaires d\'un revenu total ($)']
del merged_df['Revenu total médian avant impôts en 2015 parmi les bénéficiaires d\'un revenu total ($)']

merged_df['Revenu médian des ménages avant impôts'] = merged_df[['Revenu total médian des ménages en 2015($) avant impôts',
                                                                 'Revenu total médian des ménages en 2020($) avant impôts']].mean(axis=1)
merged_df['Revenu médian des ménages après impôts'] = merged_df[['Revenu total médian des ménages en 2015($) après impôts',
                                                                 'Revenu total médian des ménages en 2020($) après impôts']].mean(axis=1)

del merged_df['Revenu total médian des ménages en 2015($) avant impôts']
del merged_df['Revenu total médian des ménages en 2020($) avant impôts']
del merged_df['Revenu total médian des ménages en 2015($) après impôts']
del merged_df['Revenu total médian des ménages en 2020($) après impôts']
```

```
merged_df.head(100)
```

	code_geo	nom_geo	Population, 2021	Logement occupé selon période de construction - 2016 à 2021 (%)	Population, 2011	Logements privés occupés par des résidents habituels	Nombre moyen de pièces par logement	Population, 2016	Superficie des terres en kilomètres carrés	Taille moyenne des ménages privés	Taux de chômage
0	2401023	Les Îles-de-la-Madeleine	12190.00	3.58	12291.00	5465.50	6.20	12010.00	163.88	2.15	11.95
1	2401042	Grosse-Île	464.00	8.00	490.00	216.50	6.35	465.00	34.94	2.15	14.90
2	2402005	Percé	3095.00	2.58	3312.00	1536.50	6.05	3103.00	432.09	2.00	20.40
3	2402010	Sainte-Thérèse-de-Gaspé	979.00	2.11	1055.00	468.50	6.05	1015.00	34.43	2.15	9.25
4	2402015	Grande-Rivière	3384.00	2.44	3456.00	1626.50	5.75	3408.00	87.89	2.10	12.50
5	2402028	Chandler	7490.00	0.84	7703.00	3508.50	5.95	7546.00	418.89	2.10	10.50
6	2402047	Port-Daniel--Gascons	2271.00	3.24	2453.00	1030.50	6.25	2210.00	300.95	2.15	13.85
7	2402902	Mont-Alexandre	0.00	NaN	0.00	0.00	NaN	0.00	1801.55	NaN	NaN
8	2403005	Gaspé	15063.00	3.91	15163.00	6596.00	6.30	14568.00	1120.10	2.20	10.35
9	2403010	Cloridorme	607.00	0.00	743.00	325.50	6.55	671.00	159.18	1.95	13.45
10	2403015	Petite-Vallée	157.00	0.00	178.00	82.50	7.70	170.00	39.92	2.00	0.00
11	2403020	Grande-Vallée	1077.00	3.77	1137.00	511.50	6.15	1057.00	144.33	2.05	15.05
12	2403025	Murdochville	643.00	0.00	764.00	320.00	6.80	651.00	61.11	2.00	15.55
13	2403902	Rivière-Saint-Jean	0.00	NaN	0.00	0.00	NaN	0.00	1749.60	NaN	NaN
14	2403904	Collines-du-Basque	0.00	NaN	0.00	0.00	NaN	0.00	819.36	NaN	NaN
15	2404005	Sainte-Madeleine-de-la-Rivière-Madeleine	289.00	0.00	334.00	155.50	6.45	289.00	262.85	1.85	31.00
16	2404010	Saint-Maxime-du-Mont-Louis	1047.00	0.00	1118.00	498.50	6.05	1134.00	233.15	2.10	19.30
17	2404015	Mont-Saint-Pierre	186.00	0.00	192.00	92.50	6.65	155.00	52.23	1.80	26.75
18	2404020	Rivière-à-Claude	141.00	0.00	130.00	73.50	6.65	128.00	156.12	1.80	11.10
19	2404025	Marsoui	289.00	0.00	309.00	142.00	6.25	275.00	181.57	1.95	18.00
20	2404030	La Martre	194.00	0.00	245.00	91.50	6.40	243.00	175.34	2.35	22.75
21	2404037	Sainte-Anne-des-Monts	6121.00	1.98	6933.00	3025.00	6.15	6437.00	263.80	2.05	16.30
22	2404047	Cap-Chat	2516.00	1.63	2623.00	1174.50	6.45	2476.00	181.68	2.05	14.50
23	2404902	Mont-Albert	167.00	0.00	204.00	89.50	5.60	179.00	3466.82	1.95	23.90
24	2404904	Coulée-des-Adolphe	0.00	NaN	0.00	0.00	NaN	0.00	86.41	NaN	NaN
25	2405010	Shigawake	333.00	0.00	338.00	135.50	7.15	292.00	76.82	2.30	12.70
26	2405015	Saint-Godefroi	350.00	5.13	405.00	178.50	6.65	380.00	63.55	2.05	10.25
27	2405020	Hope Town	334.00	0.00	344.00	140.50	6.40	339.00	51.08	2.45	23.90
28	2405025	Hope	584.00	3.33	630.00	276.50	6.60	568.00	70.55	2.05	12.95
29	2405032	Paspébiac	3033.00	2.84	3198.00	1418.50	6.40	3164.00	94.72	2.15	14.40
30	2405040	New Carlisle	1336.00	4.17	1358.00	588.50	6.25	1388.00	67.88	2.15	19.10
31	2405045	Bonaventure	2733.00	4.56	2775.00	1263.00	6.10	2706.00	104.42	2.10	13.35

32	2405050	Saint-Elzéar	464.00	0.00	467.00	207.50	6.50	458.00	203.87	2.15	13.05
33	2405055	Saint-Siméon	1207.00	3.48	1179.00	563.00	6.20	1171.00	56.79	2.10	13.75
34	2405060	Caplan	1966.00	4.89	2039.00	911.00	6.05	2024.00	85.50	2.05	9.85
35	2405065	Saint-Alphonse	711.00	3.17	691.00	311.50	6.05	699.00	112.08	2.30	12.90
36	2405070	New Richmond	3683.00	2.62	3810.00	1689.00	6.10	3706.00	171.92	2.15	10.90
37	2405077	Cascapédia--Saint-Jules	764.00	8.45	741.00	335.50	6.35	730.00	163.87	2.20	26.85
38	2405902	Rivière-Bonaventure	59.00	NaN	25.00	27.50	NaN	35.00	3056.37	1.60	NaN
39	2406005	Maria	2760.00	5.15	2536.00	1119.50	6.55	2615.00	94.98	2.20	7.45
40	2406013	Carleton-sur-Mer	4081.00	4.22	3991.00	1967.00	6.25	4073.00	221.45	2.05	12.95
41	2406020	Nouvelle	1782.00	2.96	1689.00	787.00	6.60	1670.00	233.55	2.15	14.55
42	2406025	Escuminac	575.00	0.00	588.00	254.50	5.95	544.00	108.45	2.25	16.45
43	2406030	Pointe-à-la-Croix	1344.00	1.56	1457.00	647.50	5.70	1399.50	390.85	2.10	14.15
44	2406035	Ristigouche-Partie-Sud-Est	170.00	0.00	167.00	81.50	6.05	171.00	51.73	2.10	30.00
45	2406040	Saint-André-de-Restigouche	154.00	0.00	157.00	76.50	6.45	161.00	143.95	2.05	28.20
46	2406045	Matapédia	566.00	0.00	664.00	310.50	6.20	645.00	71.57	1.90	18.00
47	2406050	Saint-Alexis-de-Matapédia	519.00	0.00	548.00	240.50	6.55	500.00	84.50	2.05	22.45
48	2406055	Saint-François-d'Assise	679.00	3.08	706.00	313.50	6.00	644.00	178.95	2.05	24.20
49	2406060	L'Ascension-de-Patapédia	148.00	0.00	190.00	88.00	6.60	164.00	95.96	1.80	33.35
50	2406802	Gesgapegiag	637.00	10.42	688.00	250.00	6.65	653.00	2.14	2.55	26.35
51	2406804	Listuguj	NaN	NaN	1865.00	579.00	6.40	1377.50	44.25	2.10	32.60
52	2406902	Rivière-Nouvelle	0.00	NaN	0.00	0.00	NaN	0.00	1087.91	NaN	NaN
53	2406904	Ruisseau-Ferguson	0.00	NaN	0.00	0.00	NaN	0.00	677.21	NaN	NaN
54	2407005	Sainte-Marguerite-Marie	183.00	0.00	203.00	85.00	6.45	166.00	85.98	2.10	20.50
55	2407010	Sainte-Florence	367.00	0.00	414.00	189.00	6.25	384.00	103.53	2.00	13.55
56	2407018	Causapscal	2147.00	1.42	2458.00	1061.50	6.20	2304.00	161.61	2.05	14.60
57	2407025	Albertville	239.00	0.00	256.00	117.50	6.05	226.00	103.19	2.00	29.25
58	2407030	Saint-Léon-le-Grand	968.00	6.10	970.00	420.00	6.55	953.00	128.37	2.25	14.15
59	2407035	Saint-Zénon-du-Lac-Humqui	370.00	5.26	366.00	181.00	5.95	359.00	113.25	2.05	13.85
60	2407040	Sainte-Irène	369.00	0.00	341.00	167.00	6.65	327.00	135.17	2.10	12.65
61	2407047	Amqui	5999.00	1.23	6322.00	2854.00	6.20	6178.00	121.09	2.05	8.65
62	2407057	Lac-au-Saumon	1488.00	2.48	1453.00	584.00	6.35	1450.00	81.06	2.25	9.45
63	2407065	Saint-Alexandre-des-Lacs	298.00	0.00	258.00	142.00	6.75	297.50	89.94	2.20	14.95
64	2407070	Saint-Tharcisius	411.00	0.00	464.00	158.50	6.10	368.00	78.97	2.25	18.05
65	2407075	Saint-Vianney	446.00	0.00	477.00	208.00	6.50	441.00	145.80	2.15	7.50

66	2407080	Val-Brillant	899.00	0.00	955.00	417.50	6.40	927.00	77.97	2.20	13.85
67	2407085	Sayabec	1706.00	1.23	1864.00	763.50	6.05	1712.50	130.52	2.15	9.75
68	2407090	Saint-Cléophas	321.00	0.00	334.00	144.00	6.75	333.00	97.33	2.25	13.60
69	2407095	Saint-Moïse	542.00	0.00	577.00	265.50	6.00	580.00	109.87	2.05	10.25
70	2407100	Saint-Noël	392.00	0.00	434.00	184.50	6.30	398.00	45.83	2.10	5.55
71	2407105	Saint-Damase	382.00	0.00	397.00	171.50	6.75	391.00	116.56	2.35	7.15
72	2407902	Routhierville	20.00	NaN	15.00	11.00	NaN	18.00	627.21	NaN	NaN
73	2407904	Rivière-Vaseuse	0.00	NaN	5.00	0.00	NaN	0.00	275.43	NaN	NaN
74	2407906	Rivière-Patapédia-Est	0.00	NaN	0.00	0.00	NaN	0.00	15.91	NaN	NaN
75	2407908	Lac-Casault	30.00	NaN	5.00	25.00	NaN	23.50	1448.59	1.50	NaN
76	2407910	Ruisseau-des-Mineurs	5.00	NaN	0.00	2.00	NaN	0.00	922.63	NaN	NaN
77	2407912	Lac-Alfred	0.00	NaN	0.00	0.00	NaN	0.00	77.53	NaN	NaN
78	2407914	Lac-Matapédia	10.00	NaN	5.00	46.50	5.60	88.50	71.22	2.00	25.00
79	2408005	Les Méchins	995.00	1.89	1107.00	513.50	5.95	987.00	442.39	1.95	17.15
80	2408010	Saint-Jean-de-Cherbourg	163.00	0.00	193.00	55.50	6.50	125.50	114.00	2.15	43.55
81	2408015	Grosses-Roches	375.00	0.00	411.00	185.00	6.25	350.00	63.80	1.80	17.10
82	2408023	Sainte-Félicité	1100.00	0.00	1175.00	502.00	6.40	1076.00	91.06	2.10	11.60
83	2408030	Saint-Adelme	484.00	0.00	485.00	262.00	6.25	606.00	100.61	2.25	14.70
84	2408035	Saint-René-de-Matane	961.00	0.00	1089.00	466.50	6.15	978.00	254.57	2.05	14.65
85	2408040	Sainte-Paule	247.00	0.00	201.00	122.00	6.40	233.00	83.54	1.95	7.70
86	2408053	Matane	13987.00	1.86	14462.00	6931.00	6.00	14311.00	195.79	1.95	8.90
87	2408065	Saint-Léandre	375.00	0.00	402.00	188.50	6.35	400.00	104.87	2.05	15.95
88	2408073	Saint-Ulric	1567.00	0.00	1642.00	768.00	6.20	1585.00	120.55	2.05	9.10
89	2408080	Baie-des-Sables	613.00	0.00	609.00	273.50	6.85	628.00	65.13	2.15	13.85
90	2408902	Rivière-Bonjour	16.00	NaN	10.00	15.00	NaN	21.50	1680.72	1.80	NaN
91	2409005	La Rédemption	384.00	0.00	488.00	183.00	6.20	432.00	116.90	2.20	11.65
92	2409010	Saint-Charles-Garnier	222.00	0.00	271.00	111.50	6.30	240.00	84.78	2.05	13.90
93	2409015	Les Hauteurs	485.00	7.50	524.00	214.00	6.35	504.00	102.98	2.30	15.80
94	2409020	Sainte-Jeanne-d'Arc	217.00	0.00	313.00	120.50	6.90	280.00	110.44	2.10	22.90
95	2409025	Saint-Gabriel-de-Rimouski	1177.00	0.00	1180.00	510.50	6.80	1167.00	127.01	2.25	9.45
96	2409030	Saint-Donat	835.00	2.35	890.00	397.50	6.60	876.00	95.69	2.15	4.95
97	2409035	Sainte-Angèle-de-Mérici	989.00	0.00	999.00	440.50	6.40	953.00	107.80	2.20	5.60
98	2409040	Padoue	250.00	0.00	273.00	110.00	6.70	245.00	66.69	2.25	13.35
99	2409048	Métis-sur-Mer	594.00	3.33	607.00	295.50	6.85	579.50	48.41	1.95	17.85

PART 0 - IMPORT FILES, CODE_MUN, GEOCODING

drive + librairies

```
#-------------------------------------------------------------------------------
# Set up drive
#-------------------------------------------------------------------------------
from google.colab import drive
from google.colab import files
drive.mount('/content/gdrive')
```

```
Mounted at /content/gdrive
```

```
#-------------------------------------------------------------------------------
# Import libaries
#-------------------------------------------------------------------------------
import pandas as pd
import numpy as np

pd.set_option('display.max_columns', None)
pd.set_option('display.max_rows', None)


pd.options.display.float_format = '{:.2f}'.format
```

code_mun

```
code_mun = pd.read_csv(r'/content/gdrive/MyDrive/mémoire/Données/code_geo_mun.csv', encoding= "utf-8")
```

```
code_mun.head()
```

	mcode	munnom	madr1	madr2	madr3	madr4	mcodpos	mcourriel	mweb	mtel	
0	46005	Abercorn	10, rue des Églises O.	Abercorn	10, rue des Églises O.	Abercorn	J0E1B0	info@abercorn.ca	https://municipalites-du-quebec.ca/abercorn/bi...	4505382664	
1	48028	Acton Vale	1025, rue Boulay	Acton Vale	1025, rue Boulay	Acton Vale	J0H1A0	actonvale@ville.actonvale.qc.ca	www.ville.actonvale.qc.ca	4505462703	45
2	31056	Adstock	35, rue Principale O.	Adstock	35, rue Principale O.	Adstock	G0N1S0	dg@adstock.ca	www.adstock.ca	4184222135	41
3	98030	Aguanish	106, route Jacques-Cartier	Aguanish	C.P. 47	Aguanish	G0G1A0	direction@mun.aguanish.org	www.aguanish.org	4185332323	41
4	99125	Akulivik	C.P. 50	Akulivik	C.P. 50	Akulivik	J0M1V0	sec-treasurer@nvakulivik.ca	www.nvakulivik.ca	8194962222	81

```
# ajouter "24" devant tous les mcode (car ils sont dans ce format dans les census)
code_mun['mcode'] = '24' + code_mun['mcode'].astype(str)

#change la colonne regadm : j'ai ça "Estrie (05)" et je veux ça : "Estrie"
code_mun['regadm'] = code_mun['regadm'].str.split(" ").str[0].str.strip()

#seulement garder les colonnes mcode, munnom, mdes, regadm
code_mun = code_mun[['mcode', 'munnom', 'mdes', 'regadm', 'mcodpos']]
```

```
len(code_mun)
```

```
1284
```

```
code_mun.head(15)
```

Afficher la sortie masquée

```
print(code_mun.loc[code_mun["munnom"] == "Les Îles-de-la-Madeleine"])
```

Afficher la sortie masquée

PART 1 - HYDRO QC DATA

donnees

```
#-------------------------------------------------------------------------------
# Import files and do basic cleaning
#-------------------------------------------------------------------------------
# Electricity data from hydro QC
elec_df_raw = pd.read_csv(r'/content/gdrive/MyDrive/mémoire/Données/conso_elec_imput.csv')
elec_res = elec_df_raw[elec_df_raw['SECTEUR'] == "RÉSIDENTIEL"]
```

```
elec_res.head()
```

	Unnamed: 0	REGION_ADM_QC_TXT	MRC_TXT	NOM_MUNICIPALITE	ANNEE_MOIS	SECTEUR	Total (kWh)	MOIS_NUM	Consommation_Normalisee	A
0	0	Montérégie	Acton	Acton Vale	2016-01	RÉSIDENTIEL	8163358.00	1	0.72	
1	1	Montérégie	Acton	Acton Vale	2016-02	RÉSIDENTIEL	7159392.00	2	0.58	
2	2	Montérégie	Acton	Acton Vale	2016-03	RÉSIDENTIEL	6484247.00	3	0.49	
3	3	Montérégie	Acton	Acton Vale	2016-04	RÉSIDENTIEL	5245742.00	4	0.32	
4	4	Montérégie	Acton	Acton Vale	2016-05	RÉSIDENTIEL	3656639.00	5	0.10	

```
# nouvelle colonne imputées + normales où quand total kwh n'est pas null, on le prend sinon consommation_denormalisee

import numpy as np
elec_res['imputées + normales'] = np.where(elec_res['Total (kWh)'].notnull(), elec_res['Total (kWh)'], elec_res['Consommation_Denormalis
```

```
# nettoyage

elec_res = elec_res.drop('Unnamed: 0', axis=1)
elec_res = elec_res.rename(columns={'Total (kWh)': 'donnes_originales', 'imputées + normales': 'Total (kWh)'})
```

```
elec_res.head()
```

	REGION_ADM_QC_TXT	MRC_TXT	NOM_MUNICIPALITE	ANNEE_MOIS	SECTEUR	donnes_originales	MOIS_NUM	Consommation_Normalisee	ANN
0	Montérégie	Acton	Acton Vale	2016-01	RÉSIDENTIEL	8163358.00	1	0.72	
1	Montérégie	Acton	Acton Vale	2016-02	RÉSIDENTIEL	7159392.00	2	0.58	
2	Montérégie	Acton	Acton Vale	2016-03	RÉSIDENTIEL	6484247.00	3	0.49	
3	Montérégie	Acton	Acton Vale	2016-04	RÉSIDENTIEL	5245742.00	4	0.32	
4	Montérégie	Acton	Acton Vale	2016-05	RÉSIDENTIEL	3656639.00	5	0.10	

EXPLORATION

Général

```
elec_df_raw.head()
```

	Unnamed: 0	REGION_ADM_QC_TXT	MRC_TXT	NOM_MUNICIPALITE	ANNEE_MOIS	SECTEUR	Total (kWh)	MOIS_NUM	Consommation_Normalisee	A
0	0	Montérégie	Acton	Acton Vale	2016-01	RÉSIDENTIEL	8163358.00	1	0.72	
1	1	Montérégie	Acton	Acton Vale	2016-02	RÉSIDENTIEL	7159392.00	2	0.58	
2	2	Montérégie	Acton	Acton Vale	2016-03	RÉSIDENTIEL	6484247.00	3	0.49	
3	3	Montérégie	Acton	Acton Vale	2016-04	RÉSIDENTIEL	5245742.00	4	0.32	
4	4	Montérégie	Acton	Acton Vale	2016-05	RÉSIDENTIEL	3656639.00	5	0.10	

```
len(elec_df_raw['NOM_MUNICIPALITE'].unique())
```

```
437
```

```
len(elec_df_raw)
```

```
42240
```

```
#on voit que toutes les municipalités ont une région associée

mrc_miss = elec_df_raw.loc[elec_df_raw['REGION_ADM_QC_TXT'].isna()]
print(mrc_miss)
```

```
Empty DataFrame
Columns: [Unnamed: 0, REGION_ADM_QC_TXT, MRC_TXT, NOM_MUNICIPALITE, ANNEE_MOIS, SECTEUR, Total (kWh), MOIS_NUM, Consommation_Normali
Index: []
```

Hiver/Autres saisons

2016 - 2023

```
elec_df_raw['ANNEE_MOIS'].max()
```

```
'2023-12'
```

```
# Sélectionner les mois d'hiver (novembre à mars) a partir de 2016
elec_df_raw['ANNEE_MOIS'] = pd.to_datetime(elec_df_raw['ANNEE_MOIS'])
elec_df = elec_res.copy()
elec_df_hiver = elec_df.loc[elec_df_raw['ANNEE_MOIS'].dt.month.isin([11, 12, 1, 2, 3])]

# Calculer le nombre de valeurs manquantes par municipalité
missing_counts = elec_df_hiver.groupby(['NOM_MUNICIPALITE', 'REGION_ADM_QC_TXT',    'MRC_TXT'])['Total (kWh)'].apply(lambda x: x.isnull(

# Créer un tableau de fréquences des nombres de valeurs manquantes
frequency_table = missing_counts.value_counts().sort_index()

# Ajouter une colonne pour les pourcentages
total_municipalities = len(missing_counts)
frequency_table_percent = frequency_table / total_municipalities * 100

# Ajouter une colonne pour les pourcentages cumulés
frequency_table_percent_cum = frequency_table_percent.cumsum()

# Afficher le tableau de fréquences
result_df = pd.DataFrame({
    'Nombre de valeurs manquantes': frequency_table.index,
    'Nombre de municipalités': frequency_table.values,
    'Pourcentage de municipalités': frequency_table_percent.values,
    'Pourcentage de municipalités (cumulé)': frequency_table_percent_cum.values
})

result_df
```

Afficher la sortie masquée

```
# Calculer le nombre de valeurs manquantes par municipalité
missing_counts = elec_df.groupby(['NOM_MUNICIPALITE', 'REGION_ADM_QC_TXT',  'MRC_TXT'])['Total (kWh)'].apply(lambda x: x.isnull().sum()

# Créer un tableau de fréquences des nombres de valeurs manquantes
frequency_table = missing_counts.value_counts().sort_index()
```

```
# Ajouter une colonne pour les pourcentages
total_municipalities = len(missing_counts)
frequency_table_percent = frequency_table / total_municipalities * 100

# Ajouter une colonne pour les pourcentages cumulés
frequency_table_percent_cum = frequency_table_percent.cumsum()

# Afficher le tableau de fréquences
result_df = pd.DataFrame({
    'Nombre de valeurs manquantes': frequency_table.index,
    'Nombre de municipalités': frequency_table.values,
    'Pourcentage de municipalités': frequency_table_percent.values,
    'Pourcentage de municipalités (cumulé)': frequency_table_percent_cum.values
})

result_df
```

	Nombre de valeurs manquantes	Nombre de municipalités	Pourcentage de municipalités	Pourcentage de municipalités (cumulé)
0	0	405	100.00	100.00

```
# parmi les municipalités qui ont au moins une valeur Total (kWh), lesquelles n'en ont aucune en hiver ?
elec_df_one_val = elec_df.dropna(subset=['Total (kWh)'])

elec_df_one_val['ANNEE_MOIS'] = pd.to_datetime(elec_df_one_val['ANNEE_MOIS'])
elec_df_win = elec_df_one_val[(elec_df_one_val['ANNEE_MOIS'].dt.month.isin([11, 12, 1, 2, 3]))]
elec_df_val_win = elec_df_win.dropna(subset=['Total (kWh)'])

print("Le nombre de villes avec au moins une valeur : ")
print(len(elec_df_one_val['NOM_MUNICIPALITE'].unique()))
print("Le nombre de villes avec au moins une valeur en hiver : ")
print(len(elec_df_val_win['NOM_MUNICIPALITE'].unique()))
```

```
---------------------------------------------------------------------------
AttributeError                            Traceback (most recent call last)
<ipython-input-30-a2f68510b849> in <cell line: 3>()
      1 # parmi les municipalités qui ont au moins une valeur Total (kWh), lesquelles n'en ont aucune en hiver ?
      2 elec_df_one_val = elec_df.dropna(subset=['Total (kWh)'])
----> 3 elec_df_win = elec_df_one_val[(elec_df_one_val['ANNEE_MOIS'].dt.month.isin([11, 12, 1, 2, 3]))]
      4 elec_df_val_win = elec_df_win.dropna(subset=['Total (kWh)'])
      5

                                 2 frames
/usr/local/lib/python3.10/dist-packages/pandas/core/indexes/accessors.py in __new__(cls, data)
    641             return PeriodProperties(data, orig)
    642
--> 643         raise AttributeError("Can only use .dt accessor with datetimelike values")

AttributeError: Can only use .dt accessor with datetimelike values
```

```
# comparer la consommation médiane en hiver et celle en été

# Sélectionner les mois d'hiver (novembre à mars)
elec_df_hiver = elec_df[(elec_df['ANNEE_MOIS'].dt.month.isin([11, 12, 1, 2, 3]))]

# Sélectionner les mois d'été (juin à août)
elec_df_ete = elec_df[(elec_df['ANNEE_MOIS'].dt.month.isin([4, 5, 6, 7, 8, 9, 10]))]

# Calculer la consommation médiane en hiver
median_hiver = elec_df_hiver['Total (kWh)'].median()

# Calculer la consommation médiane en été
median_ete = elec_df_ete['Total (kWh)'].median()

# Afficher les résultats
print("Consommation médiane en hiver :", median_hiver)
print("Consommation médiane en été :", median_ete)

# Comparer les consommations
if median_hiver > median_ete:
  print("La consommation médiane est plus élevée en hiver.")
elif median_hiver < median_ete:
  print("La consommation médiane est plus élevée en été.")
else:
```

```
  print("La consommation médiane est la même en hiver et en été.")
```

```
# boxplot de la distribution de ma consommation médiane en hiver puis en été par municipalité

import seaborn as sns
import matplotlib.pyplot as plt

# Calculer la consommation médiane en hiver par municipalité
median_hiver_par_municipalite = elec_df_hiver.groupby('NOM_MUNICIPALITE')['Total (kWh)'].median()

# Calculer la consommation médiane en été par municipalité
median_ete_par_municipalite = elec_df_ete.groupby('NOM_MUNICIPALITE')['Total (kWh)'].median()

# Créer un DataFrame pour les données du boxplot
df_boxplot = pd.DataFrame({
    'Saison': ['Hiver'] * len(median_hiver_par_municipalite) + ['Été'] * len(median_ete_par_municipalite),
    'Consommation médiane (kWh)': list(median_hiver_par_municipalite) + list(median_ete_par_municipalite)
})

# Créer le boxplot
plt.figure(figsize=(10, 6))
sns.boxplot(x='Saison', y='Consommation médiane (kWh)', data=df_boxplot)
plt.title('Distribution de la consommation médiane en hiver et en été par municipalité')
plt.show()
```

```
# municipalités dont les consommations médianes sont très différentes de ces valeurs (été et hiver) puis liste les moi avec leur consomm

# Calculer la consommation médiane en hiver et en été par municipalité
median_consommation = elec_df.groupby(['NOM_MUNICIPALITE', elec_df['ANNEE_MOIS'].dt.month.isin([11, 12, 1, 2, 3])])['Total (kWh)'].media
median_consommation.columns = ['Ete', 'Hiver']

# Calculer l'écart à la médiane globale pour chaque municipalité
median_consommation['Ecart_Ete'] = abs(median_consommation['Ete'] - median_ete)
median_consommation['Ecart_Hiver'] = abs(median_consommation['Hiver'] - median_hiver)

# Trier les municipalités par écart à la médiane
median_consommation_sorted = median_consommation.sort_values(by=['Ecart_Ete', 'Ecart_Hiver'], ascending=False)

# Afficher les municipalités avec les écarts les plus importants
print(median_consommation_sorted.head(10))

# Lister les mois avec leur consommation hivernale et estivale pour une municipalité donnée
municipalite = "Montréal"  # Remplacer par la municipalité souhaitée

# Filtrer les données pour la municipalité sélectionnée
elec_df_municipalite = elec_df[elec_df['NOM_MUNICIPALITE'] == municipalite]

# Calculer la consommation mensuelle
consommation_mensuelle = elec_df_municipalite.groupby(['ANNEE_MOIS'])['Total (kWh)'].sum()

# Afficher les résultats
print("\nConsommation mensuelle pour", municipalite, ":")
print(consommation_mensuelle)
```

MERGE ET CLEANING

```
filtered_elec_df = elec_res.copy()
```

```
len(filtered_elec_df['NOM_MUNICIPALITE'].unique())
```

```
437
```

```
#--------------------on s'intéresse aux noms de municipalités présents dans plusieurs mrc et/ou régions

# Groupe les lignes par NOM_MUNICIPALITE et agrège les valeurs de MRC_TXT dans une liste
municipality_mrc = filtered_elec_df.groupby('NOM_MUNICIPALITE')['MRC_TXT'].agg(set).reset_index()

# Filtre les municipalités avec plus d'un MRC unique
municipality_mrc_multiple = municipality_mrc[municipality_mrc['MRC_TXT'].apply(len) > 1]

# Affiche le résultat
print(municipality_mrc_multiple)
```

Afficher la sortie masquée

```
# Groupe les lignes par NOM_MUNICIPALITE et agrège les valeurs uniques de REGION_ADM_QC_TXT dans une liste
municipality_region = filtered_elec_df.groupby('NOM_MUNICIPALITE')['REGION_ADM_QC_TXT'].agg(set).reset_index()

# Filtre les municipalités avec plus d'une région unique
municipality_region_multiple = municipality_region[municipality_region['REGION_ADM_QC_TXT'].apply(len) > 1]

# Affiche le résultat
print(municipality_region_multiple)
```

Afficher la sortie masquée

```
#voir les lignes dupliquées (= aucune)

duplicate_rows = filtered_elec_df[filtered_elec_df.duplicated()]
print(duplicate_rows)
```

```
Empty DataFrame
Columns: [REGION_ADM_QC_TXT, MRC_TXT, NOM_MUNICIPALITE, ANNEE_MOIS, SECTEUR, donnes_originales, MOIS_NUM, Consommation_Normalisee, A
Index: []
```

```
# modifier Pessamit dans code_num en Betsiamites et Courcelles--Saint-Évariste en Courcelles --> deux municipalités qui ont changé de nc
#et qui n'ont pas été actualisées par HQ

code_mun.loc[code_mun['munnom'] == 'Pessamit', 'munnom'] = 'Betsiamites'
code_mun.loc[code_mun['munnom'] == 'Courcelles--Saint-Évariste', 'munnom'] = 'Courcelles'



# Merge elec_df et code_mun sur municipalité et region
merged_df = pd.merge(filtered_elec_df, code_mun, left_on=['NOM_MUNICIPALITE', 'REGION_ADM_QC_TXT'], right_on=['munnom', 'regadm'], how=


merged_df.head()
```

	REGION_ADM_QC_TXT	MRC_TXT	NOM_MUNICIPALITE	ANNEE_MOIS	SECTEUR	donnes_originales	MOIS_NUM	Consommation_Normalisee	ANN
0	Montérégie	Acton	Acton Vale	2016-01	RÉSIDENTIEL	8163358.00	1	0.72	
1	Montérégie	Acton	Acton Vale	2016-02	RÉSIDENTIEL	7159392.00	2	0.58	
2	Montérégie	Acton	Acton Vale	2016-03	RÉSIDENTIEL	6484247.00	3	0.49	
3	Montérégie	Acton	Acton Vale	2016-04	RÉSIDENTIEL	5245742.00	4	0.32	
4	Montérégie	Acton	Acton Vale	2016-05	RÉSIDENTIEL	3656639.00	5	0.10	

```
# del munnom_x  mdes_x  regadm_x mcode_y    munnom_y    mdes_y  regadm_y    mcodpos_y

merged_df = merged_df.drop(['munnom', 'mdes', 'regadm', 'SECTEUR'], axis=1)



# Afficher toutes les municipalités de merged_df qui n'ont pas de code_geo
missing_codes = merged_df[merged_df['mcode'].isna()]['NOM_MUNICIPALITE'].unique()
print(missing_codes)
```

```
['Courcelles' 'Dollard-Des Ormeaux' 'La Morandière' 'Nemiscau' 'Winneway']
```

```
# ENlever les lignes avec Winneway de merged_df

merged_df = merged_df[merged_df['NOM_MUNICIPALITE'] != 'Winneway']



# ajouter mcodpos et mcode de code_mun de la municipalité La Morandière-Rochebaucourt à la municipalité dans merged_df La morandière

la_morandiere_rochebaucourt_data = code_mun[code_mun['munnom'] == 'La Morandière-Rochebaucourt']

if not la_morandiere_rochebaucourt_data.empty:
  mcodpos_la_morandiere_rochebaucourt = la_morandiere_rochebaucourt_data['mcodpos'].iloc[0]
  mcode_la_morandiere_rochebaucourt = la_morandiere_rochebaucourt_data['mcode'].iloc[0]
```

```
  merged_df.loc[merged_df['NOM_MUNICIPALITE'] == 'La Morandière', 'mcodpos'] = mcodpos_la_morandiere_rochebaucourt
  merged_df.loc[merged_df['NOM_MUNICIPALITE'] == 'La Morandière', 'mcode'] = mcode_la_morandiere_rochebaucourt



# récupèrer dans df_courcelles et les informations sur courcelles de code_mun puis ajoute les à merged_df pour avoir les colonnes mcode

# Récupérer les informations sur Courcelles dans code_mun
df_courcelles = code_mun[code_mun['munnom'] == 'Courcelles']

# Extraire les valeurs de 'mcode' et 'mcodpos'
mcode_courcelles = df_courcelles['mcode'].iloc[0]
mcodpos_courcelles = df_courcelles['mcodpos'].iloc[0]

# Ajouter les valeurs à merged_df pour Courcelles
merged_df.loc[merged_df['NOM_MUNICIPALITE'] == 'Courcelles', 'mcode'] = mcode_courcelles
merged_df.loc[merged_df['NOM_MUNICIPALITE'] == 'Courcelles', 'mcodpos'] = mcodpos_courcelles



# changer le nom de municiplaité Dollard-des Ormeaux" en "Dollard-des-Ormeaux" dans merged-df

merged_df.loc[merged_df['NOM_MUNICIPALITE'] == 'Dollard-Des Ormeaux', 'NOM_MUNICIPALITE'] = 'Dollard-des-Ormeaux'

# Récupérer les informations sur Dollard-des-Ormeaux dans code_mun
df_dollard = code_mun[code_mun['munnom'] == 'Dollard-des-Ormeaux']

# Extraire les valeurs de 'mcode' et 'mcodpos'
mcode_dollard = df_dollard['mcode'].iloc[0]
mcodpos_dollard = df_dollard['mcodpos'].iloc[0]

# Ajouter les valeurs à merged_df pour Dollard-des-Ormeaux
merged_df.loc[merged_df['NOM_MUNICIPALITE'] == 'Dollard-des-Ormeaux', 'mcode'] = mcode_dollard
merged_df.loc[merged_df['NOM_MUNICIPALITE'] == 'Dollard-des-Ormeaux', 'mcodpos'] = mcodpos_dollard


# fonction pour calculer mois hivernaux et autres

def calculate_stats(df):
  winter_months = [11, 12, 1, 2, 3]
  is_winter = df['ANNEE_MOIS'].dt.month.isin(winter_months)
  winter_df = df[is_winter]
  summer_df = df[~is_winter]
  return pd.Series({
      'median_total_kwh_winter': winter_df['Total (kWh)'].median(),
      'mean_total_kwh_winter': winter_df['Total (kWh)'].mean(),
      'median_total_kwh_summer': summer_df['Total (kWh)'].median(),
      'mean_total_kwh_summer': summer_df['Total (kWh)'].mean(),
      'median_total_kwh_all': df['Total (kWh)'].median(),
      'mean_total_kwh_all': df['Total (kWh)'].mean(),
      'mcode': df['mcode'].iloc[0],  # Take the first value, assuming it's the same for the group
      'mcodpos' : df['mcodpos'].iloc[0]
  })

merged_df['ANNEE_MOIS'] = pd.to_datetime(merged_df['ANNEE_MOIS'])
merged_df_cleaned = merged_df.groupby(['NOM_MUNICIPALITE', 'REGION_ADM_QC_TXT']).apply(calculate_stats).reset_index()
```

```
<ipython-input-66-d10c581bd379>:21: DeprecationWarning: DataFrameGroupBy.apply operated on the grouping columns. This behavior is de
  merged_df_cleaned = merged_df.groupby(['NOM_MUNICIPALITE', 'REGION_ADM_QC_TXT']).apply(calculate_stats).reset_index()
```

```
# renommer les colonnes 'REGION_ADM_QC_TXT', 'NOM_MUNICIPALITE', 'mcode' par    "region" "nom_geo" "code_geo"

grouped_df = merged_df_cleaned.rename(columns={'REGION_ADM_QC_TXT': 'region', 'NOM_MUNICIPALITE': 'nom_geo', 'mcode': 'code_geo', 'mcod



grouped_df.head(100)
```

	nom_geo	region	median_total_kwh_winter	mean_total_kwh_winter	median_total_kwh_summer	mean_total_kwh_summer
0	Acton Vale	Montérégie	7471105.50	7342004.72	3520431.50	3774089.86
1	Aguanish	Côte-Nord	317612.00	308225.33	138171.50	156157.62
2	Akulivik	Nord-du-Québec	150509.07	152289.96	113064.41	115500.64
3	Akwesasne	Montérégie	1849454.43	1806491.27	1027017.79	1060787.26
4	Albertville	Bas-Saint-Laurent	215224.00	212729.90	116931.50	128677.51
5	Amqui	Bas-Saint-Laurent	5714833.50	5653613.30	3695964.69	3844199.15
6	Auclair	Bas-Saint-Laurent	447692.50	444469.03	234309.50	262890.45
7	Aupaluk	Nord-du-Québec	70664.50	70376.93	49022.50	50473.59
8	Austin	Estrie	4346085.00	4293900.90	1648454.50	1846520.30
9	Authier	Abitibi-Témiscamingue	209588.52	207948.38	117316.90	124925.22
10	Baie-D'Urfé	Montréal	5503273.50	5507386.97	2764132.50	2826615.64
11	Baie-Johan-Beetz	Côte-Nord	142711.04	141595.08	106266.54	108566.40
12	Baie-Saint-Paul	Capitale-Nationale	8416780.00	8339247.85	3589910.50	4035432.70
13	Baie-des-Sables	Bas-Saint-Laurent	617461.50	618386.03	288508.50	326723.04
14	Baie-du-Febvre	Centre-du-Québec	1035131.50	1018802.15	436368.50	475602.30
15	Beauceville	Chaudière-Appalaches	5854756.50	5740464.05	2818868.00	3067680.89
16	Beaumont	Chaudière-Appalaches	3101225.00	3054816.05	1633573.50	1702996.16
17	Bedford	Estrie	3526872.50	3494314.18	1751963.19	1883214.16
18	Beloeil	Montérégie	21852200.00	21488264.30	11533970.00	11745012.05
19	Berthierville	Lanaudière	4449149.00	4418030.70	1973401.00	2069267.77
20	Betsiamites	Côte-Nord	2219117.00	2185111.98	1198561.00	1246442.05
21	Blainville	Laurentides	59200809.50	58893108.05	33333495.50	33646212.05
22	Blanc-Sablon	Côte-Nord	1690489.00	1670062.25	871046.50	899859.27
23	Boileau	Outaouais	711279.50	713385.75	270538.00	302941.95

```
# enlever toutes les villes du Nord-du-Québec

grouped_df = grouped_df[grouped_df.region != 'Nord-du-Québec']
```

27	Brossard	Montérégie	49510000.05	48738575.84	21004214.22	22022507.20

```
len(grouped_df)
```

421

29	Brvson	Outaouais	744010.50	732495.47	328003.00	351988.91

CSV

31	Bury	Estrie	1150617.00	1121192.32	551930.50	609873.75

```
grouped_df.to_csv('grouped_df_imput.csv', encoding = 'latin1')
!cp grouped_df_imput.csv /content/gdrive/MyDrive/mémoire/Données/
```

33	Béthanie	Montérégie	258233.40	253844.32	132680.33	138058.96

PART 2 - CENSUS DATA

35	Campbell's Bay	Outaouais	729364.49	709821.34	331166.45	354868.30

a. Census data 2021

37	[illegible]	Appalaches	2978798.00	2856496.49	1410729.62	1507557.24

```
# ------------------------------------------------------------------------
# CENSUS DATA 2021
df = pd.read_csv(r'/content/gdrive/MyDrive/mémoire/Données/Census 2021/clean_stat_can_21_utf.csv', encoding= "utf_8")
```

```
# Rename columns of 2021 data
df = df.rename(columns={'CODE_GÉO_ALT':'code_geo', 'NOM_GÉO' : 'nom_geo', 'NOM_CARACTÉRISTIQUE' : 'nom_car', 'C1_CHIFFRE_TOTAL' : 'total
                        })
```

```
<ipython-input-4-93bcc83db9d7>:3: DtypeWarning: Columns (1) have mixed types. Specify dtype option on import or set low_memory=False
  df = pd.read_csv(r'/content/gdrive/MyDrive/mémoire/Données/Census 2021/clean_stat_can_21_utf.csv', encoding= "utf_8")
```

Québec

```
len(df)
```

```
39959628
```

Chateauguay Monteregie

```
df.head()
```

	code_geo	nom_geo	nom_car	total
0	1	Canada	Population, 2021	36991981.00
1	1	Canada	Population, 2016	35151728.00
2	1	Canada	Variation en pourcentage de la population, 201...	5.20
3	1	Canada	Total des logements privés	16284235.00
4	1	Canada	Logements privés occupés par des résidents hab...	14978941.00

51 Daveluyville 2321385.50 2307643.42 1126555.00 1207995.86

```
##garder seulement les lignes où le code géographique est composé de 7 chiffres (les municipalités)

df_filtered = df[df['code_geo'].astype(str).str.len() == 7]
```

```
#j'ai ça dans ma colonne NOM_GEO : "Les Îles-de-la-Madeleine, Municipalité (MÉ) " du census 2021 et je veux juste avoir ça : "Les Îles-d

df_filtered['nom_geo'] = df_filtered['nom_geo'].str.split(',').str[0]
```

Afficher la sortie masquée

Dunamer Outaouais

```
df_filtered.head(250)
```

Afficher la sortie masquée

Abitibi-

```
len(df_filtered)
```

```
3372942
```

```
# Cette fonction stocke les changements réalisés
changes = []

def log_change(old_value, new_value):
    changes.append({'Ancien_nom_car': old_value, 'Nouveau_nom_car': new_value})

# Code modifié pour enregistrer les changements de nom_car

### Type de logement
start_indices = df_filtered[df_filtered["nom_car"] == "Total - Logements privés occupés par type de construction résidentielle - Données
row_names = ["Type de construction résidentielle - Maison individuelle non attenante",
             "Type de construction résidentielle - Maison jumelée",
             "Type de construction résidentielle - Maison en rangée",
             "Type de construction résidentielle - Appartement ou plain-pied dans un duplex",
             "Type de construction résidentielle - Appartement dans un immeuble de moins de cinq étages",
             "Type de construction résidentielle - Appartement dans un immeuble de cinq étages ou plus"
             ]
for index in start_indices:
    for i in range(6):
        old_value = df_filtered.at[index + i + 1, "nom_car"]
        new_value = row_names[i]
        if old_value != new_value:  # To ensure that we only log actual changes
            df_filtered.at[index + i + 1, "nom_car"] = new_value
            log_change(old_value, new_value)

### Revenu des ménages
start_indices = df_filtered[df_filtered["nom_car"] == "Total - Statistiques du revenu pour les ménages privés - Données intégrales (100
row_names = ["Revenu total médian des ménages en 2020($) avant impôts",
             "Revenu total médian des ménages en 2020($) après impôts"]

for index in start_indices:
    for i in range(2):
```

```
            old_value = df_filtered.at[index + i + 1, "nom_car"]
            new_value = row_names[i]
            if old_value != new_value:
                df_filtered.at[index + i + 1, "nom_car"] = new_value
                log_change(old_value, new_value)

    ### Période de construction
    start_indices = df_filtered[df_filtered["nom_car"] == "Total - Logements privés occupés selon la période de construction - Données-échar
    row_names = [

                 "Logement occupé selon période de construction - 1960 ou avant",
                 "Logement occupé selon période de construction - 1961 à 1980",
                 "Logement occupé selon période de construction - 1981 à 1990",
                 "Logement occupé selon période de construction - 1991 à 2000",
                 "Logement occupé selon période de construction - 2001 à 2005",
                 "Logement occupé selon période de construction - 2006 à 2010",
                 "Logement occupé selon période de construction - 2011 à 2015",
                 "Logement occupé selon période de construction - 2016 à 2021"
                 ]
    for index in start_indices:
        for i in range(8):
            old_value = df_filtered.at[index + i + 1, "nom_car"]
            new_value = row_names[i]
            if old_value != new_value:
                df_filtered.at[index + i + 1, "nom_car"] = new_value
                log_change(old_value, new_value)

    ### Réparations majeures requises
    start_indices = df_filtered[df_filtered["nom_car"] == "Total - Logements privés occupés selon l'état du logement - Données-échantillon (
    row_names = ["Logement occupé selon l'état du logement - Réparations majeures requises"
                 ]
    for index in start_indices:
        for i in range(1):
            old_value = df_filtered.at[index + i + 2, "nom_car"]
            new_value = row_names[i]
            if old_value != new_value:
                df_filtered.at[index + i + 2, "nom_car"] = new_value
                log_change(old_value, new_value)

    ### Mode d'occupation
    start_indices = df_filtered[df_filtered["nom_car"] == "Total - Ménages privés selon le mode d'occupation - Données-échantillon (25 %)"].
    row_names = ["Ménages privés selon le mode d'occupation - Propriétaire",
                 "Ménages privés selon le mode d'occupation - Locataire",
                 "Ménages privés selon le mode d'occupation - Logement fourni par le gouvernement local"
                 ]
    for index in start_indices:
        for i in range(3):
            old_value = df_filtered.at[index + i + 1, "nom_car"]
            new_value = row_names[i]
            if old_value != new_value:
                df_filtered.at[index + i + 1, "nom_car"] = new_value
                log_change(old_value, new_value)

    ### Education
    start_indices = df_filtered[df_filtered["nom_car"] == "Total - Plus haut certificat, diplôme ou grade pour la population âgée de 25 à 64
    row_names = ["Plus haut diplôme - Aucun certificat, diplôme ou grade",
                 "Plus haut diplôme - Diplôme d'études secondaires ou attestation d'équivalence",
                 "Plus haut diplôme - Certificat, diplôme ou grade d’études postsecondaires"
                 ]
    for index in start_indices:
        for i in range(3):
            old_value = df_filtered.at[index + i + 1, "nom_car"]
            new_value = row_names[i]
            if old_value != new_value:
                df_filtered.at[index + i + 1, "nom_car"] = new_value
                log_change(old_value, new_value)

    ### Groupes d'âge
    start_indices = df_filtered[df_filtered["nom_car"] == "Total - Groupes d'âge de la population - Données intégrales (100 %)"].index
    row_names = ["Groupes d'âge de la population - 65 ans et plus"]

    for index in start_indices:
        for i in range(1):
            old_value = df_filtered.at[index + i + 16, "nom_car"]
            new_value = row_names[i]
            if old_value != new_value:
                df_filtered.at[index + i + 16, "nom_car"] = new_value
                log_change(old_value, new_value)

    ### Frais de logements
    start_indices = df_filtered[df_filtered["nom_car"] == "Total - Ménages propriétaires et locataires dont le revenu total du ménage est su
    row_names = ["Ménages dont 30 % ou plus du revenu est consacré aux frais de logement seulement"]
```

```
for index in start_indices:
    for i in range(1):
        old_value = df_filtered.at[index + i + 2, "nom_car"]
        new_value = row_names[i]
        if old_value != new_value:
            df_filtered.at[index + i + 2, "nom_car"] = new_value
            log_change(old_value, new_value)

 ### Revenus médians
start_indices = df_filtered[df_filtered["nom_car"] == "Total - Statistiques du revenu en 2020 pour la population âgée de 15 ans et plus
row_names = ["Revenu total médian avant impôts en 2020 parmi les bénéficiaires d'un revenu total ($)"]

for index in start_indices:
    for i in range(1):
        old_value = df_filtered.at[index + i + 2, "nom_car"]
        new_value = row_names[i]
        if old_value != new_value:
            df_filtered.at[index + i + 2, "nom_car"] = new_value
            log_change(old_value, new_value)

start_indices = df_filtered[df_filtered["nom_car"] == "Total - Statistiques du revenu en 2020 pour la population âgée de 15 ans et plus
row_names = ["Revenu total médian après impôts en 2020 parmi les bénéficiaires d'un revenu total ($)"]

for index in start_indices:
    for i in range(1):
        old_value = df_filtered.at[index + i + 4, "nom_car"]
        new_value = row_names[i]
        if old_value != new_value:
            df_filtered.at[index + i + 4, "nom_car"] = new_value
            log_change(old_value, new_value)

# Crée une DataFrame pour les changements et afficher les valeurs uniques
change_df = pd.DataFrame(changes)
unique_changes = change_df.drop_duplicates()
print(unique_changes)
```

Afficher la sortie masquée

```
df_filtered.head(500)
```

Afficher la sortie masquée

```
vars = [
    "Revenu total médian après impôts en 2020 parmi les bénéficiaires d'un revenu total ($)",
    "Revenu total médian avant impôts en 2020 parmi les bénéficiaires d'un revenu total ($)",
    "Revenu total médian des ménages en 2020($) avant impôts",
    "Revenu total médian des ménages en 2020($) après impôts",
    "Population, 2021",
    "Population, 2016",

    "Superficie des terres en kilomètres carrés",
    "Taille moyenne des ménages privés",
    "Nombre moyen de pièces par logement",
    "Logements privés occupés par des résidents habituels",
    "Total des logements privés",
    "Taux de chômage",

    #variables pour calculter mes taux

    "Total - Groupes d'âge de la population - Données intégrales (100 %)",

    "Total - Logements privés occupés selon l'état du logement - Données-échantillon (25 %)",

    "Total - Logements privés occupés par type de construction résidentielle - Données intégrales (100 %)",

    "Total - Logements privés occupés selon la période de construction - Données-échantillon (25 %)",

    "Total - Ménages privés selon le mode d'occupation - Données-échantillon (25 %)",

    "Total - Plus haut certificat, diplôme ou grade pour la population âgée de 25 à 64 ans dans les ménages privés - Données-échantillor

    "Total - Ménages propriétaires et locataires dont le revenu total du ménage est supérieur à zéro, dans les logements privés non agri
]


rate_vars = [
    "Type de construction résidentielle - Maison individuelle non attenante",
    "Type de construction résidentielle - Maison jumelée",
    "Type de construction résidentielle - Maison en rangée",
    "Type de construction résidentielle - Appartement ou plain-pied dans un duplex",
```

```
    "Type de construction résidentielle - Appartement dans un immeuble de moins de cinq étages",
    "Type de construction résidentielle - Appartement dans un immeuble de cinq étages ou plus",

    "Logement occupé selon période de construction - 1960 ou avant",
    "Logement occupé selon période de construction - 1961 à 1980",
    "Logement occupé selon période de construction - 1981 à 1990",
    "Logement occupé selon période de construction - 1991 à 2000",
    "Logement occupé selon période de construction - 2001 à 2005",
    "Logement occupé selon période de construction - 2006 à 2010",
    "Logement occupé selon période de construction - 2011 à 2015",
    "Logement occupé selon période de construction - 2016 à 2021",

    "Logement occupé selon l'état du logement - Réparations majeures requises",

    "Ménages privés selon le mode d'occupation - Propriétaire",
    "Ménages privés selon le mode d'occupation - Locataire",
    "Ménages privés selon le mode d'occupation - Logement fourni par le gouvernement local",

    "Plus haut diplôme - Aucun certificat, diplôme ou grade",
    "Plus haut diplôme - Diplôme d'études secondaires ou attestation d'équivalence",
    "Plus haut diplôme - Certificat, diplôme ou grade d’études postsecondaires",

    "Groupes d'âge de la population - 65 ans et plus",

    "Ménages dont 30 % ou plus du revenu est consacré aux frais de logement seulement"

]


# garder seulement les lignes ou nom_car = var + rate_var

all_vars = vars + rate_vars
df_filtered = df_filtered[df_filtered['nom_car'].isin(all_vars)]



all_vars
```

Afficher la sortie masquée

```
weird_var = set(all_vars) - set(df_filtered['nom_car'].unique())


weird_var
```

```
set()
```

```
df.iloc[1450:1500]
```

Afficher la sortie masquée

```
df_filtered.head(100)
```

Afficher la sortie masquée

```
len(df_filtered['code_geo'].unique())
```

```
1282
```

```
len(df_filtered['nom_geo'].unique())
```

```
1227
```

```
# nom_géo qui ont différents code_géo

# Groupe les lignes par 'nom_geo' et agrège les valeurs uniques de 'code_geo' dans une liste
grouped = df_filtered.groupby('nom_geo')['code_geo'].agg(set).reset_index()

# Filtre les noms_geo avec plus d'un code_geo unique
multiple_codes = grouped[grouped['code_geo'].apply(len) > 1]

# Affiche le résultat
print(multiple_codes)
```

Afficher la sortie masquée

```
# Créer une liste de toutes les municipalités uniques avec leur code géographique
unique_municipalities = df_filtered[['code_geo', 'nom_geo']].drop_duplicates()
```

```
# Transformer le DataFrame pour le fusionner avec les variables cibles
# Ici, nous utilisons le pivot pour réorganiser les données
df_pivoted = df_filtered.pivot_table(index=['code_geo', 'nom_geo'],
                                     columns='nom_car',
                                     values='total',
                                     fill_value=np.nan).reset_index()

# Maintenant, fusionner les municipalités uniques avec le DataFrame pivoté
df_final = unique_municipalities.merge(df_pivoted, on=['code_geo', 'nom_geo'], how='left')


df_final.head()
```

	code_geo	nom_geo	Groupes d'âge de la population - 65 ans et plus	Logement occupé selon l'état du logement - Réparations majeures requises	Logement occupé selon période de construction - 1960 ou avant	Logement occupé selon période de construction - 1961 à 1980	Logement occupé selon période de construction - 1981 à 1990	Logement occupé selon période de construction - 1991 à 2000	Logement occupé selon période de construction - 2001 à 2005	Logement occupé selon période de construction - 2006 à 2010
0	2401023	Les Îles-de-la-Madeleine	3485.00	520.00	1280.00	1760.00	1070.00	525.00	215.00	305.00
1	2401042	Grosse-Île	105.00	20.00	85.00	70.00	30.00	20.00	15.00	0.00
2	2402005	Percé	1065.00	160.00	675.00	470.00	175.00	85.00	30.00	30.00
3	2402010	Sainte-Thérèse-de-Gaspé	300.00	55.00	135.00	160.00	100.00	15.00	20.00	25.00
4	2402015	Grande-Rivière	1010.00	160.00	515.00	595.00	225.00	90.00	50.00	60.00

```
print(df_final.columns)
```

Afficher la sortie masquée

```
len(all_vars)
```

```
42
```

```
len(df_final.columns)
```

```
44
```

```
# quelles sont les colonnes qu'il y a dans all_vars et pas dans df_final

set(all_vars) - set(df_final.columns)
```

```
set()
```

```
len(df_final.columns)
```

```
44
```

```
# combien de valeurs manquantes et non manquantes il y a dans chacune de mes colonnes de df_final

missing_counts = df_final.isnull().sum()
non_missing_counts = df_final.notnull().sum()

result_df = pd.DataFrame({
    'Missing Values': missing_counts,
    'Non-Missing Values': non_missing_counts,
    'Pourcentage de valeurs manquantes totales' : (missing_counts / len(df_final))*100
})

print(result_df)
```

Afficher la sortie masquée

```
# pourcentages de lignes qui ont 40 valeurs nulles, puis 39, puis 38, etc. jusqu'à 0

# Calculer le nombre de valeurs nulles par ligne
null_counts = df_final.isnull().sum(axis=1)

# Compter le nombre de lignes avec chaque nombre de valeurs nulles
null_counts_frequency = null_counts.value_counts().sort_index(ascending=False)

# Calculer le pourcentage de lignes pour chaque nombre de valeurs nulles
null_counts_percentage = (null_counts_frequency / len(df_final)) * 100

# Afficher les pourcentages
print(null_counts_percentage)
```

```
41    0.31
40    0.31
37    8.27
28    0.08
27    0.39
6     5.69
2     0.47
0    84.48
Name: count, dtype: float64
```

```
null_counts_frequency
```

	count
41	4
40	4
37	106
28	1
27	5
6	73
2	6
0	1083

dtype: int64

```
# Calcul des pourcentages pour chaque colonne de `rate_vars`

# 1. Type de construction résidentielle - Maison individuelle non attenante (%)
df_final["Type de construction résidentielle - Maison individuelle non attenante (%)"] = (
    df_final["Type de construction résidentielle - Maison individuelle non attenante"] /
    df_final["Total - Logements privés occupés par type de construction résidentielle - Données intégrales (100 %)"]
) * 100

del df_final["Type de construction résidentielle - Maison individuelle non attenante"]

# 2. Type de construction résidentielle - Maison jumelée (%)
df_final["Type de construction résidentielle - Maison jumelée (%)"] = (
    df_final["Type de construction résidentielle - Maison jumelée"] /
    df_final["Total - Logements privés occupés par type de construction résidentielle - Données intégrales (100 %)"]
) * 100

del df_final["Type de construction résidentielle - Maison jumelée"]

# 3. Type de construction résidentielle - Maison en rangée (%)
df_final["Type de construction résidentielle - Maison en rangée (%)"] = (
```

```
    df_final["Type de construction résidentielle - Maison en rangée"] /
    df_final["Total - Logements privés occupés par type de construction résidentielle - Données intégrales (100 %)"]
) * 100

del df_final["Type de construction résidentielle - Maison en rangée"]

# 4. Type de construction résidentielle - Appartement ou plain-pied dans un duplex (%)
df_final["Type de construction résidentielle - Appartement ou plain-pied dans un duplex (%)"] = (
    df_final["Type de construction résidentielle - Appartement ou plain-pied dans un duplex"] /
    df_final["Total - Logements privés occupés par type de construction résidentielle - Données intégrales (100 %)"]
) * 100

del df_final["Type de construction résidentielle - Appartement ou plain-pied dans un duplex"]

# 5. Type de construction résidentielle - Appartement dans un immeuble de moins de cinq étages (%)
df_final["Type de construction résidentielle - Appartement dans un immeuble de moins de cinq étages (%)"] = (
    df_final["Type de construction résidentielle - Appartement dans un immeuble de moins de cinq étages"] /
    df_final["Total - Logements privés occupés par type de construction résidentielle - Données intégrales (100 %)"]
) * 100

del df_final["Type de construction résidentielle - Appartement dans un immeuble de moins de cinq étages"]

# 6. Type de construction résidentielle - Appartement dans un immeuble de cinq étages ou plus (%)
df_final["Type de construction résidentielle - Appartement dans un immeuble de cinq étages ou plus (%)"] = (
    df_final["Type de construction résidentielle - Appartement dans un immeuble de cinq étages ou plus"] /
    df_final["Total - Logements privés occupés par type de construction résidentielle - Données intégrales (100 %)"]
) * 100

del df_final["Type de construction résidentielle - Appartement dans un immeuble de cinq étages ou plus"]
del df_final["Total - Logements privés occupés par type de construction résidentielle - Données intégrales (100 %)"]

# 7. Logement occupé selon période de construction - 1960 ou avant (%)
df_final["Logement occupé selon période de construction - 1960 ou avant (%)"] = (
    df_final["Logement occupé selon période de construction - 1960 ou avant"] /
    df_final["Total - Logements privés occupés selon la période de construction - Données-échantillon (25 %)"]
) * 100

del df_final["Logement occupé selon période de construction - 1960 ou avant"]

# 8. Logement occupé selon période de construction - 1961 à 1980 (%)
df_final["Logement occupé selon période de construction - 1961 à 1980 (%)"] = (
    df_final["Logement occupé selon période de construction - 1961 à 1980"] /
    df_final["Total - Logements privés occupés selon la période de construction - Données-échantillon (25 %)"]
) * 100

del df_final["Logement occupé selon période de construction - 1961 à 1980"]

# 9. Logement occupé selon période de construction - 1981 à 1990 (%)
df_final["Logement occupé selon période de construction - 1981 à 1990 (%)"] = (
    df_final["Logement occupé selon période de construction - 1981 à 1990"] /
    df_final["Total - Logements privés occupés selon la période de construction - Données-échantillon (25 %)"]
) * 100

del df_final["Logement occupé selon période de construction - 1981 à 1990"]

# 10. Logement occupé selon période de construction - 1991 à 2000 (%)
df_final["Logement occupé selon période de construction - 1991 à 2000 (%)"] = (
    df_final["Logement occupé selon période de construction - 1991 à 2000"] /
    df_final["Total - Logements privés occupés selon la période de construction - Données-échantillon (25 %)"]
) * 100

del df_final["Logement occupé selon période de construction - 1991 à 2000"]

# 11. Logement occupé selon période de construction - 2001 à 2005 (%)
df_final["Logement occupé selon période de construction - 2001 à 2005 (%)"] = (
    df_final["Logement occupé selon période de construction - 2001 à 2005"] /
    df_final["Total - Logements privés occupés selon la période de construction - Données-échantillon (25 %)"]
) * 100

del df_final["Logement occupé selon période de construction - 2001 à 2005"]

# 12. Logement occupé selon période de construction - 2006 à 2010 (%)
df_final["Logement occupé selon période de construction - 2006 à 2010 (%)"] = (
    df_final["Logement occupé selon période de construction - 2006 à 2010"] /
    df_final["Total - Logements privés occupés selon la période de construction - Données-échantillon (25 %)"]
) * 100

del df_final["Logement occupé selon période de construction - 2006 à 2010"]

# 13. Logement occupé selon période de construction - 2011
df_final["Logement occupé selon période de construction - 2011 à 2015 (%)"] = (
    df_final["Logement occupé selon période de construction - 2011 à 2015"] /
```

```
    df_final["Total - Logements privés occupés selon la période de construction - Données-échantillon (25 %)"]
) * 100

del df_final["Logement occupé selon période de construction - 2011 à 2015"]

# 14. Logement occupé selon période de construction - 2016 à 2021 (%)
df_final["Logement occupé selon période de construction - 2016 à 2021 (%)"] = (
    df_final["Logement occupé selon période de construction - 2016 à 2021"] /
    df_final["Total - Logements privés occupés selon la période de construction - Données-échantillon (25 %)"]
) * 100

del df_final["Logement occupé selon période de construction - 2016 à 2021"]
del df_final["Total - Logements privés occupés selon la période de construction - Données-échantillon (25 %)"]

# 15. Logement occupé selon l'état du logement - Réparations majeures requises (%)
df_final["Logement occupé selon l'état du logement - Réparations majeures requises (%)"] = (
    df_final["Logement occupé selon l'état du logement - Réparations majeures requises"] /
    df_final["Total - Logements privés occupés selon l'état du logement - Données-échantillon (25 %)"]
) * 100

del df_final["Logement occupé selon l'état du logement - Réparations majeures requises"]
del df_final["Total - Logements privés occupés selon l'état du logement - Données-échantillon (25 %)"]

# 16. Ménages privés selon le mode d'occupation - Propriétaire (%)
df_final["Ménages privés selon le mode d'occupation - Propriétaire (%)"] = (
    df_final["Ménages privés selon le mode d'occupation - Propriétaire"] /
    df_final["Total - Ménages privés selon le mode d'occupation - Données-échantillon (25 %)"]
) * 100

del df_final["Ménages privés selon le mode d'occupation - Propriétaire"]

# 17. Ménages privés selon le mode d'occupation - Locataire (%)
df_final["Ménages privés selon le mode d'occupation - Locataire (%)"] = (
    df_final["Ménages privés selon le mode d'occupation - Locataire"] /
    df_final["Total - Ménages privés selon le mode d'occupation - Données-échantillon (25 %)"]
) * 100

del df_final["Ménages privés selon le mode d'occupation - Locataire"]

# 18. Ménages privés selon le mode d'occupation - Logement fourni par le gouvernement local (%)
df_final["Ménages privés selon le mode d'occupation - Logement fourni par le gouvernement local (%)"] = (
    df_final["Ménages privés selon le mode d'occupation - Logement fourni par le gouvernement local"] /
    df_final["Total - Ménages privés selon le mode d'occupation - Données-échantillon (25 %)"]
) * 100

del df_final["Ménages privés selon le mode d'occupation - Logement fourni par le gouvernement local"]
del df_final["Total - Ménages privés selon le mode d'occupation - Données-échantillon (25 %)"]

# 19. Plus haut diplôme - Aucun certificat, diplôme ou grade (%)
df_final["Plus haut diplôme - Aucun certificat, diplôme ou grade (%)"] = (
    df_final["Plus haut diplôme - Aucun certificat, diplôme ou grade"] /
    df_final["Total - Plus haut certificat, diplôme ou grade pour la population âgée de 25 à 64 ans dans les ménages privés - Données-éc
) * 100

del df_final["Plus haut diplôme - Aucun certificat, diplôme ou grade"]

# 20. Plus haut diplôme - Diplôme d'études secondaires ou attestation d'équivalence (%)
df_final["Plus haut diplôme - Diplôme d'études secondaires ou attestation d'équivalence (%)"] = (
    df_final["Plus haut diplôme - Diplôme d'études secondaires ou attestation d'équivalence"] /
    df_final["Total - Plus haut certificat, diplôme ou grade pour la population âgée de 25 à 64 ans dans les ménages privés - Données-éc
) * 100

del df_final["Plus haut diplôme - Diplôme d'études secondaires ou attestation d'équivalence"]

# 21. Plus haut diplôme - Certificat, diplôme ou grade d’études postsecondaires (%)
df_final["Plus haut diplôme - Certificat, diplôme ou grade d’études postsecondaires (%)"] = (
    df_final["Plus haut diplôme - Certificat, diplôme ou grade d’études postsecondaires"] /
    df_final["Total - Plus haut certificat, diplôme ou grade pour la population âgée de 25 à 64 ans dans les ménages privés - Données-éc
) * 100

del df_final["Plus haut diplôme - Certificat, diplôme ou grade d’études postsecondaires"]
del df_final["Total - Plus haut certificat, diplôme ou grade pour la population âgée de 25 à 64 ans dans les ménages privés - Données-éc

# 22. Groupes d'âge de la population - 65 ans et plus (%)
df_final["Groupes d'âge de la population - 65 ans et plus (%)"] = (
    df_final["Groupes d'âge de la population - 65 ans et plus"] /
    df_final["Total - Groupes d'âge de la population - Données intégrales (100 %)"]
) * 100

del df_final["Groupes d'âge de la population - 65 ans et plus"]
del df_final["Total - Groupes d'âge de la population - Données intégrales (100 %)"]
```

```
# 23. Ménages dont 30 % ou plus du revenu est consacré aux frais de logement seulement (%)
df_final["Ménages dont 30 % ou plus du revenu est consacré aux frais de logement seulement (%)"] = (
    df_final["Ménages dont 30 % ou plus du revenu est consacré aux frais de logement seulement"] /
    df_final["Total - Ménages propriétaires et locataires dont le revenu total du ménage est supérieur à zéro, dans les logements privés
) * 100

del df_final["Ménages dont 30 % ou plus du revenu est consacré aux frais de logement seulement"]
del df_final["Total - Ménages propriétaires et locataires dont le revenu total du ménage est supérieur à zéro, dans les logements privés

# Affichage du DataFrame mis à jour
df_final.head()
```

	code_geo	nom_geo	Logements privés occupés par des résidents habituels	Nombre moyen de pièces par logement	Population, 2016	Population, 2021	Revenu total médian après impôts en 2020 parmi les bénéficiaires d'un revenu total ($)	Revenu total médian avant impôts en 2020 parmi les bénéficiaires d'un revenu total ($)	Revenu total médian des ménages en 2020($) après impôts	Revenu total médian des ménages en 2020($) avant impôts	Superficie des terres en kilomètres carrés	m
0	2401023	Les Îles-de-la-Madeleine	5610.00	6.20	12010.00	12190.00	35200.00	39600.00	65500.00	75000.00	155.06	
1	2401042	Grosse-Île	220.00	6.40	465.00	464.00	34400.00	39600.00	75000.00	86000.00	32.28	
2	2402005	Percé	1550.00	6.00	3103.00	3095.00	31000.00	34400.00	51200.00	56400.00	432.81	
3	2402010	Sainte-Thérèse-de-Gaspé	475.00	6.00	1015.00	979.00	33200.00	37200.00	58400.00	65000.00	34.45	
4	2402015	Grande-Rivière	1643.00	5.80	3408.00	3384.00	31400.00	35600.00	54000.00	60400.00	87.92	

```
# Affichage du DataFrame mis à jour
df_final.head(20)
```

	code_geo	nom_geo	Logements privés occupés par des résidents habituels	Nombre moyen de pièces par logement	Population, 2016	Population, 2021	Revenu total médian après impôts en 2020 parmi les bénéficiaires d'un revenu total ($)	Revenu total médian avant impôts en 2020 parmi les bénéficiaires d'un revenu total ($)	Revenu total médian des ménages en 2020($) après impôts	Revenu total médian des ménages en 2020($) avant impôts	Superfi des ter kilomèt car
0	2401023	Les Îles-de-la-Madeleine	5610.00	6.20	12010.00	12190.00	35200.00	39600.00	65500.00	75000.00	155
1	2401042	Grosse-Île	220.00	6.40	465.00	464.00	34400.00	39600.00	75000.00	86000.00	32
2	2402005	Percé	1550.00	6.00	3103.00	3095.00	31000.00	34400.00	51200.00	56400.00	432
3	2402010	Sainte-Thérèse-de-Gaspé	475.00	6.00	1015.00	979.00	33200.00	37200.00	58400.00	65000.00	34
4	2402015	Grande-Rivière	1643.00	5.80	3408.00	3384.00	31400.00	35600.00	54000.00	60400.00	87
5	2402028	Chandler	3572.00	5.90	7546.00	7490.00	31600.00	35200.00	54400.00	60000.00	418
6	2402047	Port-Daniel--Gascons	1079.00	6.20	2210.00	2271.00	30200.00	33200.00	53200.00	59200.00	300
7	2402902	Mont-Alexandre	0.00	NaN	0.00	0.00	NaN	NaN	NaN	NaN	1800
8	2403005	Gaspé	6780.00	6.30	14568.00	15063.00	36000.00	41200.00	63200.00	72000.00	1118
9	2403010	Cloridorme	315.00	6.50	671.00	607.00	29200.00	32000.00	50000.00	55200.00	158
10	2403015	Petite-Vallée	82.00	7.50	170.00	157.00	NaN	NaN	NaN	NaN	39
11	2403020	Grande-Vallée	530.00	6.10	1057.00	1077.00	31800.00	34800.00	55600.00	60400.00	144
12	2403025	Murdochville	327.00	6.70	651.00	643.00	33200.00	37200.00	52000.00	57600.00	60
13	2403902	Rivière-Saint-Jean	0.00	NaN	0.00	0.00	NaN	NaN	NaN	NaN	1748
14	2403904	Collines-du-Basque	0.00	NaN	0.00	0.00	NaN	NaN	NaN	NaN	817
15	2404005	Sainte-Madeleine-de-la-Rivière-Madeleine	161.00	6.50	289.00	289.00	23400.00	23800.00	39600.00	40800.00	262
16	2404010	Saint-Maxime-du-Mont-Louis	501.00	6.00	1134.00	1047.00	28200.00	30600.00	48400.00	53200.00	232
17	2404015	Mont-Saint-Pierre	98.00	6.30	155.00	186.00	NaN	NaN	NaN	NaN	52
18	2404020	Rivière-à-Claude	76.00	6.30	128.00	141.00	NaN	NaN	NaN	NaN	155
19	2404025	Marsoui	149.00	6.20	275.00	289.00	27400.00	28800.00	46000.00	50400.00	181

b.CENSUS 2016

```
# importer CENSUS DATA 2016

df2 = pd.read_csv(r'/content/gdrive/MyDrive/mémoire/Données/Census 2016/clean_stat_can_2016_clean_utf.csv',
                  encoding = "utf-8" )

# Rename columns to match 2021 data
df2 = df2.rename(columns={'CODE_GÉO_ALT':'code_geo', 'NIVEAU_GÉO' : 'type_geo', 'NOM_GÉO' : 'nom_geo', 'DIM: Profil des aires de diffusi
})
```

```
<ipython-input-35-ca695eb64185>:5: DtypeWarning: Columns (0) have mixed types. Specify dtype option on import or set low_memory=Fals
  df2 = pd.read_csv(r'/content/gdrive/MyDrive/mémoire/Données/Census 2016/clean_stat_can_2016_clean_utf.csv',
```

```
len(df2)
```

```
33801621
```

```
df2.head()
```

	nom_geo	code_geo	nom_car	total
0	Canada	1	Population, 2016	35151728
1	Canada	1	Population, 2011	33476688
2	Canada	1	Variation en pourcentage de la population, 201...	5.0
3	Canada	1	Total des logements privés	15412443
4	Canada	1	Logements privés occupés par des résidents hab...	14072079

```
##garder seulement les lignes où le code géographique est composé de 7 chiffres (les municipalités)

df2_filtered = df2[df2['code_geo'].astype(str).str.len() == 7]


df2_filtered.head(250)
```

Afficher la sortie masquée

```
# nom_géo qui ont différents code_géo

# Groupe les lignes par 'nom_geo' et agrège les valeurs uniques de 'code_geo' dans une liste
grouped = df2_filtered.groupby('nom_geo')['code_geo'].agg(set).reset_index()

# Filtre les noms_geo avec plus d'un code_geo unique
multiple_codes = grouped[grouped['code_geo'].apply(len) > 1]

# Affiche le résultat
print(multiple_codes)
print(len(multiple_codes))
```

Afficher la sortie masquée

```
# Cette fonction stocke les changements réalisés
changes = []

def log_change(old_value, new_value):
    changes.append({'Ancien_nom_car': old_value, 'Nouveau_nom_car': new_value})

# Code modifié pour enregistrer les changements de nom_car

### Type de logement
start_indices = df2_filtered[df2_filtered["nom_car"] == "Total - Logements privés occupés par type de construction résidentielle - Donne
row_names = ["Type de construction résidentielle - Maison individuelle non attenante",
             "Type de construction résidentielle - Appartement dans un immeuble de cinq étages ou plus"
            ]

for index in start_indices:
    for i in range(2):
        old_value = df2_filtered.at[index + i + 1, "nom_car"]
        new_value = row_names[i]
        if old_value != new_value:  # To ensure that we only log actual changes
            df2_filtered.at[index + i + 1, "nom_car"] = new_value
            log_change(old_value, new_value)

start_indices = df2_filtered[df2_filtered["nom_car"] == "Total - Logements privés occupés par type de construction résidentielle - Donne
row_names = [ "Type de construction résidentielle - Maison jumelée",
              "Type de construction résidentielle - Maison en rangée",
              "Type de construction résidentielle - Appartement ou plain-pied dans un duplex",
              "Type de construction résidentielle - Appartement dans un immeuble de moins de cinq étages",
              ]
for index in start_indices:
    for i in range(4):
        old_value = df2_filtered.at[index + i + 4, "nom_car"]
        new_value = row_names[i]
        if old_value != new_value:  # To ensure that we only log actual changes
            df2_filtered.at[index + i + 4, "nom_car"] = new_value
            log_change(old_value, new_value)

### Revenu des ménages

### Revenu des ménages
start_indices = df2_filtered[df2_filtered["nom_car"] == "Total - Statistiques du revenu en 2015 pour les ménages privés selon la taille
row_names = ["Revenu total médian des ménages en 2015($) avant impôts",
             "Revenu total médian des ménages en 2015($) après impôts"]
```

```
for index in start_indices:
    for i in range(2):
        old_value = df2_filtered.at[index + i + 1, "nom_car"]
        new_value = row_names[i]
        if old_value != new_value:
            df2_filtered.at[index + i + 1, "nom_car"] = new_value
            log_change(old_value, new_value)

start_indices = df2_filtered[df2_filtered["nom_car"] == "Total - Statistiques du revenu en 2015 pour la population âgée de 15 ans et plu
row_names = ["Revenu total médian avant impôts en 2015 parmi les bénéficiaires d'un revenu total ($)"]

for index in start_indices:
    for i in range(1):
        old_value = df2_filtered.at[index + i + 2, "nom_car"]
        new_value = row_names[i]
        if old_value != new_value:
            df2_filtered.at[index + i + 2, "nom_car"] = new_value
            log_change(old_value, new_value)

start_indices = df2_filtered[df2_filtered["nom_car"] == "Total - Statistiques du revenu en 2015 pour la population âgée de 15 ans et plu
row_names = [
             "Revenu total médian après impôts en 2015 parmi les bénéficiaires d'un revenu total ($)"]

for index in start_indices:
    for i in range(1):
        old_value = df2_filtered.at[index + i + 4, "nom_car"]
        new_value = row_names[i]
        if old_value != new_value:
            df2_filtered.at[index + i + 4, "nom_car"] = new_value
            log_change(old_value, new_value)

### Période de construction
start_indices = df2_filtered[df2_filtered["nom_car"] == "Total - Logements privés occupés selon la période de construction - Données-éch
row_names = [

             "Logement occupé selon période de construction - 1960 ou avant",
             "Logement occupé selon période de construction - 1961 à 1980",
             "Logement occupé selon période de construction - 1981 à 1990",
             "Logement occupé selon période de construction - 1991 à 2000",
             "Logement occupé selon période de construction - 2001 à 2005",
             "Logement occupé selon période de construction - 2006 à 2010",
             "Logement occupé selon période de construction - 2011 à 2016"
             ]
for index in start_indices:
    for i in range(7):
        old_value = df2_filtered.at[index + i + 1, "nom_car"]
        new_value = row_names[i]
        if old_value != new_value:
            df2_filtered.at[index + i + 1, "nom_car"] = new_value
            log_change(old_value, new_value)

### Réparations majeures requises
start_indices = df2_filtered[df2_filtered["nom_car"] == "Total - Logements privés occupés selon l'état du logement - Données-échantillor
row_names = ["Logement occupé selon l'état du logement - Réparations majeures requises"
             ]
for index in start_indices:
    for i in range(1):
        old_value = df2_filtered.at[index + i + 2, "nom_car"]
        new_value = row_names[i]
        if old_value != new_value:
            df2_filtered.at[index + i + 2, "nom_car"] = new_value
            log_change(old_value, new_value)

### Mode d'occupation
start_indices = df2_filtered[df2_filtered["nom_car"] == "Total - Ménages privés selon le mode d'occupation - Données-échantillons (25 %)
row_names = ["Ménages privés selon le mode d'occupation - Propriétaire",
             "Ménages privés selon le mode d'occupation - Locataire",
             "Ménages privés selon le mode d'occupation - Logement fourni par le gouvernement local"
             ]
for index in start_indices:
    for i in range(3):
        old_value = df2_filtered.at[index + i + 1, "nom_car"]
        new_value = row_names[i]
        if old_value != new_value:
            df2_filtered.at[index + i + 1, "nom_car"] = new_value
            log_change(old_value, new_value)

### Education
start_indices = df2_filtered[df2_filtered["nom_car"] == "Total - Plus haut certificat, diplôme ou grade pour la population âgée de 25 à
row_names = ["Plus haut diplôme - Aucun certificat, diplôme ou grade",
             "Plus haut diplôme - Diplôme d'études secondaires ou attestation d'équivalence",
```

```
            "Plus haut diplôme - Certificat, diplôme ou grade d’études postsecondaires"
            ]
for index in start_indices:
    for i in range(3):
        old_value = df2_filtered.at[index + i + 1, "nom_car"]
        new_value = row_names[i]
        if old_value != new_value:
            df2_filtered.at[index + i + 1, "nom_car"] = new_value
            log_change(old_value, new_value)

### Groupes d'âge
start_indices = df2_filtered[df2_filtered["nom_car"] == "Total - Répartition (%) de la population par grands groupes d'âge - Données int
row_names = ["Groupes d'âge de la population - 65 ans et plus"]

for index in start_indices:
    for i in range(1):
        old_value = df2_filtered.at[index + i + 3, "nom_car"]
        new_value = row_names[i]
        if old_value != new_value:
            df2_filtered.at[index + i + 3, "nom_car"] = new_value
            log_change(old_value, new_value)

### Frais de logements
start_indices = df2_filtered[df2_filtered["nom_car"] == "Total - Ménages propriétaires et locataires dont le revenu total du ménage est
row_names = ["Ménages dont 30 % ou plus du revenu est consacré aux frais de logement seulement"]

for index in start_indices:
    for i in range(1):
        old_value = df2_filtered.at[index + i + 2, "nom_car"]
        new_value = row_names[i]
        if old_value != new_value:
            df2_filtered.at[index + i + 2, "nom_car"] = new_value
            log_change(old_value, new_value)

# Crée une DataFrame pour les changements et afficher les valeurs uniques
change_df = pd.DataFrame(changes)
unique_changes = change_df.drop_duplicates()
print(unique_changes)
```

```
                                        Ancien_nom_car  \
0                       Maison individuelle non attenante
1      Appartement dans un immeuble de cinq étages ou...
2570                                     Maison jumelée
2571                                   Maison en rangée
2572             Appartement ou plain-pied dans un duplex
2573   Appartement dans un immeuble de moins de cinq ...
7710          Revenu total médian des ménages en 2015 ($)
7711   Revenu après impôt médian des ménages en 2015 ($)
10280  Revenu total médian en 2015 parmi les bénéfici...
11565  Revenu après impôt médian en 2015 parmi les bé...
12850                                       1960 ou avant
12851                                         1961 à 1980
12852                                         1981 à 1990
12853                                         1991 à 2000
12854                                         2001 à 2005
12855                                         2006 à 2010
12856                                         2011 à 2016
21845                      Réparations majeures requises
23130                                       Propriétaire
23131                                          Locataire
23132                                  Logement de bande
26985                  Aucun certificat, diplôme ou grade
26986  Diplôme d'études secondaires ou attestation d'...
26987  Certificat, diplôme ou grade d'études postseco...
30840                                     65 ans et plus
32125  30 % ou plus du revenu est consacré aux frais ...

                                       Nouveau_nom_car
0      Type de construction résidentielle - Maison in...
1      Type de construction résidentielle - Apparteme...
2570   Type de construction résidentielle - Maison ju...
2571   Type de construction résidentielle - Maison en...
2572   Type de construction résidentielle - Apparteme...
2573   Type de construction résidentielle - Apparteme...
7710   Revenu total médian des ménages en 2015($) ava...
7711   Revenu total médian des ménages en 2015($) apr...
10280  Revenu total médian avant impôts en 2015 parmi...
11565  Revenu total médian après impôts en 2015 parmi...
12850  Logement occupé selon période de construction ...
12851  Logement occupé selon période de construction ...
12852  Logement occupé selon période de construction ...
12853  Logement occupé selon période de construction ...
12854  Logement occupé selon période de construction ...
12855  Logement occupé selon période de construction ...
12856  Logement occupé selon période de construction ...
21845  Logement occupé selon l'état du logement - Rép...
```

```
23130  Ménages privés selon le mode d'occupation - Pr...
23131  Ménages privés selon le mode d'occupation - Lo...
23132  Ménages privés selon le mode d'occupation - Lo...
26985  Plus haut diplôme - Aucun certificat, diplôme ...
26986  Plus haut diplôme - Diplôme d'études secondair...
26987  Plus haut diplôme - Certificat, diplôme ou gra...
30840    Groupes d'âge de la population - 65 ans et plus
32125  Ménages dont 30 % ou plus du revenu est consac...
```

```
vars = [
    "Revenu total médian après impôts en 2015 parmi les bénéficiaires d'un revenu total ($)",
    "Revenu total médian avant impôts en 2015 parmi les bénéficiaires d'un revenu total ($)",
    "Population, 2011",
    "Population, 2016",
    "Superficie des terres en kilomètres carrés",
    "Taille moyenne des ménages privés",
    "Nombre moyen de pièces par logement",
    "Logements privés occupés par des résidents habituels",
    "Total des logements privés",
    "Taux de chômage",
    "Revenu total médian des ménages en 2015($) avant impôts",
    "Revenu total médian des ménages en 2015($) après impôts",

    #variables pour calculer mes taux

    "Total - Logements privés occupés par type de construction résidentielle - Données intégrales (100 %)",

    "Total - Logements privés occupés selon la période de construction - Données-échantillons (25 %)",

    "Total - Logements privés occupés selon l'état du logement - Données-échantillons (25 %)",

    "Total - Ménages privés selon le mode d'occupation - Données-échantillons (25 %)",

    "Total - Plus haut certificat, diplôme ou grade pour la population âgée de 25 à 64 ans dans les ménages privés - Données-échantillor

    "Total - Groupes d'âge et âge moyen de la population - Données intégrales (100 %)",

    "Total - Ménages propriétaires et locataires dont le revenu total du ménage est supérieur à zéro, dans les logements privés non agri

]

rate_vars = rate_vars = [
    "Type de construction résidentielle - Maison individuelle non attenante",
    "Type de construction résidentielle - Maison jumelée",
    "Type de construction résidentielle - Maison en rangée",
    "Type de construction résidentielle - Appartement ou plain-pied dans un duplex",
    "Type de construction résidentielle - Appartement dans un immeuble de moins de cinq étages",
    "Type de construction résidentielle - Appartement dans un immeuble de cinq étages ou plus",

    "Logement occupé selon période de construction - 1960 ou avant",
    "Logement occupé selon période de construction - 1961 à 1980",
    "Logement occupé selon période de construction - 1981 à 1990",
    "Logement occupé selon période de construction - 1991 à 2000",
    "Logement occupé selon période de construction - 2001 à 2005",
    "Logement occupé selon période de construction - 2006 à 2010",
    "Logement occupé selon période de construction - 2011 à 2016",

    "Logement occupé selon l'état du logement - Réparations majeures requises",

    "Ménages privés selon le mode d'occupation - Propriétaire",
    "Ménages privés selon le mode d'occupation - Locataire",
    "Ménages privés selon le mode d'occupation - Logement fourni par le gouvernement local",

    "Plus haut diplôme - Aucun certificat, diplôme ou grade",
    "Plus haut diplôme - Diplôme d'études secondaires ou attestation d'équivalence",
    "Plus haut diplôme - Certificat, diplôme ou grade d’études postsecondaires",

    "Groupes d'âge de la population - 65 ans et plus",

    "Ménages dont 30 % ou plus du revenu est consacré aux frais de logement seulement"

]
```

```
# garder lignes ou nom_car = var + rate_var

all_vars = vars + rate_vars
df2_filtered = df2_filtered[df2_filtered['nom_car'].isin(all_vars)]
```

```
df2_filtered['total'] = pd.to_numeric(df2_filtered['total'], errors='coerce')

# Créer une liste de toutes les municipalités uniques avec leur code géographique
unique_municipalities = df2_filtered[['code_geo', 'nom_geo']].drop_duplicates()

# Transformer le DataFrame pour le fusionner avec les variables cibles
# Ici, nous utilisons le pivot pour réorganiser les données
df2_pivoted = df2_filtered.pivot_table(index=['code_geo', 'nom_geo'],
                                       columns='nom_car',
                                       values='total',
                                       fill_value=np.nan).reset_index()

# Maintenant, fusionner les municipalités uniques avec le DataFrame pivoté
df2_final = unique_municipalities.merge(df2_pivoted, on=['code_geo', 'nom_geo'], how='left')
```

```
<ipython-input-44-a28ebde5e847>:2: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus
  df2_filtered['total'] = pd.to_numeric(df2_filtered['total'], errors='coerce')
```

```
df2_final.head()
```

	code_geo	nom_geo	Groupes d'âge de la population - 65 ans et plus	Logement occupé selon l'état du logement - Réparations majeures requises	Logement occupé selon période de construction - 1960 ou avant	Logement occupé selon période de construction - 1961 à 1980	Logement occupé selon période de construction - 1981 à 1990	Logement occupé selon période de construction - 1991 à 2000	Logement occupé selon période de construction - 2001 à 2005	Logement occupé selon période de construction - 2006 à 2010
0	2401023	Les Îles-de-la-Madeleine	24.40	355.00	1285.00	1830.00	965.00	485.00	220.00	310.00
1	2401042	Grosse-Île	18.30	40.00	75.00	95.00	20.00	15.00	10.00	0.00
2	2402005	Percé	28.30	210.00	650.00	455.00	170.00	135.00	40.00	45.00
3	2402010	Sainte-Thérèse-de-Gaspé	23.60	45.00	135.00	210.00	60.00	20.00	10.00	20.00
4	2402015	Grande-Rivière	23.60	145.00	500.00	620.00	220.00	105.00	35.00	75.00

```
# je me demande combien de valeurs manquantes et non manquantes il y a dans chacune de mes colonnes

# Count missing and non-missing values for each column
missing_counts = df2_filtered.isnull().sum()
non_missing_counts = df2_filtered.notnull().sum()

# Create a DataFrame to display the results
result_df = pd.DataFrame({
    'Missing Values': missing_counts,
    'Non-Missing Values': non_missing_counts
})

print(result_df)
```

```
           Missing Values  Non-Missing Values
nom_geo                 0               52685
code_geo                0               52685
nom_car                 0               52685
total                4490               48195
```

```
df2_final.head()
```

	code_geo	nom_geo	Groupes d'âge de la population - 65 ans et plus	Logement occupé selon l'état du logement - Réparations majeures requises	Logement occupé selon période de construction - 1960 ou avant	Logement occupé selon période de construction - 1961 à 1980	Logement occupé selon période de construction - 1981 à 1990	Logement occupé selon période de construction - 1991 à 2000	Logement occupé selon période de construction - 2001 à 2005	Logement occupé selon période de construction - 2006 à 2010
0	2401023	Les Îles-de-la-Madeleine	24.40	355.00	1285.00	1830.00	965.00	485.00	220.00	310.00
1	2401042	Grosse-Île	18.30	40.00	75.00	95.00	20.00	15.00	10.00	0.00
2	2402005	Percé	28.30	210.00	650.00	455.00	170.00	135.00	40.00	45.00
3	2402010	Sainte-Thérèse-de-Gaspé	23.60	45.00	135.00	210.00	60.00	20.00	10.00	20.00
4	2402015	Grande-Rivière	23.60	145.00	500.00	620.00	220.00	105.00	35.00	75.00

```
weird_var = set(all_vars) - set(df2_filtered['nom_car'].unique())

weird_var
```

```
set()
```

```
len(df2_final.columns)
```

```
43
```

```
# pourcentages de lignes qui ont 40 valeurs nulles, puis 39, puis 38, etc. jusqu'à 0

# Calculer le nombre de valeurs nulles par ligne
null_counts = df2_final.isnull().sum(axis=1)

# Compter le nombre de lignes avec chaque nombre de valeurs nulles
null_counts_frequency = null_counts.value_counts().sort_index(ascending=False)

# Calculer le pourcentage de lignes pour chaque nombre de valeurs nulles
null_counts_percentage = (null_counts_frequency / len(df2_final)) * 100

# Afficher les pourcentages
print(null_counts_percentage)
```

```
40    0.31
36    8.48
26    0.08
24    0.16
16    0.08
6     0.08
4     5.76
2     0.47
1     0.16
0    84.44
Name: count, dtype: float64
```

```
# Calcul des pourcentages pour chaque colonne de `rate_vars`

# 1. Type de construction résidentielle - Maison individuelle non attenante (%)
df2_final["Type de construction résidentielle - Maison individuelle non attenante (%)"] = (
    df2_final["Type de construction résidentielle - Maison individuelle non attenante"] /
    df2_final["Total - Logements privés occupés par type de construction résidentielle - Données intégrales (100 %)"]
) * 100

del df2_final["Type de construction résidentielle - Maison individuelle non attenante"]

# 2. Type de construction résidentielle - Maison jumelée (%)
df2_final["Type de construction résidentielle - Maison jumelée (%)"] = (
    df2_final["Type de construction résidentielle - Maison jumelée"] /
    df2_final["Total - Logements privés occupés par type de construction résidentielle - Données intégrales (100 %)"]
```

```
) * 100

del df2_final["Type de construction résidentielle - Maison jumelée"]

# 3. Type de construction résidentielle - Maison en rangée (%)
df2_final["Type de construction résidentielle - Maison en rangée (%)"] = (
    df2_final["Type de construction résidentielle - Maison en rangée"] /
    df2_final["Total - Logements privés occupés par type de construction résidentielle - Données intégrales (100 %)"]
) * 100

del df2_final["Type de construction résidentielle - Maison en rangée"]

# 4. Type de construction résidentielle - Appartement ou plain-pied dans un duplex (%)
df2_final["Type de construction résidentielle - Appartement ou plain-pied dans un duplex (%)"] = (
    df2_final["Type de construction résidentielle - Appartement ou plain-pied dans un duplex"] /
    df2_final["Total - Logements privés occupés par type de construction résidentielle - Données intégrales (100 %)"]
) * 100

del df2_final["Type de construction résidentielle - Appartement ou plain-pied dans un duplex"]

# 5. Type de construction résidentielle - Appartement dans un immeuble de moins de cinq étages (%)
df2_final["Type de construction résidentielle - Appartement dans un immeuble de moins de cinq étages (%)"] = (
    df2_final["Type de construction résidentielle - Appartement dans un immeuble de moins de cinq étages"] /
    df2_final["Total - Logements privés occupés par type de construction résidentielle - Données intégrales (100 %)"]
) * 100

del df2_final["Type de construction résidentielle - Appartement dans un immeuble de moins de cinq étages"]

# 6. Type de construction résidentielle - Appartement dans un immeuble de cinq étages ou plus (%)
df2_final["Type de construction résidentielle - Appartement dans un immeuble de cinq étages ou plus (%)"] = (
    df2_final["Type de construction résidentielle - Appartement dans un immeuble de cinq étages ou plus"] /
    df2_final["Total - Logements privés occupés par type de construction résidentielle - Données intégrales (100 %)"]
) * 100

del df2_final["Type de construction résidentielle - Appartement dans un immeuble de cinq étages ou plus"]
del df2_final["Total - Logements privés occupés par type de construction résidentielle - Données intégrales (100 %)"]

# 7. Logement occupé selon période de construction - 1960 ou avant (%)
df2_final["Logement occupé selon période de construction - 1960 ou avant (%)"] = (
    df2_final["Logement occupé selon période de construction - 1960 ou avant"] /
    df2_final["Total - Logements privés occupés selon la période de construction - Données-échantillons (25 %)"]
) * 100

del df2_final["Logement occupé selon période de construction - 1960 ou avant"]

# 8. Logement occupé selon période de construction - 1961 à 1980 (%)
df2_final["Logement occupé selon période de construction - 1961 à 1980 (%)"] = (
    df2_final["Logement occupé selon période de construction - 1961 à 1980"] /
    df2_final["Total - Logements privés occupés selon la période de construction - Données-échantillons (25 %)"]
) * 100

del df2_final["Logement occupé selon période de construction - 1961 à 1980"]

# 9. Logement occupé selon période de construction - 1981 à 1990 (%)
df2_final["Logement occupé selon période de construction - 1981 à 1990 (%)"] = (
    df2_final["Logement occupé selon période de construction - 1981 à 1990"] /
    df2_final["Total - Logements privés occupés selon la période de construction - Données-échantillons (25 %)"]
) * 100

del df2_final["Logement occupé selon période de construction - 1981 à 1990"]

# 10. Logement occupé selon période de construction - 1991 à 2000 (%)
df2_final["Logement occupé selon période de construction - 1991 à 2000 (%)"] = (
    df2_final["Logement occupé selon période de construction - 1991 à 2000"] /
    df2_final["Total - Logements privés occupés selon la période de construction - Données-échantillons (25 %)"]
) * 100


del df2_final["Logement occupé selon période de construction - 1991 à 2000"]

# 11. Logement occupé selon période de construction - 2001 à 2005 (%)
df2_final["Logement occupé selon période de construction - 2001 à 2005 (%)"] = (
    df2_final["Logement occupé selon période de construction - 2001 à 2005"] /
    df2_final["Total - Logements privés occupés selon la période de construction - Données-échantillons (25 %)"]
) * 100

del df2_final["Logement occupé selon période de construction - 2001 à 2005"]

# 12. Logement occupé selon période de construction - 2006 à 2010 (%)
df2_final["Logement occupé selon période de construction - 2006 à 2010 (%)"] = (
    df2_final["Logement occupé selon période de construction - 2006 à 2010"] /
    df2_final["Total - Logements privés occupés selon la période de construction - Données-échantillons (25 %)"]
```

```
) * 100

del df2_final["Logement occupé selon période de construction - 2006 à 2010"]

# 13. Logement occupé selon période de construction - 2011 --> je choisis par souci de nomenclature de passer cette colonne à entre 2011
df2_final["Logement occupé selon période de construction - 2011 à 2015 (%)"] = (
    df2_final["Logement occupé selon période de construction - 2011 à 2016"] /
    df2_final["Total - Logements privés occupés selon la période de construction - Données-échantillons (25 %)"]
) * 100

del df2_final["Logement occupé selon période de construction - 2011 à 2016"]
del df2_final["Total - Logements privés occupés selon la période de construction - Données-échantillons (25 %)"]

# 15. Logement occupé selon l'état du logement - Réparations majeures requises (%)
df2_final["Logement occupé selon l'état du logement - Réparations majeures requises (%)"] = (
    df2_final["Logement occupé selon l'état du logement - Réparations majeures requises"] /
    df2_final["Total - Logements privés occupés selon l'état du logement - Données-échantillons (25 %)"]
) * 100

del df2_final["Logement occupé selon l'état du logement - Réparations majeures requises"]
del df2_final["Total - Logements privés occupés selon l'état du logement - Données-échantillons (25 %)"]

# 16. Ménages privés selon le mode d'occupation - Propriétaire (%)
df2_final["Ménages privés selon le mode d'occupation - Propriétaire (%)"] = (
    df2_final["Ménages privés selon le mode d'occupation - Propriétaire"] /
    df2_final["Total - Ménages privés selon le mode d'occupation - Données-échantillons (25 %)"]
) * 100

del df2_final["Ménages privés selon le mode d'occupation - Propriétaire"]

# 17. Ménages privés selon le mode d'occupation - Locataire (%)
df2_final["Ménages privés selon le mode d'occupation - Locataire (%)"] = (
    df2_final["Ménages privés selon le mode d'occupation - Locataire"] /
    df2_final["Total - Ménages privés selon le mode d'occupation - Données-échantillons (25 %)"]
) * 100

del df2_final["Ménages privés selon le mode d'occupation - Locataire"]

# 18. Ménages privés selon le mode d'occupation - Logement fourni par le gouvernement local (%)
df2_final["Ménages privés selon le mode d'occupation - Logement fourni par le gouvernement local (%)"] = (
    df2_final["Ménages privés selon le mode d'occupation - Logement fourni par le gouvernement local"] /
    df2_final["Total - Ménages privés selon le mode d'occupation - Données-échantillons (25 %)"]
) * 100

del df2_final["Ménages privés selon le mode d'occupation - Logement fourni par le gouvernement local"]
del df2_final["Total - Ménages privés selon le mode d'occupation - Données-échantillons (25 %)"]

# 19. Plus haut diplôme - Aucun certificat, diplôme ou grade (%)
df2_final["Plus haut diplôme - Aucun certificat, diplôme ou grade (%)"] = (
    df2_final["Plus haut diplôme - Aucun certificat, diplôme ou grade"] /
    df2_final["Total - Plus haut certificat, diplôme ou grade pour la population âgée de 25 à 64 ans dans les ménages privés - Données-é
) * 100

del df2_final["Plus haut diplôme - Aucun certificat, diplôme ou grade"]

# 20. Plus haut diplôme - Diplôme d'études secondaires ou attestation d'équivalence (%)
df2_final["Plus haut diplôme - Diplôme d'études secondaires ou attestation d'équivalence (%)"] = (
    df2_final["Plus haut diplôme - Diplôme d'études secondaires ou attestation d'équivalence"] /
    df2_final["Total - Plus haut certificat, diplôme ou grade pour la population âgée de 25 à 64 ans dans les ménages privés - Données-é
) * 100

del df2_final["Plus haut diplôme - Diplôme d'études secondaires ou attestation d'équivalence"]

# 21. Plus haut diplôme - Certificat, diplôme ou grade d’études postsecondaires (%)
df2_final["Plus haut diplôme - Certificat, diplôme ou grade d’études postsecondaires (%)"] = (
    df2_final["Plus haut diplôme - Certificat, diplôme ou grade d’études postsecondaires"] /
    df2_final["Total - Plus haut certificat, diplôme ou grade pour la population âgée de 25 à 64 ans dans les ménages privés - Données-é
) * 100

del df2_final["Plus haut diplôme - Certificat, diplôme ou grade d’études postsecondaires"]
del df2_final["Total - Plus haut certificat, diplôme ou grade pour la population âgée de 25 à 64 ans dans les ménages privés - Données-é

# 22. Groupes d'âge de la population - 65 ans et plus (%)
df2_final["Groupes d'âge de la population - 65 ans et plus (%)"] = (
    df2_final["Groupes d'âge de la population - 65 ans et plus"] /
    df2_final["Total - Groupes d'âge et âge moyen de la population - Données intégrales (100 %)"]
) * 100

del df2_final["Groupes d'âge de la population - 65 ans et plus"]
del df2_final["Total - Groupes d'âge et âge moyen de la population - Données intégrales (100 %)"]

# 23. Ménages dont 30 % ou plus du revenu est consacré aux frais de logement seulement (%)
```

```
df2_final["Ménages dont 30 % ou plus du revenu est consacré aux frais de logement seulement (%)"] = (
    df2_final["Ménages dont 30 % ou plus du revenu est consacré aux frais de logement seulement"] /
    df2_final["Total - Ménages propriétaires et locataires dont le revenu total du ménage est supérieur à zéro, dans les logements prive
) * 100

del df2_final["Ménages dont 30 % ou plus du revenu est consacré aux frais de logement seulement"]
del df2_final["Total - Ménages propriétaires et locataires dont le revenu total du ménage est supérieur à zéro, dans les logements prive

# Affichage du DataFrame mis à jour
df2_final.head()
```

	code_geo	nom_geo	Logements privés occupés par des résidents habituels	Nombre moyen de pièces par logement	Population, 2011	Population, 2016	Revenu total médian après impôts en 2015 parmi les bénéficiaires d'un revenu total ($)	Revenu total médian avant impôts en 2015 parmi les bénéficiaires d'un revenu total ($)	Revenu total médian des ménages en 2015($) après impôts	Revenu total médian des ménages en 2015($) avant impôts	Superficie des terres en kilomètres carrés	m
0	2401023	Les Îles-de-la-Madeleine	5321.00	6.20	12291.00	12010.00	28023.00	31541.00	53794.00	61029.00	172.71	
1	2401042	Grosse-Île	213.00	6.30	490.00	465.00	29024.00	33306.00	68352.00	81152.00	37.59	
2	2402005	Percé	1523.00	6.10	3312.00	3103.00	24073.00	26027.00	40804.00	44826.00	431.37	
3	2402010	Sainte-Thérèse-de-Gaspé	462.00	6.10	1055.00	1015.00	25472.00	27776.00	48640.00	54144.00	34.41	
4	2402015	Grande-Rivière	1610.00	5.70	3456.00	3408.00	23086.00	24709.00	41956.00	45751.00	87.86	

```
# nombre de lignes qui ont 40 valeurs nulles, puis 39, puis 38, etc. jusqu'à 0

# Calculer le nombre de valeurs nulles par ligne
null_counts = df2_final.isnull().sum(axis=1)

# Compter le nombre de lignes avec chaque nombre de valeurs nulles
null_counts_frequency = null_counts.value_counts().sort_index(ascending=False)

# Calculer le pourcentage de lignes pour chaque nombre de valeurs nulles
null_counts_percentage = (null_counts_frequency / len(df2_final)) * 100

# Afficher les pourcentages
print(null_counts_percentage)
```

```
33    0.31
29    8.48
27    0.08
19    0.16
13    0.08
5     0.08
3     5.76
2     0.16
1     2.88
0    82.02
Name: count, dtype: float64
```

```
len(df2_final['nom_geo'].unique())
```

```
1226
```

c. Merge census 2021 et 2016

```
# noms des colonnes qui sont dans df_final et pas df2_final et inversement

# Colonnes dans df_final mais pas dans df2_final
cols_in_df_final_not_in_df2_final = set(df_final.columns) - set(df2_final.columns)
print("Colonnes dans df_final mais pas dans df2_final:\n", cols_in_df_final_not_in_df2_final)

# Colonnes dans df2_final mais pas dans df_final
cols_in_df2_final_not_in_df_final = set(df2_final.columns) - set(df_final.columns)
print("\nColonnes dans df2_final mais pas dans df_final:\n", cols_in_df2_final_not_in_df_final)
```

```
Colonnes dans df_final mais pas dans df2_final:
 {'Population, 2021', 'Logement occupé selon période de construction - 2016 à 2021 (%)', "Revenu total médian après impôts en 2020 p

Colonnes dans df2_final mais pas dans df_final:
 {'Population, 2011', "Revenu total médian après impôts en 2015 parmi les bénéficiaires d'un revenu total ($)", 'Revenu total médian
```

```
# outer merge df_final et df2_final sur code_géo/nom_géo en faisant la moyenne des colonnes si la même est présente dans les deux df, en

merged_df = pd.merge(df_final, df2_final, on=['code_geo', 'nom_geo'], how='outer', suffixes=('_2021', '_2016'))

for col in merged_df.columns:
    if col.endswith('_2021') and col[:-5] + '_2016' in merged_df.columns:
        merged_df[col[:-5]] = merged_df[[col, col[:-5] + '_2016']].mean(axis=1, skipna=True)

        merged_df[col[:-5]].fillna(merged_df[col], inplace=True)
        merged_df[col[:-5]].fillna(merged_df[col[:-5] + '_2016'], inplace=True)

        merged_df.drop([col, col[:-5] + '_2016'], axis=1, inplace=True)
    elif col.endswith('_2021'):
        merged_df.rename(columns={col: col[:-5]}, inplace=True)
    elif col.endswith('_2016'):
        merged_df.rename(columns={col: col[:-5]}, inplace=True)

merged_df.head()
```

Afficher la sortie masquée

```
merged_df.columns
```

```
Index(['code_geo', 'nom_geo', 'Population, 2021',
       'Revenu total médian après impôts en 2020 parmi les bénéficiaires d'un revenu total ($)',
       'Revenu total médian avant impôts en 2020 parmi les bénéficiaires d'un revenu total ($)',
       'Revenu total médian des ménages en 2020($) après impôts',
       'Revenu total médian des ménages en 2020($) avant impôts',
       'Logement occupé selon période de construction - 2016 à 2021 (%)',
       'Population, 2011',
       'Revenu total médian après impôts en 2015 parmi les bénéficiaires d'un revenu total ($)',
       'Revenu total médian avant impôts en 2015 parmi les bénéficiaires d'un revenu total ($)',
       'Revenu total médian des ménages en 2015($) après impôts',
       'Revenu total médian des ménages en 2015($) avant impôts',
       'Logements privés occupés par des résidents habituels',
       'Nombre moyen de pièces par logement', 'Population, 2016',
       'Superficie des terres en kilomètres carrés',
       'Taille moyenne des ménages privés', 'Taux de chômage',
       'Total des logements privés',
       'Type de construction résidentielle - Maison individuelle non attenante (%)',
       'Type de construction résidentielle - Maison jumelée (%)',
       'Type de construction résidentielle - Maison en rangée (%)',
       'Type de construction résidentielle - Appartement ou plain-pied dans un duplex (%)',
       'Type de construction résidentielle - Appartement dans un immeuble de moins de cinq étages (%)',
       'Type de construction résidentielle - Appartement dans un immeuble de cinq étages ou plus (%)',
       'Logement occupé selon période de construction - 1960 ou avant (%)',
       'Logement occupé selon période de construction - 1961 à 1980 (%)',
       'Logement occupé selon période de construction - 1981 à 1990 (%)',
       'Logement occupé selon période de construction - 1991 à 2000 (%)',
       'Logement occupé selon période de construction - 2001 à 2005 (%)',
       'Logement occupé selon période de construction - 2006 à 2010 (%)',
       'Logement occupé selon période de construction - 2011 à 2015 (%)',
       'Logement occupé selon l'état du logement - Réparations majeures requises (%)',
       'Ménages privés selon le mode d'occupation - Propriétaire (%)',
       'Ménages privés selon le mode d'occupation - Locataire (%)',
       'Ménages privés selon le mode d'occupation - Logement fourni par le gouvernement local (%)',
       'Plus haut diplôme - Aucun certificat, diplôme ou grade (%)',
       'Plus haut diplôme - Diplôme d'études secondaires ou attestation d'équivalence (%)',
       'Plus haut diplôme - Certificat, diplôme ou grade d’études postsecondaires (%)',
       'Groupes d'âge de la population - 65 ans et plus (%)',
       'Ménages dont 30 % ou plus du revenu est consacré aux frais de logement seulement (%)'],
      dtype='object')
```

```
# nouvelle colonne "Revenu médian moyen après impôt" qui est la moyenne des colonnes "Revenu total médian après impôts en 2020 parmi les
```

```
merged_df['Revenu médian moyen après impôts'] = merged_df[
    ['Revenu total médian après impôts en 2020 parmi les bénéficiaires d\'un revenu total ($)',
     'Revenu total médian après impôts en 2015 parmi les bénéficiaires d\'un revenu total ($)']
].mean(axis=1)

merged_df['Revenu médian moyen avant impôts'] = merged_df[
    ['Revenu total médian avant impôts en 2020 parmi les bénéficiaires d\'un revenu total ($)',
     'Revenu total médian avant impôts en 2015 parmi les bénéficiaires d\'un revenu total ($)']
].mean(axis=1)

del merged_df['Revenu total médian après impôts en 2020 parmi les bénéficiaires d\'un revenu total ($)']
del merged_df['Revenu total médian après impôts en 2015 parmi les bénéficiaires d\'un revenu total ($)']
del merged_df['Revenu total médian avant impôts en 2020 parmi les bénéficiaires d\'un revenu total ($)']
del merged_df['Revenu total médian avant impôts en 2015 parmi les bénéficiaires d\'un revenu total ($)']

merged_df['Revenu médian des ménages avant impôts'] = merged_df[['Revenu total médian des ménages en 2015($) avant impôts',
                                                                  'Revenu total médian des ménages en 2020($) avant impôts']].mean(axis=1)
merged_df['Revenu médian des ménages après impôts'] = merged_df[['Revenu total médian des ménages en 2015($) après impôts',
                                                                  'Revenu total médian des ménages en 2020($) après impôts']].mean(axis=1)

del merged_df['Revenu total médian des ménages en 2015($) avant impôts']
del merged_df['Revenu total médian des ménages en 2020($) avant impôts']
del merged_df['Revenu total médian des ménages en 2015($) après impôts']
del merged_df['Revenu total médian des ménages en 2020($) après impôts']



merged_df.head(100)
```

	code_geo	nom_geo	Population, 2021	Logement occupé selon période de construction - 2016 à 2021 (%)	Population, 2011	Logements privés occupés par des résidents habituels	Nombre moyen de pièces par logement	Population, 2016	Superficie des terres en kilomètres carrés	Taille moyenne des ménages privés	Taux de chômage
0	2401023	Les Îles-de-la-Madeleine	12190.00	3.58	12291.00	5465.50	6.20	12010.00	163.88	2.15	11.95
1	2401042	Grosse-Île	464.00	8.00	490.00	216.50	6.35	465.00	34.94	2.15	14.90
2	2402005	Percé	3095.00	2.58	3312.00	1536.50	6.05	3103.00	432.09	2.00	20.40
3	2402010	Sainte-Thérèse-de-Gaspé	979.00	2.11	1055.00	468.50	6.05	1015.00	34.43	2.15	9.25
4	2402015	Grande-Rivière	3384.00	2.44	3456.00	1626.50	5.75	3408.00	87.89	2.10	12.50
5	2402028	Chandler	7490.00	0.84	7703.00	3508.50	5.95	7546.00	418.89	2.10	10.50
6	2402047	Port-Daniel--Gascons	2271.00	3.24	2453.00	1030.50	6.25	2210.00	300.95	2.15	13.85
7	2402902	Mont-Alexandre	0.00	NaN	0.00	0.00	NaN	0.00	1801.55	NaN	NaN
8	2403005	Gaspé	15063.00	3.91	15163.00	6596.00	6.30	14568.00	1120.10	2.20	10.35
9	2403010	Cloridorme	607.00	0.00	743.00	325.50	6.55	671.00	159.18	1.95	13.45
10	2403015	Petite-Vallée	157.00	0.00	178.00	82.50	7.70	170.00	39.92	2.00	0.00
11	2403020	Grande-Vallée	1077.00	3.77	1137.00	511.50	6.15	1057.00	144.33	2.05	15.05
12	2403025	Murdochville	643.00	0.00	764.00	320.00	6.80	651.00	61.11	2.00	15.55
13	2403902	Rivière-Saint-Jean	0.00	NaN	0.00	0.00	NaN	0.00	1749.60	NaN	NaN
14	2403904	Collines-du-Basque	0.00	NaN	0.00	0.00	NaN	0.00	819.36	NaN	NaN
15	2404005	Sainte-Madeleine-de-la-Rivière-Madeleine	289.00	0.00	334.00	155.50	6.45	289.00	262.85	1.85	31.00
16	2404010	Saint-Maxime-du-Mont-Louis	1047.00	0.00	1118.00	498.50	6.05	1134.00	233.15	2.10	19.30
17	2404015	Mont-Saint-Pierre	186.00	0.00	192.00	92.50	6.65	155.00	52.23	1.80	26.75
18	2404020	Rivière-à-Claude	141.00	0.00	130.00	73.50	6.65	128.00	156.12	1.80	11.10
19	2404025	Marsoui	289.00	0.00	309.00	142.00	6.25	275.00	181.57	1.95	18.00
20	2404030	La Martre	194.00	0.00	245.00	91.50	6.40	243.00	175.34	2.35	22.75
21	2404037	Sainte-Anne-des-Monts	6121.00	1.98	6933.00	3025.00	6.15	6437.00	263.80	2.05	16.30
22	2404047	Cap-Chat	2516.00	1.63	2623.00	1174.50	6.45	2476.00	181.68	2.05	14.50
23	2404902	Mont-Albert	167.00	0.00	204.00	89.50	5.60	179.00	3466.82	1.95	23.90
24	2404904	Coulée-des-Adolphe	0.00	NaN	0.00	0.00	NaN	0.00	86.41	NaN	NaN
25	2405010	Shigawake	333.00	0.00	338.00	135.50	7.15	292.00	76.82	2.30	12.70
26	2405015	Saint-Godefroi	350.00	5.13	405.00	178.50	6.65	380.00	63.55	2.05	10.25
27	2405020	Hope Town	334.00	0.00	344.00	140.50	6.40	339.00	51.08	2.45	23.90
28	2405025	Hope	584.00	3.33	630.00	276.50	6.60	568.00	70.55	2.05	12.95
29	2405032	Paspébiac	3033.00	2.84	3198.00	1418.50	6.40	3164.00	94.72	2.15	14.40
30	2405040	New Carlisle	1336.00	4.17	1358.00	588.50	6.25	1388.00	67.88	2.15	19.10
31	2405045	Bonaventure	2733.00	4.56	2775.00	1263.00	6.10	2706.00	104.42	2.10	13.35

32	2405050	Saint-Elzéar	464.00	0.00	467.00	207.50	6.50	458.00	203.87	2.15	13.05
33	2405055	Saint-Siméon	1207.00	3.48	1179.00	563.00	6.20	1171.00	56.79	2.10	13.75
34	2405060	Caplan	1966.00	4.89	2039.00	911.00	6.05	2024.00	85.50	2.05	9.85
35	2405065	Saint-Alphonse	711.00	3.17	691.00	311.50	6.05	699.00	112.08	2.30	12.90
36	2405070	New Richmond	3683.00	2.62	3810.00	1689.00	6.10	3706.00	171.92	2.15	10.90
37	2405077	Cascapédia--Saint-Jules	764.00	8.45	741.00	335.50	6.35	730.00	163.87	2.20	26.85
38	2405902	Rivière-Bonaventure	59.00	NaN	25.00	27.50	NaN	35.00	3056.37	1.60	NaN
39	2406005	Maria	2760.00	5.15	2536.00	1119.50	6.55	2615.00	94.98	2.20	7.45
40	2406013	Carleton-sur-Mer	4081.00	4.22	3991.00	1967.00	6.25	4073.00	221.45	2.05	12.95
41	2406020	Nouvelle	1782.00	2.96	1689.00	787.00	6.60	1670.00	233.55	2.15	14.55
42	2406025	Escuminac	575.00	0.00	588.00	254.50	5.95	544.00	108.45	2.25	16.45
43	2406030	Pointe-à-la-Croix	1344.00	1.56	1457.00	647.50	5.70	1399.50	390.85	2.10	14.15
44	2406035	Ristigouche-Partie-Sud-Est	170.00	0.00	167.00	81.50	6.05	171.00	51.73	2.10	30.00
45	2406040	Saint-André-de-Restigouche	154.00	0.00	157.00	76.50	6.45	161.00	143.95	2.05	28.20
46	2406045	Matapédia	566.00	0.00	664.00	310.50	6.20	645.00	71.57	1.90	18.00
47	2406050	Saint-Alexis-de-Matapédia	519.00	0.00	548.00	240.50	6.55	500.00	84.50	2.05	22.45
48	2406055	Saint-François-d'Assise	679.00	3.08	706.00	313.50	6.00	644.00	178.95	2.05	24.20
49	2406060	L'Ascension-de-Patapédia	148.00	0.00	190.00	88.00	6.60	164.00	95.96	1.80	33.35
50	2406802	Gesgapegiag	637.00	10.42	688.00	250.00	6.65	653.00	2.14	2.55	26.35
51	2406804	Listuguj	NaN	NaN	1865.00	579.00	6.40	1377.50	44.25	2.10	32.60
52	2406902	Rivière-Nouvelle	0.00	NaN	0.00	0.00	NaN	0.00	1087.91	NaN	NaN
53	2406904	Ruisseau-Ferguson	0.00	NaN	0.00	0.00	NaN	0.00	677.21	NaN	NaN
54	2407005	Sainte-Marguerite-Marie	183.00	0.00	203.00	85.00	6.45	166.00	85.98	2.10	20.50
55	2407010	Sainte-Florence	367.00	0.00	414.00	189.00	6.25	384.00	103.53	2.00	13.55
56	2407018	Causapscal	2147.00	1.42	2458.00	1061.50	6.20	2304.00	161.61	2.05	14.60
57	2407025	Albertville	239.00	0.00	256.00	117.50	6.05	226.00	103.19	2.00	29.25
58	2407030	Saint-Léon-le-Grand	968.00	6.10	970.00	420.00	6.55	953.00	128.37	2.25	14.15
59	2407035	Saint-Zénon-du-Lac-Humqui	370.00	5.26	366.00	181.00	5.95	359.00	113.25	2.05	13.85
60	2407040	Sainte-Irène	369.00	0.00	341.00	167.00	6.65	327.00	135.17	2.10	12.65
61	2407047	Amqui	5999.00	1.23	6322.00	2854.00	6.20	6178.00	121.09	2.05	8.65
62	2407057	Lac-au-Saumon	1488.00	2.48	1453.00	584.00	6.35	1450.00	81.06	2.25	9.45
63	2407065	Saint-Alexandre-des-Lacs	298.00	0.00	258.00	142.00	6.75	297.50	89.94	2.20	14.95
64	2407070	Saint-Tharcisius	411.00	0.00	464.00	158.50	6.10	368.00	78.97	2.25	18.05
65	2407075	Saint-Vianney	446.00	0.00	477.00	208.00	6.50	441.00	145.80	2.15	7.50

66	2407080	Val-Brillant	899.00	0.00	955.00	417.50	6.40	927.00	77.97	2.20	13.85
67	2407085	Sayabec	1706.00	1.23	1864.00	763.50	6.05	1712.50	130.52	2.15	9.75
68	2407090	Saint-Cléophas	321.00	0.00	334.00	144.00	6.75	333.00	97.33	2.25	13.60
69	2407095	Saint-Moïse	542.00	0.00	577.00	265.50	6.00	580.00	109.87	2.05	10.25
70	2407100	Saint-Noël	392.00	0.00	434.00	184.50	6.30	398.00	45.83	2.10	5.55
71	2407105	Saint-Damase	382.00	0.00	397.00	171.50	6.75	391.00	116.56	2.35	7.15
72	2407902	Routhierville	20.00	NaN	15.00	11.00	NaN	18.00	627.21	NaN	NaN
73	2407904	Rivière-Vaseuse	0.00	NaN	5.00	0.00	NaN	0.00	275.43	NaN	NaN
74	2407906	Rivière-Patapédia-Est	0.00	NaN	0.00	0.00	NaN	0.00	15.91	NaN	NaN
75	2407908	Lac-Casault	30.00	NaN	5.00	25.00	NaN	23.50	1448.59	1.50	NaN
76	2407910	Ruisseau-des-Mineurs	5.00	NaN	0.00	2.00	NaN	0.00	922.63	NaN	NaN
77	2407912	Lac-Alfred	0.00	NaN	0.00	0.00	NaN	0.00	77.53	NaN	NaN
78	2407914	Lac-Matapédia	10.00	NaN	5.00	46.50	5.60	88.50	71.22	2.00	25.00
79	2408005	Les Méchins	995.00	1.89	1107.00	513.50	5.95	987.00	442.39	1.95	17.15
80	2408010	Saint-Jean-de-Cherbourg	163.00	0.00	193.00	55.50	6.50	125.50	114.00	2.15	43.55
81	2408015	Grosses-Roches	375.00	0.00	411.00	185.00	6.25	350.00	63.80	1.80	17.10
82	2408023	Sainte-Félicité	1100.00	0.00	1175.00	502.00	6.40	1076.00	91.06	2.10	11.60
83	2408030	Saint-Adelme	484.00	0.00	485.00	262.00	6.25	606.00	100.61	2.25	14.70
84	2408035	Saint-René-de-Matane	961.00	0.00	1089.00	466.50	6.15	978.00	254.57	2.05	14.65
85	2408040	Sainte-Paule	247.00	0.00	201.00	122.00	6.40	233.00	83.54	1.95	7.70
86	2408053	Matane	13987.00	1.86	14462.00	6931.00	6.00	14311.00	195.79	1.95	8.90
87	2408065	Saint-Léandre	375.00	0.00	402.00	188.50	6.35	400.00	104.87	2.05	15.95
88	2408073	Saint-Ulric	1567.00	0.00	1642.00	768.00	6.20	1585.00	120.55	2.05	9.10
89	2408080	Baie-des-Sables	613.00	0.00	609.00	273.50	6.85	628.00	65.13	2.15	13.85
90	2408902	Rivière-Bonjour	16.00	NaN	10.00	15.00	NaN	21.50	1680.72	1.80	NaN
91	2409005	La Rédemption	384.00	0.00	488.00	183.00	6.20	432.00	116.90	2.20	11.65
92	2409010	Saint-Charles-Garnier	222.00	0.00	271.00	111.50	6.30	240.00	84.78	2.05	13.90
93	2409015	Les Hauteurs	485.00	7.50	524.00	214.00	6.35	504.00	102.98	2.30	15.80
94	2409020	Sainte-Jeanne-d'Arc	217.00	0.00	313.00	120.50	6.90	280.00	110.44	2.10	22.90
95	2409025	Saint-Gabriel-de-Rimouski	1177.00	0.00	1180.00	510.50	6.80	1167.00	127.01	2.25	9.45
96	2409030	Saint-Donat	835.00	2.35	890.00	397.50	6.60	876.00	95.69	2.15	4.95
97	2409035	Sainte-Angèle-de-Mérici	989.00	0.00	999.00	440.50	6.40	953.00	107.80	2.20	5.60
98	2409040	Padoue	250.00	0.00	273.00	110.00	6.70	245.00	66.69	2.25	13.35
99	2409048	Métis-sur-Mer	594.00	3.33	607.00	295.50	6.85	579.50	48.41	1.95	17.85

```
# duo nom_géo/code_géo présents dans df_final et pas df2_final et inversement

# Municipalités dans df_final mais pas dans df2_final
code_geo_in_df_final_not_in_df2_final = df_final[~df_final['code_geo'].isin(df2_final['code_geo'])] [['nom_geo', 'code_geo']]
print("Municipalités dans df_final mais pas dans df2_final:")
print(code_geo_in_df_final_not_in_df2_final)

# Municipalités dans df2_final mais pas dans df_final
code_geo_in_df2_final_not_in_df_final = df2_final[~df2_final['code_geo'].isin(df_final['code_geo'])] [['nom_geo', 'code_geo']]
print("\nMunicipalités dans df2_final mais pas dans df_final:")
print(code_geo_in_df2_final_not_in_df_final)
```

```
Municipalités dans df_final mais pas dans df2_final:
                     nom_geo  code_geo
476            Daveluyville   2439152
731             L'Épiphanie   2460037
1038  Laverlochère-Angliers   2485052
1271               Ivujivik   2499885

Municipalités dans df2_final mais pas dans df_final:
                    nom_geo  code_geo
476   Sainte-Anne-du-Sault   2439150
477           Daveluyville   2439155
732            L'Épiphanie   2460035
733            L'Épiphanie   2460040
1040          Laverlochère   2485050
1046              Angliers   2485080
1282              Ivujivik   2499895
```

```
# duo nom_géo/code_géo présents dans df_final et pas df2_final et inversement

# Municipalités dans df_final mais pas dans df2_final
nom_geo_in_df_final_not_in_df2_final = df_final[~df_final['nom_geo'].isin(df2_final['nom_geo'])] [['nom_geo', 'code_geo']]
print("Municipalités dans df_final mais pas dans df2_final:")
print(nom_geo_in_df_final_not_in_df2_final)

# Municipalités dans df2_final mais pas dans df_final
nom_geo_in_df2_final_not_in_df_final = df2_final[~df2_final['nom_geo'].isin(df_final['nom_geo'])] [['nom_geo', 'code_geo']]
print("\nMunicipalités dans df2_final mais pas dans df_final:")
print(nom_geo_in_df2_final_not_in_df_final)
```

```
Municipalités dans df_final mais pas dans df2_final:
                                nom_geo  code_geo
129             Saint-Simon-de-Rimouski   2411055
169           Saint-André-de-Kamouraska   2414040
247   Sainte-Famille-de-l'Île-d'Orléans   2420010
484                    Val-des-Sources   2440043
757     Communauté Atikamekw de Manawan   2462802
1038              Laverlochère-Angliers   2485052
1106                          Lac Simon   2489804
1114             Communauté de Wemotaci   2490802
1115                       Obedjiwan 28   2490804
1200                          Uashat 27   2497802
1225                          Romaine 2   2498804
1226                         Nutashkuan   2498806

Municipalités dans df2_final mais pas dans df_final:
                     nom_geo  code_geo
169             Saint-André   2414040
247          Sainte-Famille   2420010
476   Sainte-Anne-du-Sault   2439150
485                Asbestos   2440043
759                 Manawan   2462802
1040          Laverlochère   2485050
1046              Angliers   2485080
1117              Wemotaci   2490802
1118             Obedjiwan   2490804
1203                Uashat   2497802
1228            La Romaine   2498804
```

```
len(merged_df.columns)
```

```
38
```

```
# fréquence du nombre de variables selon le nombre de leurs colonnes avec des valeurs nulles dans merged_df

# Calculer le nombre de valeurs nulles par colonne
null_counts_per_column = merged_df.isnull().sum()

# Créer un DataFrame pour stocker les fréquences
frequency_df = pd.DataFrame(null_counts_per_column.value_counts(), columns=['Frequency'])

# Renommer l'index pour plus de clarté
```

```
frequency_df.index.name = 'Number of Null Values'

# Afficher le DataFrame de fréquences
print(frequency_df)
```

Afficher la sortie masquée

```
merged_df.head(100)
```

Afficher la sortie masquée

```
len(merged_df)
```

```
1300
```

d.csv

```
merged_df.to_csv('/content/gdrive/MyDrive/mémoire/Données/merged_df_menage_imput.csv', encoding = 'utf_8')
```

PARTIE 3 : merged hydro QC et statcan 2016/2021

```
grouped_df = pd.read_csv(r"/content/gdrive/MyDrive/mémoire/Données/grouped_df_imput.csv", encoding='latin-1')
```

```
merged_df = pd.read_csv(r"/content/gdrive/MyDrive/mémoire/Données/merged_df_menage.csv", encoding='utf_8')
```

```
del grouped_df['Unnamed: 0']
grouped_df.head(500)
```

Afficher la sortie masquée

```
del merged_df['Unnamed: 0']
merged_df.head()
```

	code_geo	nom_geo	Population, 2021	Logement occupé selon période de construction - 2016 à 2021 (%)	Population, 2011	Logements privés occupés par des résidents habituels	Nombre moyen de pièces par logement	Population, 2016	Superficie des terres en kilomètres carrés	Taille moyenne des ménages privés	Taux de chômage	Total logem pr
0	2401023	Les Îles-de-la-Madeleine	12190.00	3.58	12291.00	5465.50	6.20	12010.00	163.88	2.15	11.95	631
1	2401042	Grosse-Île	464.00	8.00	490.00	216.50	6.35	465.00	34.94	2.15	14.90	25
2	2402005	Percé	3095.00	2.58	3312.00	1536.50	6.05	3103.00	432.09	2.00	20.40	187
3	2402010	Sainte-Thérèse-de-Gaspé	979.00	2.11	1055.00	468.50	6.05	1015.00	34.43	2.15	9.25	50
4	2402015	Grande-Rivière	3384.00	2.44	3456.00	1626.50	5.75	3408.00	87.89	2.10	12.50	171

```
# inner merged_df avec grouped_df sur le code géo + nom_géo et donne moi deux df : une des nomgéo/codegéo qui n'étaient pas dans merged

merged_df['code_geo'] = merged_df['code_geo'].astype(str)
grouped_df['code_geo'] = grouped_df['code_geo'].astype(str)

inner_merged_df = pd.merge(merged_df, grouped_df, on=['code_geo', 'nom_geo'], how='inner')

not_in_merged = grouped_df[~grouped_df['code_geo'].isin(inner_merged_df['code_geo'])] [['nom_geo', 'code_geo']]

not_in_grouped = merged_df[~merged_df['code_geo'].isin(inner_merged_df['code_geo'])] [['nom_geo', 'code_geo']]

print("Municipalities in grouped_df but not in merged_df:")
print(len(not_in_merged))
```

```
print("\nMunicipalities in merged_df but not in grouped_df:")
print(len(not_in_grouped))


print("\nMunicipalities in merged_df but not in grouped_df:")
print(len(inner_merged_df))
```

```
Municipalities in grouped_df but not in merged_df:
32

Municipalities in merged_df but not in grouped_df:
909

Municipalities in merged_df but not in grouped_df:
389
```

```
# lignes des municipalités dans grouped_df mais pas merged_df

not_in_merged = grouped_df[~grouped_df['code_geo'].isin(inner_merged_df['code_geo'])]
print(not_in_merged)
```

```
                     nom_geo                          region  \
4                  Albertville              Bas-Saint-Laurent
5                        Amqui              Bas-Saint-Laurent
13             Baie-des-Sables              Bas-Saint-Laurent
20                 Betsiamites                     Côte-Nord
39            Carleton-sur-Mer  Gaspésie--Îles-de-la-Madeleine
41                    Chandler  Gaspésie--Îles-de-la-Madeleine
47                  Courcelles                        Estrie
55         Dollard-des-Ormeaux                      Montréal
71                       Gaspé  Gaspésie--Îles-de-la-Madeleine
73                 Gesgapegiag  Gaspésie--Îles-de-la-Madeleine
75                 Grand-Métis              Bas-Saint-Laurent
96    L'Ascension-de-Patapédia  Gaspésie--Îles-de-la-Madeleine
104                  La Martre  Gaspésie--Îles-de-la-Madeleine
105             La Morandière           Abitibi-Témiscamingue
113               Lac-Casault              Bas-Saint-Laurent
136              Les Hauteurs              Bas-Saint-Laurent
137   Les Îles-de-la-Madeleine  Gaspésie--Îles-de-la-Madeleine
139                   Listuguj  Gaspésie--Îles-de-la-Madeleine
155         Mont-Saint-Pierre  Gaspésie--Îles-de-la-Madeleine
183                    Padoue              Bas-Saint-Laurent
185             Petite-Vallée  Gaspésie--Îles-de-la-Madeleine
188              Plessisville               Centre-du-Québec
240     Saint-Charles-Garnier              Bas-Saint-Laurent
275  Saint-Gabriel-de-Rimouski              Bas-Saint-Laurent
305       Saint-Léon-le-Grand              Bas-Saint-Laurent
321     Saint-Octave-de-Métis              Bas-Saint-Laurent
334      Saint-René-de-Matane              Bas-Saint-Laurent
351  Saint-Zénon-du-Lac-Humqui              Bas-Saint-Laurent
363     Sainte-Anne-des-Monts  Gaspésie--Îles-de-la-Madeleine
377            Sainte-Félicité              Bas-Saint-Laurent
383               Sainte-Irène              Bas-Saint-Laurent
418               Val-Brillant              Bas-Saint-Laurent

     median_total_kwh_winter  mean_total_kwh_winter  median_total_kwh_summer  \
4                  215224.00              212729.90                116931.50
5                 5714833.50             5653613.30               3695964.69
13                 617461.50              618386.03                288508.50
20                2219117.00             2185111.98               1198561.00
39                4562149.00             4515891.53               1998456.50
41                8916038.35             8864721.65               4068474.90
47                 780198.00              756078.45                344272.50
55               37032914.50            36182891.27              19442799.27
71               16039334.00            15749893.93               7848432.44
73                 791713.00              768491.93                351400.44
75                 234771.00              238878.33                123868.00
96                 138807.00              136528.11                 76051.50
104                207669.50              205489.81                100844.41
105                206530.36              200473.75                 93846.50
113                 23423.50               23500.17                 10252.00
136                418727.00              422017.78                212294.50
137              11623910.00            11336195.78               5981359.50
139               2204637.50             2204709.70               1361396.50
155                276398.00              275440.43                113837.06
183                183218.00              183535.90                 91327.00
185                213302.50              209897.75                102857.28
188               8592291.00             8489321.93               4104499.00
240                157985.50              162851.65                 94239.50
```

```
# code geo dans merged_df vs grouped_df

# Liste des noms dans not_in_merged
noms_a_trouver = not_in_merged['nom_geo'].tolist()
```

```
# Parcourir les noms dans merged_df et trouver les correspondances approximatives
correspondances = []
for nom_a_trouver in noms_a_trouver:
  for index, row in merged_df.iterrows():
    nom_merged = row['nom_geo']
    code_geo_merged = row['code_geo']
    # Vérifier si le nom_a_trouver est une sous-chaîne de nom_merged (insensible à la casse)
    if nom_a_trouver.lower() in nom_merged.lower():
      # Trouver le code_geo correspondant dans grouped_df
      code_geo_grouped = grouped_df.loc[grouped_df['nom_geo'] == nom_a_trouver, 'code_geo'].values
      if code_geo_grouped.size > 0:
        correspondances.append((nom_a_trouver, code_geo_grouped[0], nom_merged, code_geo_merged))

# Afficher les correspondances trouvées avec les codes geo
print("Correspondances approximatives trouvées avec codes geo:")
for correspondance in correspondances:
  print(f"- {correspondance[0]} (code geo grouped: {correspondance[1]}) pourrait correspondre à {correspondance[2]} (code geo merged: {
```

```
Correspondances approximatives trouvées avec codes geo:
- Albertville (code geo grouped: 247025) pourrait correspondre à Albertville (code geo merged: 2407025)
- Amqui (code geo grouped: 247047) pourrait correspondre à Amqui (code geo merged: 2407047)
- Baie-des-Sables (code geo grouped: 248080) pourrait correspondre à Baie-des-Sables (code geo merged: 2408080)
- Carleton-sur-Mer (code geo grouped: 246013) pourrait correspondre à Carleton-sur-Mer (code geo merged: 2406013)
- Chandler (code geo grouped: 242028) pourrait correspondre à Chandler (code geo merged: 2402028)
- Courcelles (code geo grouped: 2429027) pourrait correspondre à Courcelles (code geo merged: 2430090)
- Gaspé (code geo grouped: 243005) pourrait correspondre à Sainte-Thérèse-de-Gaspé (code geo merged: 2402010)
- Gaspé (code geo grouped: 243005) pourrait correspondre à Gaspé (code geo merged: 2403005)
- Gesgapegiag (code geo grouped: 246802) pourrait correspondre à Gesgapegiag (code geo merged: 2406802)
- Grand-Métis (code geo grouped: 249060) pourrait correspondre à Grand-Métis (code geo merged: 2409060)
- L'Ascension-de-Patapédia (code geo grouped: 246060) pourrait correspondre à L'Ascension-de-Patapédia (code geo merged: 2406060)
- La Martre (code geo grouped: 244030) pourrait correspondre à La Martre (code geo merged: 2404030)
- La Morandière (code geo grouped: 2488012) pourrait correspondre à La Morandière (code geo merged: 2488015)
- Lac-Casault (code geo grouped: 247908) pourrait correspondre à Lac-Casault (code geo merged: 2407908)
- Les Hauteurs (code geo grouped: 249015) pourrait correspondre à Les Hauteurs (code geo merged: 2409015)
- Les Îles-de-la-Madeleine (code geo grouped: 241023) pourrait correspondre à Les Îles-de-la-Madeleine (code geo merged: 2401023)
- Listuguj (code geo grouped: 246804) pourrait correspondre à Listuguj (code geo merged: 2406804)
- Mont-Saint-Pierre (code geo grouped: 244015) pourrait correspondre à Mont-Saint-Pierre (code geo merged: 2404015)
- Padoue (code geo grouped: 249040) pourrait correspondre à Padoue (code geo merged: 2409040)
- Petite-Vallée (code geo grouped: 243015) pourrait correspondre à Petite-Vallée (code geo merged: 2403015)
- Plessisville (code geo grouped: 2432043) pourrait correspondre à Plessisville (code geo merged: 2432040)
- Plessisville (code geo grouped: 2432043) pourrait correspondre à Plessisville (code geo merged: 2432045)
- Saint-Charles-Garnier (code geo grouped: 249010) pourrait correspondre à Saint-Charles-Garnier (code geo merged: 2409010)
- Saint-Gabriel-de-Rimouski (code geo grouped: 249025) pourrait correspondre à Saint-Gabriel-de-Rimouski (code geo merged: 2409025)
- Saint-Léon-le-Grand (code geo grouped: 247030) pourrait correspondre à Saint-Léon-le-Grand (code geo merged: 2407030)
- Saint-Léon-le-Grand (code geo grouped: 247030) pourrait correspondre à Saint-Léon-le-Grand (code geo merged: 2451035)
- Saint-Octave-de-Métis (code geo grouped: 249055) pourrait correspondre à Saint-Octave-de-Métis (code geo merged: 2409055)
- Saint-René-de-Matane (code geo grouped: 248035) pourrait correspondre à Saint-René-de-Matane (code geo merged: 2408035)
- Saint-Zénon-du-Lac-Humqui (code geo grouped: 247035) pourrait correspondre à Saint-Zénon-du-Lac-Humqui (code geo merged: 2407035)
- Sainte-Anne-des-Monts (code geo grouped: 244037) pourrait correspondre à Sainte-Anne-des-Monts (code geo merged: 2404037)
- Sainte-Félicité (code geo grouped: 248023) pourrait correspondre à Sainte-Félicité (code geo merged: 2408023)
- Sainte-Félicité (code geo grouped: 248023) pourrait correspondre à Sainte-Félicité (code geo merged: 2417025)
- Sainte-Irène (code geo grouped: 247040) pourrait correspondre à Sainte-Irène (code geo merged: 2407040)
- Val-Brillant (code geo grouped: 247080) pourrait correspondre à Val-Brillant (code geo merged: 2407080)
```

```
# arranger toutes les municipalités qui correspondent dans merged_df et grouped_df
#(pb de code geo ou villes qui ont changé comme Plessisville ou Courcelles)

# Modifier les codes geo dans grouped_df pour les municipalités correspondantes
for correspondance in correspondances:
  code_geo_grouped = correspondance[1]
  nouveau_code_geo = code_geo_grouped[:2] + '0' + code_geo_grouped[2:]
  grouped_df.loc[grouped_df['code_geo'] == code_geo_grouped, 'code_geo'] = nouveau_code_geo

# moyenne dans merged df des valeurs des deux lignes plessisville puis change le code geo pour 2432045

# Trouver les indices des lignes contenant "Plessisville"
plessisville_indices = merged_df[merged_df['nom_geo'].str.contains("Plessisville")].index

numeric_merged_df = merged_df.loc[plessisville_indices].select_dtypes(include='number')

# Calculer la moyenne des valeurs numériques pour ces lignes
plessisville_mean = numeric_merged_df.mean()

# Créer une nouvelle ligne avec les valeurs moyennes
new_row = plessisville_mean.copy()

# Mettre à jour le code geo et le nom de la nouvelle ligne
new_row['code_geo'] = '24032043'
new_row['nom_geo'] = 'Plessisville'

# Supprimer les lignes originales de Plessisville
merged_df.drop(plessisville_indices, inplace=True)

# Ajouter la nouvelle ligne avec les valeurs moyennes
```

librairies + drive + renommer colonnes

```
#---------------------------------------------------------------------------
# Set up drive
#---------------------------------------------------------------------------
from google.colab import drive
from google.colab import files
drive.mount('/content/gdrive')
```

```
Drive already mounted at /content/gdrive; to attempt to forcibly remount, call drive.mount("/content/gdrive", force_remount=True).
```

```
%load_ext rpy2.ipython
```

```
%%R

if (!requireNamespace("dplyr", quietly = TRUE)) {
  install.packages("dplyr")
}

library(dplyr)
library(tidyverse)

data_sub <- read.csv("/content/gdrive/MyDrive/mémoire/Données/final_merged_df_menage_imput.csv")
names(data_sub)<-str_replace_all(names(data_sub), c(" " = "." , "," = "" ))

colnames(data_sub)
```

Afficher la sortie masquée

```
%%R

data_sub <- data_sub[data_sub$nom_geo != "Montréal", ]
data_sub <- data_sub[data_sub$region != "Nord-du-Québec", ]
```

```
%%R
# Supprimer les lignes avec valeurs manquantes
data_sub <- na.omit(data_sub)
```

```
%%R
# Renommer les colonnes
data_sub <- data_sub %>%
  rename(
    revenu = Revenu.médian.des.ménages.après.impôts,
    consommation_med = median_total_kwh_all,
    consommation_moy = mean_total_kwh_all,
    population = Population..2021,
    reparations_majeures_requises = Logement.occupé.selon.l.état.du.logement...Réparations.majeures.requises....,
    locataires = Ménages.privés.selon.le.mode.d.occupation...Locataire....,
    personnes_agees = Groupes.d.âge.de.la.population...65.ans.et.plus....,
  )

#data_sub <- data_sub[data_sub$revenu < 90000,]

# Ajuster la consommation
data_sub <- data_sub %>%
  mutate(
         logement_construit_avant_1981 = Logement.occupé.selon.période.de.construction...1960.ou.avant....
         + Logement.occupé.selon.période.de.construction...1961.à.1980....,
        id = row_number()
         )
```

```
%%R

head(data_sub)
```

```
     X code_geo                  nom_geo population
1    0  2401023  Les Îles-de-la-Madeleine      12190
3    2  2403005                    Gaspé      15063
7    6  2404037     Sainte-Anne-des-Monts       6121
8    7  2406013          Carleton-sur-Mer       4081
13 12  2407030       Saint-Léon-le-Grand        968
14 13  2407035 Saint-Zénon-du-Lac-Humqui        370
   Logement.occupé.selon.période.de.construction...2016.à.2021....
1                                                         3.584229
3                                                         3.908555
```

```
7                                                    1.980198
8                                                    4.218362
13                                                   6.097561
14                                                   5.263158
   Population..2011 Logements.privés.occupés.par.des.résidents.habituels
1             12291                                                5465.5
3             15163                                                6596.0
7              6933                                                3025.0
8              3991                                                1967.0
13              970                                                 420.0
14              366                                                 181.0
   Nombre.moyen.de.pièces.par.logement Population..2016
1                                 6.20            12010
3                                 6.30            14568
7                                 6.15             6437
8                                 6.25             4073
13                                6.55              953
14                                5.95              359
   Superficie.des.terres.en.kilomètres.carrés Taille.moyenne.des.ménages.privés
1                                     163.885                              2.15
3                                    1120.100                              2.20
7                                     263.800                              2.05
8                                     221.450                              2.05
13                                    128.365                              2.25
14                                    113.255                              2.05
   Taux.de.chômage Total.des.logements.privés
1            11.95                     6318.0
3            10.35                     7213.0
7            16.30                     3330.0
8            12.95                     2179.0
13           14.15                      443.5
14           13.85                      283.5
   Type.de.construction.résidentielle...Maison.individuelle.non.attenante....
1                                                                    84.54165
3                                                                    74.03739
7                                                                    66.61203
8                                                                    74.36091
13                                                                   89.88095
14                                                                   90.27149
   Type.de.construction.résidentielle...Maison.jumelée....
1                                                3.4742605
3                                                2.7704914
7                                                3.4713571
8                                                5.9663203
13                                               0.5952381
14                                               0.0000000
   Type.de.construction.résidentielle...Maison.en.rangée....
```

calcul du score

```
%%R

if (!requireNamespace("neighbr", quietly = TRUE)) {
  install.packages("neighbr")
}
```

```
%%R

# Charger les librairies nécessaires
library(neighbr)

# 1. Sélectionner les variables continues d'intérêt
variable_interet <- data_sub %>%
  select(id, reparations_majeures_requises, personnes_agees, logement_construit_avant_1981, tavg_winter_mean)

# Ajouter une colonne 'id' pour identifier chaque municipalité
variable_interet <- variable_interet %>%
  mutate(id = row_number())  # Assigner un identifiant unique à chaque municipalité

# 2. Définir le nombre de voisins (k)
k <- 10

# Utiliser la fonction knn pour calculer les k-voisins les plus proches
# Dans ce cas, le train_set et test_set seront les mêmes, car on compare les municipalités entre elles
municipalites_knn <- knn(
  train_set = variable_interet,
  test_set = variable_interet  %>% select(-id),  # Même ensemble de test et de train
  k = k,
  comparison_measure = "euclidean",  # Mesure de la distance
  return_ranked_neighbors = k ,  # Retourner les k voisins
  id = "id"
)
```

```
%%R

print(municipalites_knn$test_set_scores)
```

Afficher la sortie masquée

```
%%R

head(clusters, 20)
```

Afficher la sortie masquée

```
%%R

# Compter le nombre d'apparitions de "1" dans le dataframe
count_ones <- sum(municipalites_knn$test_set_scores == 2, na.rm = TRUE)

# Afficher le résultat
print(count_ones)

# Filtrer les lignes où n'importe quelle colonne contient 1
rows_with_ones_any <- municipalites_knn$test_set_scores %>%
  filter(if_any(everything(), ~ . == 1))

# Afficher les résultats
print(rows_with_ones_any)
```

```
[1] 15
   neighbor1 neighbor2 neighbor3 neighbor4 neighbor5 neighbor6 neighbor7
1          1       337       154       168       166       279       313
2        154       337       279       313         1       168       126
3        166       299       222         1       337        36       154
4        168       212       207       154         1        18       189
5        222        36       166       189       299       357       168
6        246       247       301       299       222       300         1
7        247       246       301       299       222       300         1
8        299       166       222         1       301       246       337
9        300       301         1       329       299       168       337
10       301       300       246       247       299         1       168
11       313       337       154        90       305       279       362
12       337       154       313       279         1       126       168
   neighbor8 neighbor9 neighbor10
1        194       212        207
2        305       244        207
3        307       313        357
4         36         4        279
5          4       244          1
6        166       168        189
7        166       285        168
8         36       168        154
9        154       219        246
10       222       337        154
11       126        67          1
12       166       305        362
```

```
%%R

print(neighbor_data)
```

```
WARNING:rpy2.rinterface_lib.callbacks:R[write to console]: Error in (function (expr, envir = parent.frame(), enclos = if (is.list(er
  object 'neighbor_data' not found


Error in (function (expr, envir = parent.frame(), enclos = if (is.list(envir) ||  :
  object 'neighbor_data' not found
```

```
%%R

# Charger les librairies nécessaires
library(ggplot2)
library(tidyr)

# 1. Préparer les données des voisins avec leurs variables d'intérêt
# Transformer le tableau de scores en format long
neighbor_ids <- municipalites_knn$test_set_scores %>%
  pivot_longer(cols = starts_with("neighbor"), names_to = "neighbor_num", values_to = "id")

# Joindre avec les données des municipalités pour obtenir les variables d'intérêt
neighbor_data <- neighbor_ids %>%
  left_join(data_sub, by = c("neighbor_id" = "id")) %>%
```

```
  select(neighbor_id,
         neighbor_num,
         reparations_majeures_requises,
         locataires,
         personnes_agees,
         logement_construit_avant_1981,
         tavg_winter_mean)

# 2. Obtenir les données de la municipalité d'origine
municipality_data <- municipalites_knn$test_set_scores %>%
  select(id, neighbor1, neighbor2, neighbor3, neighbor4, neighbor5, neighbor6, neighbor7, neighbor8, neighbor9, neighbor10) %>%
  left_join(data_sub, by = c("id" = "id"))

# 3. Fusionner les données de la municipalité et des voisins
full_data <- municipality_data %>%
  select(id, nom_geo, reparations_majeures_requises, locataires, personnes_agees, logement_construit_avant_1981, tavg_winter_mean) %>%
  bind_rows(neighbor_data)

# 4. Créer des graphiques pour chaque variable d'intérêt
variables_interet <- c("reparations_majeures_requises", "locataires", "personnes_agees", "logement_construit_avant_1981", "tavg_winter_r

for (variable in variables_interet) {
  ggplot(full_data, aes_string(x = "nom_geo", y = variable, color = "as.factor(neighbor_num)")) +
    geom_boxplot() +
    labs(title = paste("Distribution de", variable, "par municipalité et voisins"),
         x = "Municipalités",
         y = variable,
         color = "Voisin") +
    theme(axis.text.x = element_text(angle = 90, hjust = 1)) +
    theme_minimal() +
    ggsave(paste0("graph_", variable, ".png"), width = 10, height = 6)  # Enregistrer le graphique
}
```

```
Error in `left_join()`:
! Join columns in `x` must be present in the data.
✖ Problem with `neighbor_id`.
Run `rlang::last_trace()` to see where the error occurred.

Error in left_join(., data_sub, by = c(neighbor_id = "id")) :
✖ Problem with `neighbor_id`.
```

```
%%R

write.csv(municipalites_knn$test_set_scores, "/content/gdrive/MyDrive/mémoire/Données/voisins_10_imput.csv", row.names = FALSE, fileEnc
```

```
%%R

write.csv(data_sub, "/content/gdrive/MyDrive/mémoire/Données/data_voisins_imput.csv", row.names = FALSE, fileEncoding = "UTF-8")
```

Double-cliquez (ou appuyez sur Entrée) pour modifier

```
#-------------------------------------------------------------------------------
# Set up drive
#-------------------------------------------------------------------------------
from google.colab import drive
from google.colab import files
drive.mount('/content/gdrive')
```

```
Mounted at /content/gdrive
```

```
#-------------------------------------------------------------------------------
# Import libaries
#-------------------------------------------------------------------------------
import pandas as pd
import numpy as np

pd.set_option('display.max_columns', None)
pd.set_option('display.max_rows', None)


pd.options.display.float_format = '{:.2f}'.format


df = pd.read_csv(r"/content/gdrive/MyDrive/mémoire/Données/data_voisins.csv", encoding='utf_8')


df_imput = pd.read_csv(r"/content/gdrive/MyDrive/mémoire/Données/data_voisins_imput.csv", encoding='utf_8')


df.head()
```

	X	Unnamed..0	code_geo	nom_geo	population	Logement.occupé.selon.période.de.construction...2016.à.2021....	Population..2011
0	0	0	2401023	Les Îles-de-la-Madeleine	12190.00	3.58	12291
1	3	3	2406013	Carleton-sur-Mer	4081.00	4.22	3991
2	4	4	2407030	Saint-Léon-le-Grand	968.00	6.10	970
3	5	5	2407035	Saint-Zénon-du-Lac-Humqui	370.00	5.26	366
4	6	6	2407040	Sainte-Irène	369.00	0.00	341

```
df.columns
```

```
Index(['X', 'Unnamed..0', 'code_geo', 'nom_geo', 'population',
       'Logement.occupé.selon.période.de.construction...2016.à.2021....',
       'Population..2011',
       'Logements.privés.occupés.par.des.résidents.habituels',
       'Nombre.moyen.de.pièces.par.logement', 'Population..2016',
       'Superficie.des.terres.en.kilomètres.carrés',
       'Taille.moyenne.des.ménages.privés', 'Taux.de.chômage',
       'Total.des.logements.privés',
       'Type.de.construction.résidentielle...Maison.individuelle.non.attenante....',
       'Type.de.construction.résidentielle...Maison.jumelée....',
       'Type.de.construction.résidentielle...Maison.en.rangée....',
       'Type.de.construction.résidentielle...Appartement.ou.plain.pied.dans.un.duplex....',
       'Type.de.construction.résidentielle...Appartement.dans.un.immeuble.de.moins.de.cinq.étages....',
       'Type.de.construction.résidentielle...Appartement.dans.un.immeuble.de.cinq.étages.ou.plus....',
       'Logement.occupé.selon.période.de.construction...1960.ou.avant....',
       'Logement.occupé.selon.période.de.construction...1961.à.1980....',
       'Logement.occupé.selon.période.de.construction...1981.à.1990....',
       'Logement.occupé.selon.période.de.construction...1991.à.2000....',
       'Logement.occupé.selon.période.de.construction...2001.à.2005....',
       'Logement.occupé.selon.période.de.construction...2006.à.2010....',
       'Logement.occupé.selon.période.de.construction...2011.à.2015....',
       'reparations_majeures_requises',
       'Ménages.privés.selon.le.mode.d.occupation...Propriétaire....',
       'locataires',
       'Ménages.privés.selon.le.mode.d.occupation...Logement.fourni.par.le.gouvernement.local....',
       'Plus.haut.diplôme...Aucun.certificat..diplôme.ou.grade....',
       'Plus.haut.diplôme...Diplôme.d.études.secondaires.ou.attestation.d.équivalence....',
```

```
       'Plus.haut.diplôme...Certificat..diplôme.ou.grade.d.études.postsecondaires....',
       'personnes_agees',
       'Ménages.dont.30...ou.plus.du.revenu.est.consacré.aux.frais.de.logement.seulement....',
       'Revenu.médian.moyen.après.impôts', 'Revenu.médian.moyen.avant.impôts',
       'Revenu.médian.des.ménages.avant.impôts', 'revenu', 'region',
       'median_total_kwh_winter', 'mean_total_kwh_winter',
       'median_total_kwh_summer', 'mean_total_kwh_summer', 'consommation_med',
       'consommation_moy', 'code_postal', 'location', 'tavg_winter_mean',
       'tmin_winter_min', 'tmax_winter_max', 'tavg_other_mean',
       'tmin_other_min', 'tmax_other_max', 'logement_construit_avant_1981',
       'id', 'conso_moy_mensuelle', 'cout_electricite_menage_an',
       'pourcentage_revenus_electricite', 'imputé..'],
      dtype='object')
```

```
df_imput.head()
```

	X	code_geo	nom_geo	population	Logement.occupé.selon.période.de.construction...2016.à.2021....	Population..2011	Logements
0	0	2401023	Les Îles-de-la-Madeleine	12190.00	3.58	12291	
1	2	2403005	Gaspé	15063.00	3.91	15163	
2	6	2404037	Sainte-Anne-des-Monts	6121.00	1.98	6933	
3	7	2406013	Carleton-sur-Mer	4081.00	4.22	3991	
4	12	2407030	Saint-Léon-le-Grand	968.00	6.10	970	

```
del df['Unnamed..0']
del df_imput['X']
del df['X']
```

```
df.columns
```

```
Index(['code_geo', 'nom_geo', 'population',
       'Logement.occupé.selon.période.de.construction...2016.à.2021....',
       'Population..2011',
       'Logements.privés.occupés.par.des.résidents.habituels',
       'Nombre.moyen.de.pièces.par.logement', 'Population..2016',
       'Superficie.des.terres.en.kilomètres.carrés',
       'Taille.moyenne.des.ménages.privés', 'Taux.de.chômage',
       'Total.des.logements.privés',
       'Type.de.construction.résidentielle...Maison.individuelle.non.attenante....',
       'Type.de.construction.résidentielle...Maison.jumelée....',
       'Type.de.construction.résidentielle...Maison.en.rangée....',
       'Type.de.construction.résidentielle...Appartement.ou.plain.pied.dans.un.duplex....',
       'Type.de.construction.résidentielle...Appartement.dans.un.immeuble.de.moins.de.cinq.étages....',
       'Type.de.construction.résidentielle...Appartement.dans.un.immeuble.de.cinq.étages.ou.plus....',
       'Logement.occupé.selon.période.de.construction...1960.ou.avant....',
       'Logement.occupé.selon.période.de.construction...1961.à.1980....',
       'Logement.occupé.selon.période.de.construction...1981.à.1990....',
       'Logement.occupé.selon.période.de.construction...1991.à.2000....',
       'Logement.occupé.selon.période.de.construction...2001.à.2005....',
       'Logement.occupé.selon.période.de.construction...2006.à.2010....',
       'Logement.occupé.selon.période.de.construction...2011.à.2015....',
       'reparations_majeures_requises',
       'Ménages.privés.selon.le.mode.d.occupation...Propriétaire....',
       'locataires',
       'Ménages.privés.selon.le.mode.d.occupation...Logement.fourni.par.le.gouvernement.local....',
       'Plus.haut.diplôme...Aucun.certificat..diplôme.ou.grade....',
       'Plus.haut.diplôme...Diplôme.d.études.secondaires.ou.attestation.d.équivalence....',
       'Plus.haut.diplôme...Certificat..diplôme.ou.grade.d.études.postsecondaires....',
       'personnes_agees',
       'Ménages.dont.30...ou.plus.du.revenu.est.consacré.aux.frais.de.logement.seulement....',
       'Revenu.médian.moyen.après.impôts', 'Revenu.médian.moyen.avant.impôts',
       'Revenu.médian.des.ménages.avant.impôts', 'revenu', 'region',
       'median_total_kwh_winter', 'mean_total_kwh_winter',
       'median_total_kwh_summer', 'mean_total_kwh_summer', 'consommation_med',
       'consommation_moy', 'code_postal', 'location', 'tavg_winter_mean',
       'tmin_winter_min', 'tmax_winter_max', 'tavg_other_mean',
       'tmin_other_min', 'tmax_other_max', 'logement_construit_avant_1981',
       'id', 'conso_moy_mensuelle', 'cout_electricite_menage_an',
```

```
       'pourcentage_revenus_electricite', 'imputé..'],
      dtype='object')
```

```
df_imput.columns
```

```
Index(['code_geo', 'nom_geo', 'population',
       'Logement.occupé.selon.période.de.construction...2016.à.2021....',
       'Population..2011',
       'Logements.privés.occupés.par.des.résidents.habituels',
       'Nombre.moyen.de.pièces.par.logement', 'Population..2016',
       'Superficie.des.terres.en.kilomètres.carrés',
       'Taille.moyenne.des.ménages.privés', 'Taux.de.chômage',
       'Total.des.logements.privés',
       'Type.de.construction.résidentielle...Maison.individuelle.non.attenante....',
       'Type.de.construction.résidentielle...Maison.jumelée....',
       'Type.de.construction.résidentielle...Maison.en.rangée....',
       'Type.de.construction.résidentielle...Appartement.ou.plain.pied.dans.un.duplex....',
       'Type.de.construction.résidentielle...Appartement.dans.un.immeuble.de.moins.de.cinq.étages....',
       'Type.de.construction.résidentielle...Appartement.dans.un.immeuble.de.cinq.étages.ou.plus....',
       'Logement.occupé.selon.période.de.construction...1960.ou.avant....',
       'Logement.occupé.selon.période.de.construction...1961.à.1980....',
       'Logement.occupé.selon.période.de.construction...1981.à.1990....',
       'Logement.occupé.selon.période.de.construction...1991.à.2000....',
       'Logement.occupé.selon.période.de.construction...2001.à.2005....',
       'Logement.occupé.selon.période.de.construction...2006.à.2010....',
       'Logement.occupé.selon.période.de.construction...2011.à.2015....',
       'reparations_majeures_requises',
       'Ménages.privés.selon.le.mode.d.occupation...Propriétaire....',
       'locataires',
       'Ménages.privés.selon.le.mode.d.occupation...Logement.fourni.par.le.gouvernement.local....',
       'Plus.haut.diplôme...Aucun.certificat..diplôme.ou.grade....',
       'Plus.haut.diplôme...Diplôme.d.études.secondaires.ou.attestation.d.équivalence....',
       'Plus.haut.diplôme...Certificat..diplôme.ou.grade.d.études.postsecondaires....',
       'personnes_agees',
       'Ménages.dont.30...ou.plus.du.revenu.est.consacré.aux.frais.de.logement.seulement....',
       'Revenu.médian.moyen.après.impôts', 'Revenu.médian.moyen.avant.impôts',
       'Revenu.médian.des.ménages.avant.impôts', 'revenu', 'region',
       'median_total_kwh_winter', 'mean_total_kwh_winter',
       'median_total_kwh_summer', 'mean_total_kwh_summer', 'consommation_med',
       'consommation_moy', 'code_postal', 'location', 'tavg_winter_mean',
       'tmin_winter_min', 'tmax_winter_max', 'tavg_other_mean',
       'tmin_other_min', 'tmax_other_max', 'logement_construit_avant_1981',
       'id'],
      dtype='object')
```

```
df_imput.head()
```

Afficher la sortie masquée

```
# lignes de villes qu ont le même nom_geo mais pas le même code_geo

duplicates = df[df.duplicated(subset=['nom_geo'], keep=False)]
different_code_geo = duplicates[duplicates.groupby('nom_geo')['code_geo'].transform('nunique') > 1]
print(different_code_geo[['nom_geo', 'code_geo']])



# villes et leur région de df_imput qui ont le même nom de ville et une région différente

duplicates = df_imput[df_imput.duplicated(subset=['nom_geo'], keep=False)]
different_region = duplicates[duplicates.groupby('nom_geo')['region'].transform('nunique') > 1]
print(different_region[['nom_geo', 'region']])


# nouvelle colonne "imputé ?" dans df_imput ou c'est "TRUE" si le nom_geo n'est pas dans df et FALSE s'il l'est

df_imput['imputé ?'] = ~df_imput['nom_geo'].isin(df['nom_geo'])


df_imput.head()


df[df['nom_geo'] == "Saint-Isidore"]['region']


df_imput[df_imput['nom_geo'] == "Saint-Isidore"][['region', "imputé ?"]]


# changer colonne "imputé ?"en true pour le row où nom geo == Saint-Léon-le-Grand et region == Mauricie

df_imput.loc[(df_imput['nom_geo'] == 'Saint-Léon-le-Grand') & (df_imput['region'] == 'Mauricie'), 'imputé ?'] = True


#df_imput.to_csv('/content/gdrive/MyDrive/mémoire/Données/data_imput_both.csv', encoding = 'utf_8')
```

```
# combien de municipalités avec imputé "true" et combien "imputé" false

imputed_count = df_imput['imputé ?'].sum()
not_imputed_count = len(df_imput) - imputed_count

print(f"Nombre de municipalités avec imputé True: {imputed_count}")
print(f"Nombre de municipalités avec imputé False: {not_imputed_count}")


len(df_imput)


len(df)
```

librairies + drive + renommer colonnes

1 cellule masquée

créer data frame pas imputée

3 cellules masquées

R CODE

```
df = pd.read_csv(r'/content/gdrive/MyDrive/mémoire/Données/data_imput_not.csv', encoding= "utf-8")
```

```
%load_ext rpy2.ipython
```

```
The rpy2.ipython extension is already loaded. To reload it, use:
  %reload_ext rpy2.ipython
```

```
%%R

if (!requireNamespace("dplyr", quietly = TRUE)) {
  install.packages("dplyr")
}

library(dplyr)
library(tidyverse)

data_sub <- read.csv("/content/gdrive/MyDrive/mémoire/Données/data_imput_not.csv")
names(data_sub)<-str_replace_all(names(data_sub), c(" " = "." , "," = "" ))

colnames(data_sub)
```

Afficher la sortie masquée

```
%%R
# Supprimer les lignes avec valeurs manquantes
data_sub <- na.omit(data_sub)

# Ajuster la consommation
data_sub <- data_sub %>%
  mutate(
        id = row_number()
         )
```

```
%%R

head(data_sub)
```

```
  X Unnamed..0 code_geo                  nom_geo population
1 0          0  2401023  Les Îles-de-la-Madeleine      12190
2 3          3  2406013          Carleton-sur-Mer       4081
3 4          4  2407030       Saint-Léon-le-Grand        968
4 5          5  2407035 Saint-Zénon-du-Lac-Humqui        370
5 6          6  2407040              Sainte-Irène        369
6 8          8  2407080              Val-Brillant        899
  Logement.occupé.selon.période.de.construction...2016.à.2021....
1                                                       3.584229
2                                                       4.218362
3                                                       6.097561
4                                                       5.263158
5                                                       0.000000
6                                                       0.000000
  Population..2011 Logements.privés.occupés.par.des.résidents.habituels
1            12291                                                5465.5
2             3991                                                1967.0
3              970                                                 420.0
4              366                                                 181.0
5              341                                                 167.0
6              955                                                 417.5
  Nombre.moyen.de.pièces.par.logement Population..2016
1                                6.20            12010
2                                6.25             4073
3                                6.55              953
4                                5.95              359
5                                6.65              327
```

```
6                              6.40             927
  Superficie.des.terres.en.kilomètres.carrés Taille.moyenne.des.ménages.privés
1                                    163.885                                2.15
2                                    221.450                                2.05
3                                    128.365                                2.25
4                                    113.255                                2.05
5                                    135.170                                2.10
6                                     77.970                                2.20
  Taux.de.chômage Total.des.logements.privés
1           11.95                     6318.0
2           12.95                     2179.0
3           14.15                      443.5
4           13.85                      283.5
5           12.65                      252.5
6           13.85                      500.5
  Type.de.construction.résidentielle...Maison.individuelle.non.attenante....
1                                                                   84.54165
2                                                                   74.36091
3                                                                   89.88095
4                                                                   90.27149
5                                                                   97.00893
6                                                                   84.93976
  Type.de.construction.résidentielle...Maison.jumelée....
1                                               3.4742605
2                                               5.9663203
3                                               0.5952381
4                                               0.0000000
5                                               0.0000000
6                                               2.4096386
  Type.de.construction.résidentielle...Maison.en.rangée....
1                                                 2.014227
```

calcul du score

```
%%R

if (!requireNamespace("neighbr", quietly = TRUE)) {
  install.packages("neighbr")
}
```

```
%%R

# Charger les librairies nécessaires
library(neighbr)

# 1. Sélectionner les variables continues d'intérêt
variable_interet <- data_sub %>%
  select(id, reparations_majeures_requises, personnes_agees, logement_construit_avant_1981, tavg_winter_mean)

# Ajouter une colonne 'id' pour identifier chaque municipalité
variable_interet <- variable_interet %>%
  mutate(id = row_number())  # Assigner un identifiant unique à chaque municipalité

# 2. Définir le nombre de voisins (k)
k <- 10

# Utiliser la fonction knn pour calculer les k-voisins les plus proches
# Dans ce cas, le train_set et test_set seront les mêmes, car on compare les municipalités entre elles
municipalites_knn <- knn(
  train_set = variable_interet,
  test_set = variable_interet  %>% select(-id),  # Même ensemble de test et de train
  k = k,
  comparison_measure = "euclidean",  # Mesure de la distance
  return_ranked_neighbors = k ,  # Retourner les k voisins
  id = "id"
)
```

```
%%R

print(municipalites_knn$test_set_scores)
```

Afficher la sortie masquée

```
%%R

head(clusters, 20)
```

Afficher la sortie masquée

```
%%R
# Compter le nombre d'apparitions de "1" dans le dataframe
count_ones <- sum(municipalites_knn$test_set_scores == 2, na.rm = TRUE)

# Afficher le résultat
print(count_ones)

# Filtrer les lignes où n'importe quelle colonne contient 1
rows_with_ones_any <- municipalites_knn$test_set_scores %>%
  filter(if_any(everything(), ~ . == 1))

# Afficher les résultats
print(rows_with_ones_any)
```

```
[1] 11
  neighbor1 neighbor2 neighbor3 neighbor4 neighbor5 neighbor6 neighbor7
1         1        83       173       107       120       116       180
2        83       173         1        66       116        13       103
3       165         1       180        83       123       120       116
4       173        83        50       197        66         1       107
5       180       131        91       123       197       121       146
  neighbor8 neighbor9 neighbor10
1        66        24         91
2        24       197          2
3       146       131        128
4       116        14        180
5        53       107          1
```

```
%%R

print(neighbor_data)
```

```
WARNING:rpy2.rinterface_lib.callbacks:R[write to console]: Error in (function (expr, envir = parent.frame(), enclos = if (is.list(er
  object 'neighbor_data' not found


Error in (function (expr, envir = parent.frame(), enclos = if (is.list(envir) ||  :
  object 'neighbor_data' not found
```

```
%%R

# Charger les librairies nécessaires
library(ggplot2)
library(tidyr)

# 1. Préparer les données des voisins avec leurs variables d'intérêt
# Transformer le tableau de scores en format long
neighbor_ids <- municipalites_knn$test_set_scores %>%
  pivot_longer(cols = starts_with("neighbor"), names_to = "neighbor_num", values_to = "id")

# Joindre avec les données des municipalités pour obtenir les variables d'intérêt
neighbor_data <- neighbor_ids %>%
  left_join(data_sub, by = c("neighbor_id" = "id")) %>%
  select(neighbor_id,
         neighbor_num,
         reparations_majeures_requises,
         locataires,
         personnes_agees,
         logement_construit_avant_1981,
         tavg_winter_mean)

# 2. Obtenir les données de la municipalité d'origine
municipality_data <- municipalites_knn$test_set_scores %>%
  select(id, neighbor1, neighbor2, neighbor3, neighbor4, neighbor5, neighbor6, neighbor7, neighbor8, neighbor9, neighbor10) %>%
  left_join(data_sub, by = c("id" = "id"))

# 3. Fusionner les données de la municipalité et des voisins
full_data <- municipality_data %>%
  select(id, nom_geo, reparations_majeures_requises, locataires, personnes_agees, logement_construit_avant_1981, tavg_winter_mean) %>%
  bind_rows(neighbor_data)

# 4. Créer des graphiques pour chaque variable d'intérêt
variables_interet <- c("reparations_majeures_requises", "locataires", "personnes_agees", "logement_construit_avant_1981", "tavg_winter_r

for (variable in variables_interet) {
  ggplot(full_data, aes_string(x = "nom_geo", y = variable, color = "as.factor(neighbor_num)")) +
    geom_boxplot() +
    labs(title = paste("Distribution de", variable, "par municipalité et voisins"),
         x = "Municipalités",
         y = variable,
```

```
          color = "Voisin") +
    theme(axis.text.x = element_text(angle = 90, hjust = 1)) +
    theme_minimal() +
    ggsave(paste0("graph_", variable, ".png"), width = 10, height = 6)  # Enregistrer le graphique
}
```

```
Error in `left_join()`:
! Join columns in `x` must be present in the data.
✖ Problem with `neighbor_id`.
Run `rlang::last_trace()` to see where the error occurred.

Error in left_join(., data_sub, by = c(neighbor_id = "id")) :
✖ Problem with `neighbor_id`.
```

```
%%R

write.csv(municipalites_knn$test_set_scores, "/content/gdrive/MyDrive/mémoire/Données/voisins_10.csv", row.names = FALSE, fileEncoding =
```

```
%%R

write.csv(data_sub, "/content/gdrive/MyDrive/mémoire/Données/data_voisins.csv", row.names = FALSE, fileEncoding = "UTF-8")
```

Double-cliquez (ou appuyez sur Entrée) pour modifier

drive + librairies

```
#-------------------------------------------------------------------------------
# Set up drive
#-------------------------------------------------------------------------------
from google.colab import drive
from google.colab import files
drive.mount('/content/gdrive')
```

```
Drive already mounted at /content/gdrive; to attempt to forcibly remount, call drive.mount("/content/gdrive", force_remount=True).
```

```
#-------------------------------------------------------------------------------
# Import libaries
#-------------------------------------------------------------------------------
import pandas as pd
import numpy as np

pd.set_option('display.max_columns', None)
pd.set_option('display.max_rows', None)


pd.options.display.float_format = '{:.2f}'.format


voisins = pd.read_csv(r'/content/gdrive/MyDrive/mémoire/Données/voisins_10.csv', encoding= "utf-8")
df = pd.read_csv(r'/content/gdrive/MyDrive/mémoire/Données/data_voisins.csv', encoding= "utf-8")


df.head()
```

	X	Unnamed..0	code_geo	nom_geo	population	Logement.occupé.selon.période.de.construction...2016.à.2021....	Population..2011
0	0	0	2401023	Les Îles-de-la-Madeleine	12190.00	3.58	12291
1	3	3	2406013	Carleton-sur-Mer	4081.00	4.22	3991
2	4	4	2407030	Saint-Léon-le-Grand	968.00	6.10	970
3	5	5	2407035	Saint-Zénon-du-Lac-Humqui	370.00	5.26	366
4	6	6	2407040	Sainte-Irène	369.00	0.00	341

```
del df['X']


# ajouter une colonne "conso_moy_menage"

df['conso_moy_menage'] = (df['consommation_med'] / (df['population'] + df['Population..2016']) *2) * df['Taille.moyenne.des.ménages.pri

df.head()
```

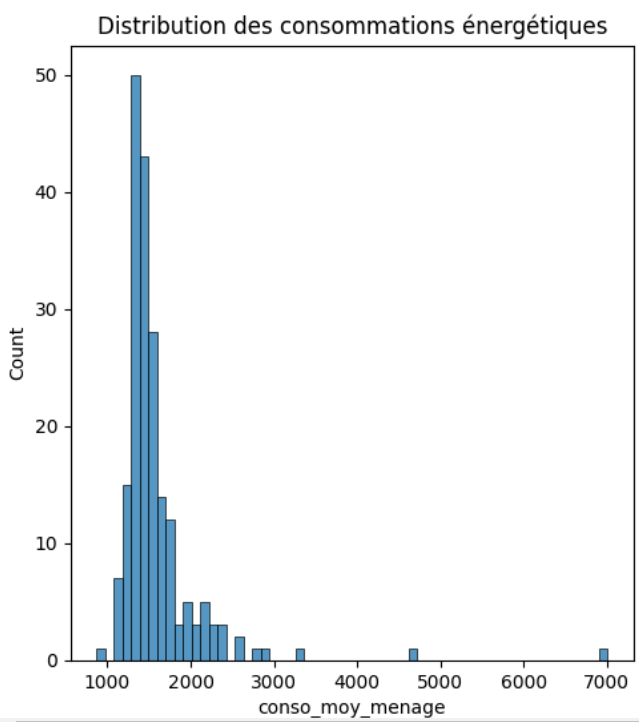
	Unnamed..0	code_geo	nom_geo	population	Logement.occupé.selon.période.de.construction...2016.à.2021....	Population..2011	L
0	0	2401023	Les Îles-de-la-Madeleine	12190.00	3.58	12291	
1	3	2406013	Carleton-sur-Mer	4081.00	4.22	3991	
2	4	2407030	Saint-Léon-le-Grand	968.00	6.10	970	
3	5	2407035	Saint-Zénon-du-Lac-Humqui	370.00	5.26	366	
4	6	2407040	Sainte-Irène	369.00	0.00	341	

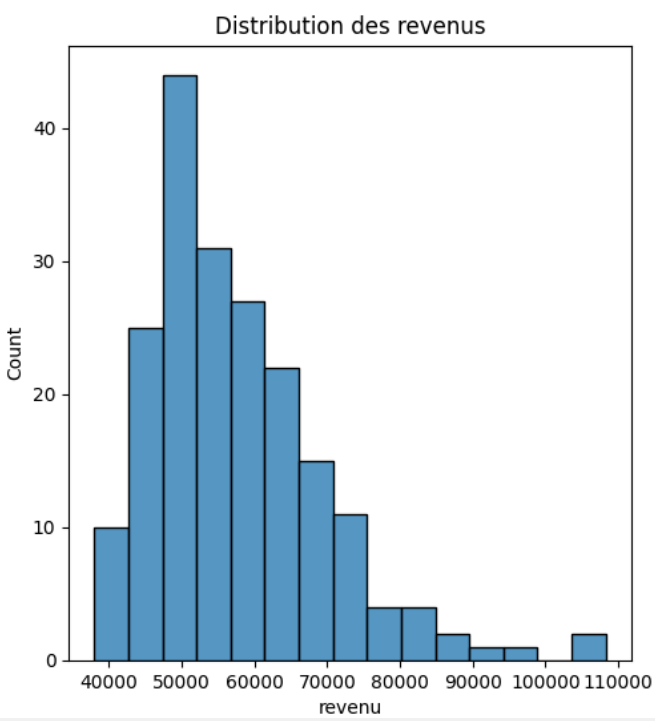
```
# lignes des villes de la région Nord-du-Québec

nord_du_quebec_rows = df[df['region'] == 'Nord-du-Québec']
nord_du_quebec_rows
```

	Unnamed..0	code_geo	nom_geo	population	Logement.occupé.selon.période.de.construction...2016.à.2021....	Population..2011	Loge

exploration

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns

# Visualiser la distribution des consommations énergétiques et des revenus
plt.figure(figsize=(12, 6))
plt.subplot(1, 2, 1)
sns.histplot(df['conso_moy_menage'], kde=False)
plt.title('Distribution des consommations énergétiques')

plt.subplot(1, 2, 2)
sns.histplot(df['revenu'], kde=False)
plt.title('Distribution des revenus')
plt.show()
```





n voisins

```
voisins.head()
```

	neighbor1	neighbor2	neighbor3	neighbor4	neighbor5	neighbor6	neighbor7	neighbor8	neighbor9	neighbor10
0	1	83	173	107	120	116	180	66	24	91
1	2	24	103	13	120	114	189	116	107	48
2	3	29	106	31	143	16	36	10	74	58
3	4	37	177	57	7	58	27	192	21	15
4	5	55	172	193	75	170	72	65	95	70

```
df['nom_geo'].nunique()
```

```
199
```

```
df.columns
```

```
Index(['Unnamed..0', 'code_geo', 'nom_geo', 'population',
       'Logement.occupé.selon.période.de.construction...2016.à.2021....',
       'Population..2011',
       'Logements.privés.occupés.par.des.résidents.habituels',
       'Nombre.moyen.de.pièces.par.logement', 'Population..2016',
       'Superficie.des.terres.en.kilomètres.carrés',
       'Taille.moyenne.des.ménages.privés', 'Taux.de.chômage',
       'Total.des.logements.privés',
       'Type.de.construction.résidentielle...Maison.individuelle.non.attenante....',
       'Type.de.construction.résidentielle...Maison.jumelée....',
       'Type.de.construction.résidentielle...Maison.en.rangée....',
       'Type.de.construction.résidentielle...Appartement.ou.plain.pied.dans.un.duplex....',
       'Type.de.construction.résidentielle...Appartement.dans.un.immeuble.de.moins.de.cinq.étages....',
       'Type.de.construction.résidentielle...Appartement.dans.un.immeuble.de.cinq.étages.ou.plus....',
       'Logement.occupé.selon.période.de.construction...1960.ou.avant....',
       'Logement.occupé.selon.période.de.construction...1961.à.1980....',
       'Logement.occupé.selon.période.de.construction...1981.à.1990....',
       'Logement.occupé.selon.période.de.construction...1991.à.2000....',
       'Logement.occupé.selon.période.de.construction...2001.à.2005....',
       'Logement.occupé.selon.période.de.construction...2006.à.2010....',
       'Logement.occupé.selon.période.de.construction...2011.à.2015....',
       'reparations_majeures_requises',
       'Ménages.privés.selon.le.mode.d.occupation...Propriétaire....',
       'locataires',
       'Ménages.privés.selon.le.mode.d.occupation...Logement.fourni.par.le.gouvernement.local....',
       'Plus.haut.diplôme...Aucun.certificat..diplôme.ou.grade....',
       'Plus.haut.diplôme...Diplôme.d.études.secondaires.ou.attestation.d.équivalence....',
       'Plus.haut.diplôme...Certificat..diplôme.ou.grade.d.études.postsecondaires....',
       'personnes_agees',
```

```
       'Ménages.dont.30...ou.plus.du.revenu.est.consacré.aux.frais.de.logement.seulement....',
       'Revenu.médian.moyen.après.impôts', 'Revenu.médian.moyen.avant.impôts',
       'Revenu.médian.des.ménages.avant.impôts', 'revenu', 'region',
       'median_total_kwh_winter', 'mean_total_kwh_winter',
       'median_total_kwh_summer', 'mean_total_kwh_summer', 'consommation_med',
       'consommation_moy', 'code_postal', 'location', 'tavg_winter_mean',
       'tmin_winter_min', 'tmax_winter_max', 'tavg_other_mean',
       'tmin_other_min', 'tmax_other_max', 'logement_construit_avant_1981',
       'id', 'conso_moy_mensuelle', 'cout_electricite_menage_an',
       'pourcentage_revenus_electricite', 'imputé..', 'conso_moy_menage'],
      dtype='object')
```

```
df.head()
```

	Unnamed..0	code_geo	nom_geo	population	Logement.occupé.selon.période.de.construction...2016.à.2021....	Population..2011	L
0	0	2401023	Les Îles-de-la-Madeleine	12190.00	3.58	12291	
1	3	2406013	Carleton-sur-Mer	4081.00	4.22	3991	
2	4	2407030	Saint-Léon-le-Grand	968.00	6.10	970	
3	5	2407035	Saint-Zénon-du-Lac-Humqui	370.00	5.26	366	
4	6	2407040	Sainte-Irène	369.00	0.00	341	

```
# pour chaque municipalité (neighbor1 dans la df "voisins"), calculer le revenu médian et moyen de son groupe (neighbor2, 3 etc. jusqu'à
#ainsi que sa consommation d'électricité médiane et moyenne en utilisant "revenu" de df ainsi que  "conso_moy_menage"

import pandas as pd
import numpy as np
def calculate_group_stats(row):
  """Calculates median and mean income and electricity consumption for a group of municipalities."""
  neighbor_list = [row[f'neighbor{i}'] for i in range(1, 11) if not pd.isnull(row[f'neighbor{i}'])]
  if not neighbor_list:
    return pd.Series({'median_income': np.nan, 'mean_income': np.nan, 'median_consumption': np.nan, 'mean_consumption': np.nan})

  group_data = df[df['id'].isin(neighbor_list)]

  median_income = group_data['revenu'].median()
  mean_income = group_data['revenu'].mean()
  median_consumption = group_data['conso_moy_menage'].median()
  mean_consumption = group_data['conso_moy_menage'].mean()

  return pd.Series({'median_income': median_income, 'mean_income': mean_income, 'median_consumption': median_consumption, 'mean_consumpt

group_stats = voisins.apply(calculate_group_stats, axis=1)

voisins = pd.concat([voisins, group_stats], axis=1)

voisins.head(50)
```

	neighbor1	neighbor2	neighbor3	neighbor4	neighbor5	neighbor6	neighbor7	neighbor8	neighbor9	neighbor10	median_income
0	1	83	173	107	120	116	180	66	24	91	56548.00
1	2	24	103	13	120	114	189	116	107	48	52332.00
2	3	29	106	31	143	16	36	10	74	58	49108.00
3	4	37	177	57	7	58	27	192	21	15	49268.00
4	5	55	172	193	75	170	72	65	95	70	58454.50
5	6	21	76	12	199	63	110	22	30	97	50219.25
6	7	25	4	178	37	151	177	34	32	192	46652.00
7	8	111	26	32	196	33	28	178	90	18	48978.75
8	9	11	86	111	19	33	8	196	18	69	49154.75
9	10	97	63	104	16	3	29	36	150	153	50039.25
10	11	9	86	111	33	19	18	8	196	178	47730.75
11	12	21	63	199	6	97	150	76	186	110	50219.25
12	13	116	66	103	2	38	120	54	128	83	55496.25
13	14	197	180	173	50	91	131	123	53	121	57978.00
14	15	16	150	186	3	31	106	58	97	21	49566.50
15	16	150	15	97	10	3	63	31	186	29	49712.00
16	17	64	38	119	124	54	128	166	156	140	62233.00
17	18	115	184	36	153	104	86	8	111	90	50576.00
18	19	86	69	142	111	18	141	8	9	60	49154.75
19	20	23	71	59	108	117	22	41	30	51	50064.25
20	21	12	6	63	97	76	150	16	186	10	50039.25
21	22	117	30	23	76	110	199	35	20	41	49618.25
22	23	20	22	110	41	76	59	117	71	30	49618.25
23	24	2	103	83	13	120	114	189	107	116	51129.25
24	25	7	178	181	151	34	37	90	4	33	45997.25
25	26	8	32	111	28	60	61	196	58	33	47956.50
26	27	28	57	58	15	73	37	32	106	74	44720.50
27	28	27	58	32	57	37	73	106	26	15	46680.00
28	29	3	31	106	143	36	74	10	16	63	49768.00
29	30	117	22	76	35	110	23	20	81	186	49530.00
30	31	29	3	74	143	106	186	16	150	63	49566.50

```
df.head(200)
```

	Unnamed..0	code_geo	nom_geo	population	Logement.occupé.selon.période.de.construction...2016.à.2021....	Population
0	0	2401023	Les Îles-de-la-Madeleine	12190.00	3.58	
1	3	2406013	Carleton-sur-Mer	4081.00	4.22	
2	4	2407030	Saint-Léon-le-Grand	968.00	6.10	
3	5	2407035	Saint-Zénon-du-Lac-Humqui	370.00	5.26	
4	6	2407040	Sainte-Irène	369.00	0.00	
5	8	2407080	Val-Brillant	899.00	0.00	
6	10	2408035	Saint-René-de-Matane	961.00	0.00	
7	11	2408080	Baie-des-Sables	613.00	0.00	
8	12	2409015	Les Hauteurs	485.00	7.50	
9	13	2409025	Saint-Gabriel-de-Rimouski	1177.00	0.00	
10	14	2409055	Saint-Octave-de-Métis	493.00	4.65	
11	15	2410015	Saint-Narcisse-de-Rimouski	1084.00	4.04	
12	17	2410043	Rimouski	48935.00	4.09	
13	18	2410060	Saint-Valérien	906.00	3.95	
14	19	2410070	Saint-Fabien	1834.00	1.16	
15	20	2410075	Saint-Eugène-de-Ladrière	413.00	4.76	
16	22	2412015	Saint-Antonin	4338.00	7.41	
17	23	2412030	Saint-Épiphane	836.00	0.00	
18	24	2412043	L'Isle-Verte	1356.00	2.34	

```
len(merge_df)
```

```
199
```

20	27	2413090	de-	763.00		0.00

```
# On détermine un seuil c_k pour la consommation et r pour le revenu

voisins['c_k'] = voisins['median_consumption'] * 0.8

r = df['revenu'].quantile(0.40)

merge_df = voisins.merge(df, left_on='neighbor1', right_on='id')

merge_df['energetically_poor'] = (merge_df['conso_moy_menage'] < merge_df['c_k']) & (merge_df['revenu'] < r)

merge_df.head()
```

	neighbor1	neighbor2	neighbor3	neighbor4	neighbor5	neighbor6	neighbor7	neighbor8	neighbor9	neighbor10	median_income	mean
0	1	83	173	107	120	116	180	66	24	91	56548.00	
1	2	24	103	13	120	114	189	116	107	48	52332.00	

```
merge_df.tail(500)

len(merge_df)
```

```
merge_df[merge_df['nom_geo'] == 'Saint-Hyacinthe']
```

	neighbor1	neighbor2	neighbor3	neighbor4	neighbor5	neighbor6	neighbor7	neighbor8	neighbor9	neighbor10	median_income	me
119	120	116	107	48	96	136	2	13	49	146	54854.75	

```
# municipalités en situation de précarité énergétique
merge_df[merge_df['energetically_poor']].head(50)
```

	neighbor1	neighbor2	neighbor3	neighbor4	neighbor5	neighbor6	neighbor7	neighbor8	neighbor9	neighbor10	median_income	me
17	18	115	184	36	153	104	86	8	111	90	50576.00	
107	108	20	117	51	30	23	22	71	118	59	50680.75	
112	113	82	71	114	118	41	59	107	96	20	49840.00	
119	120	116	107	48	96	136	2	13	49	146	54854.75	

```
# lister pour chaque ligne de la dataframe voisins (colonnes neighbor1 à neighbour 10) et imprimer toutes les listes uniques (peu impor

import pandas as pd
def get_unique_neighbor_lists(df, id_to_find):
  unique_lists = set()
  for _, row in df.iterrows():
    neighbor_list = []
    for i in range(1, 11):
      neighbor_col = f"neighbor{i}"
      if not pd.isna(row[neighbor_col]):
        neighbor_list.append(row[neighbor_col])
    if id_to_find in neighbor_list:
      unique_lists.add(frozenset(neighbor_list))
  return [list(neighbor_list) for neighbor_list in unique_lists]

all ids = set()
```

librairies + drive + renommer colonnes

↳ *7 cellules masquées*

Variable dép : Conso % revenu

Revenu

↳ *5 cellules masquées*

Logement

Réparations majeures requises

[ ] ↳ *3 cellules masquées*

Locataire

[ ] ↳ *3 cellules masquées*

date de construction

```
%%R

model_avant_1981 = lm(pourc_rev_conso ~ logement_construit_avant_1981, data = data_sub)
summary(model_avant_1981)
```

Afficher la sortie masquée

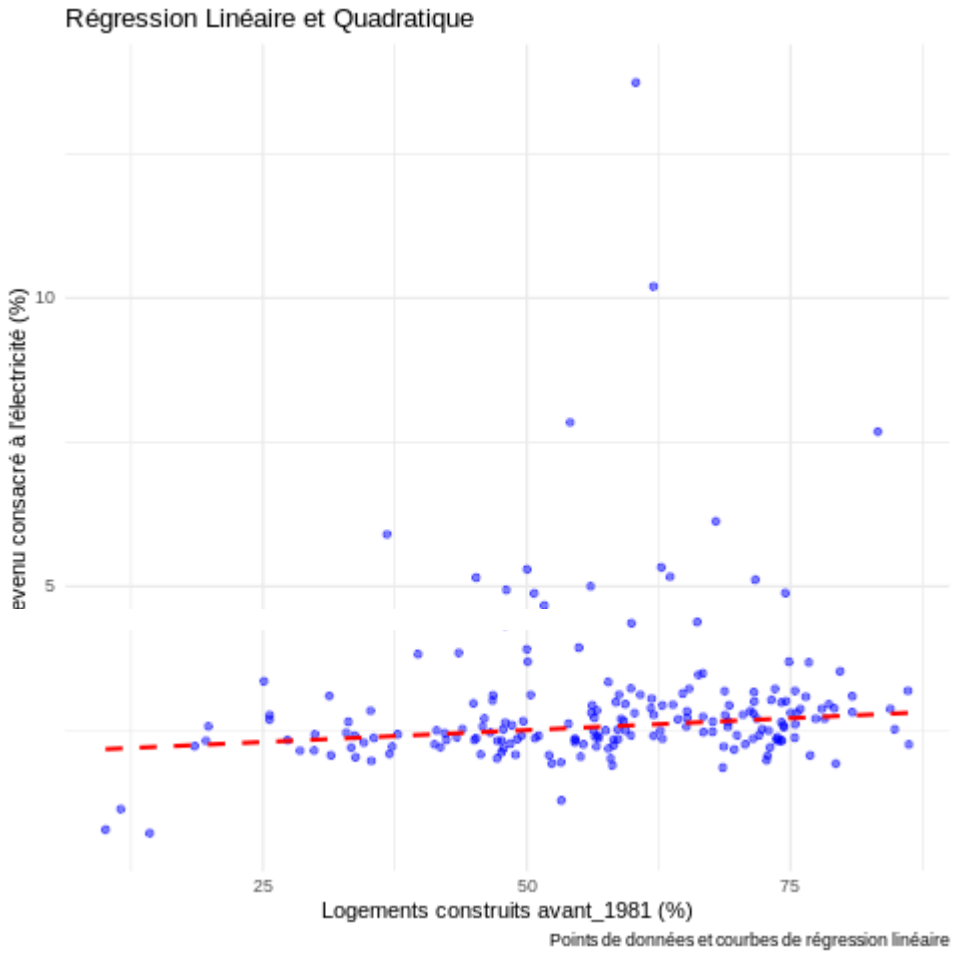
```
%%R

model_avant_1981_rob = lmRob(pourc_rev_conso ~ logement_construit_avant_1981, data = data
summary(model_avant_1981_rob)
```

```
00:00:00 left

Call:
lmRob(formula = pourc_rev_conso ~ logement_construit_avant_1981,
    data = data_sub)

Residuals:
    Min      1Q  Median      3Q     Max
-1.4907 -0.1997  0.0360  0.3832 11.1474

Coefficients:
                             Estimate Std. Error t value Pr(>|t|)
(Intercept)                  2.090478   0.117560  17.782  < 2e-16 ***
logement_construit_avant_1981 0.008267   0.001954   4.231 3.57e-05 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
WARNING:rpy2.rinterface_lib.callbacks:R[write to console]: Error in signif(x$sigma, d
  unused argument (useS4 = FALSE)

---------------------------------------------------------------------------
RRuntimeError                             Traceback (most recent call last)
<ipython-input-27-787accaf4981> in <cell line: 1>()
----> 1 get_ipython().run_cell_magic('R', '', '\nmodel_avant_1981_rob =
lmRob(pourc_rev_conso ~ logement_construit_avant_1981, data =
data_sub)\nsummary(model_avant_1981_rob)\n')

                                 8 frames
<decorator-gen-160> in R(self, line, cell, local_ns)

/usr/local/lib/python3.10/dist-packages/rpy2/rinterface.py in __call__(self, *args,
**kwargs)
    675             )
    676             if error_occured[0]:
--> 677                 raise embedded.RRuntimeError(_rinterface._geterrmessage())
    678         return res
    679
```

```
%%R

library(ggplot2)

# Ajuster les modèles
model_linear <- lmRob(pourc_rev_conso ~ logement_construit_avant_1981, data = data_sub)
#model_poly <- lmRob(consommation ~ poly(logement_construit_avant_1991, 2), data = data_s

# Créer un dataframe pour les prédictions
predicted_data <- data.frame(logement_construit_avant_1981 = seq(min(data_sub$logement_co

# Prédire les valeurs pour chaque modèle
predicted_data$consommation_linear <- predict(model_linear, newdata = predicted_data)  #
#predicted_data$consommation_poly <- predict(model_poly, newdata = predicted_data)  # Rég

# Créer le graphique
ggplot(data_sub, aes(x = logement_construit_avant_1981, y = pourc_rev_conso)) +
```

```
  geom_point(alpha = 0.5, color = "blue") +  # Points de données
  geom_line(data = predicted_data, aes(y = consommation_linear), color = "red", size = 1,
  #geom_line(data = predicted_data, aes(y = consommation_poly), color = "green", size = 1
  labs(title = "Régression Linéaire et Quadratique",
       x = "Logements construits avant_1981 (%)",
       y = "Revenu consacré à l'électricité (%)",
       caption = "Points de données et courbes de régression linéaire") +
  theme_minimal()
```

```
00:00:00 left
```



Climat

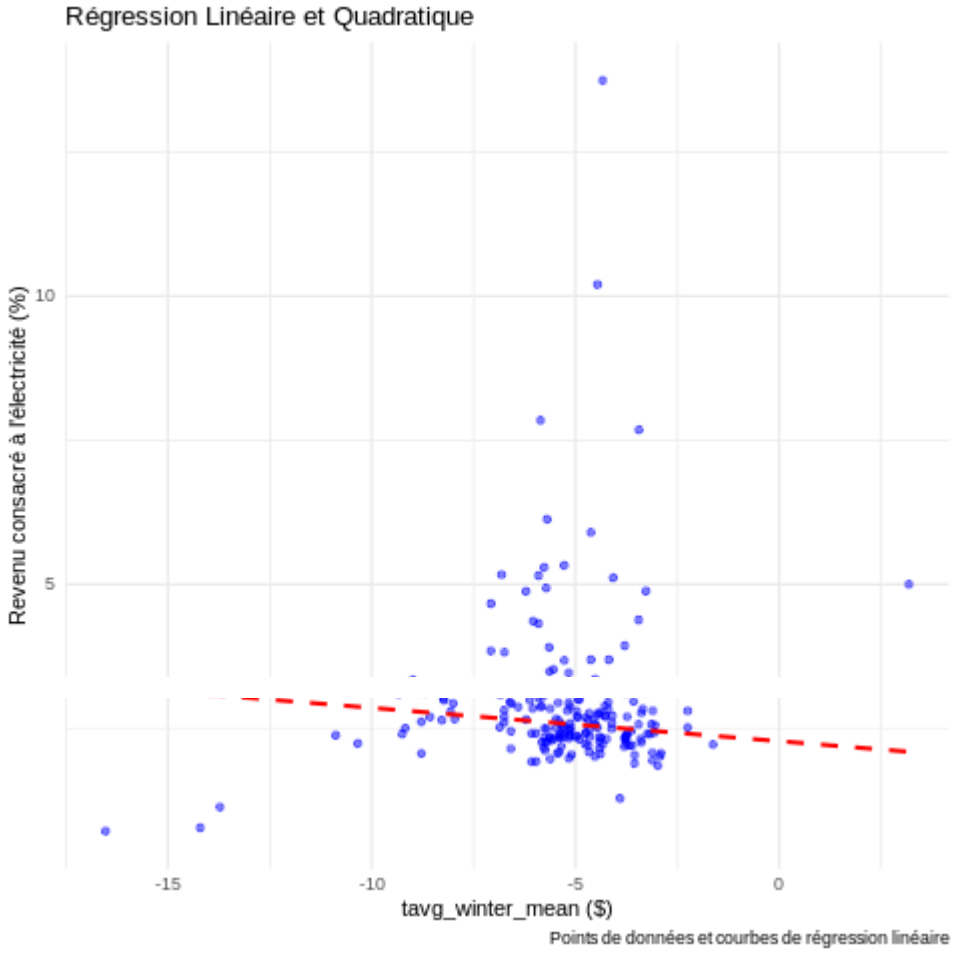
```
%%R

model_climat_rob = lmRob(pourc_rev_conso ~ tavg_winter_mean, data = data_sub)
summary(model_climat_rob)
```

```
00:00:00 left

Call:
lmRob(formula = pourc_rev_conso ~ tavg_winter_mean, data = data_sub)

Residuals:
     Min       1Q   Median       3Q      Max
-2.51467 -0.24102  0.01163  0.40192 11.20895

Coefficients:
                 Estimate Std. Error t value Pr(>|t|)
(Intercept)       2.27752    0.11556  19.709  < 2e-16 ***
tavg_winter_mean -0.05776    0.02119  -2.726  0.00699 **
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
WARNING:rpy2.rinterface_lib.callbacks:R[write to console]: Error in signif(x$sigma, d
  unused argument (useS4 = FALSE)

---------------------------------------------------------------------------
RRuntimeError                             Traceback (most recent call last)
<ipython-input-43-bcfc20c18a20> in <cell line: 1>()
----> 1 get_ipython().run_cell_magic('R', '', '\nmodel_climat_rob =
lmRob(pourc_rev_conso ~ tavg_winter_mean, data =
data_sub)\nsummary(model_climat_rob)\n')

                                 8 frames
<decorator-gen-160> in R(self, line, cell, local_ns)

/usr/local/lib/python3.10/dist-packages/rpy2/rinterface.py in __call__(self, *args,
**kwargs)
    675             )
    676             if error_occured[0]:
--> 677                 raise embedded.RRuntimeError(_rinterface._geterrmessage())
    678         return res
    679
```

```
%%R

library(ggplot2)

# Ajuster les modèles
model_linear <- lmRob(pourc_rev_conso ~ tavg_winter_mean, data = data_sub)  # Régression
#model_poly <- lmRob(consommation ~ poly(tavg_winter_mean, 2), data = data_sub)  # Régres

# Créer un dataframe pour les prédictions
predicted_data <- data.frame(tavg_winter_mean = seq(min(data_sub$tavg_winter_mean), max(d

# Prédire les valeurs pour chaque modèle
predicted_data$consommation_linear <- predict(model_linear, newdata = predicted_data)  #
#predicted_data$consommation_poly <- predict(model_poly, newdata = predicted_data)  # Rég

# Créer le graphique
ggplot(data_sub, aes(x = tavg_winter_mean, y = pourc_rev_conso)) +
  geom_point(alpha = 0.5, color = "blue") +  # Points de données
  geom_line(data = predicted_data, aes(y = consommation_linear), color = "red", size = 1,
```

```
  #geom_line(data = predicted_data, aes(y = consommation_poly), color = "green", size = 1
  labs(title = "Régression Linéaire et Quadratique",
       x = "tavg_winter_mean ($)",
       y = "Revenu consacré à l'électricité (%)",
       caption = "Points de données et courbes de régression linéaire") +
  theme_minimal()
```

```
00:00:00 left
```




```
%%R

model_climat_rob_autre = lmRob(pourc_rev_conso ~ tmax_other_max, data = data_sub)
summary(model_climat_rob_autre)
```

```
00:00:00 left

Call:
lmRob(formula = pourc_rev_conso ~ tmax_other_max, data = data_sub)

Residuals:
     Min       1Q   Median       3Q      Max
-2.10094 -0.24852  0.01104  0.42601 11.14891

Coefficients:
              Estimate Std. Error t value Pr(>|t|)
(Intercept)    3.62666    0.60866   5.958 1.16e-08 ***
tmax_other_max -0.03764    0.02196  -1.714   0.0881 .
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
WARNING:rpy2.rinterface_lib.callbacks:R[write to console]: Error in signif(x$sigma, d
  unused argument (useS4 = FALSE)

---------------------------------------------------------------------------
RRuntimeError                             Traceback (most recent call last)
<ipython-input-32-3387fa98f59a> in <cell line: 1>()
----> 1 get_ipython().run_cell_magic('R', '', '\nmodel_climat_rob_autre =
lmRob(pourc_rev_conso ~ tmax_other_max, data =
data_sub)\nsummary(model_climat_rob_autre)\n')

                                 8 frames
<decorator-gen-160> in R(self, line, cell, local_ns)

/usr/local/lib/python3.10/dist-packages/rpy2/rinterface.py in __call__(self, *args,
**kwargs)
    675             )
    676             if error_occured[0]:
--> 677                 raise embedded.RRuntimeError(_rinterface._geterrmessage())
    678         return res
    679
```

```
%%R

library(ggplot2)

# Ajuster les modèles
model_linear <- lmRob(pourc_rev_conso ~ tavg_other_mean, data = data_sub)  # Régression l
#model_poly <- lmRob(consommation ~ poly(tavg_other_mean, 2), data = data_sub)  # Régress

# Créer un dataframe pour les prédictions
predicted_data <- data.frame(tavg_other_mean = seq(min(data_sub$tavg_other_mean), max(dat

# Prédire les valeurs pour chaque modèle
predicted_data$consommation_linear <- predict(model_linear, newdata = predicted_data)  #
#predicted_data$consommation_poly <- predict(model_poly, newdata = predicted_data)  # Rég

# Créer le graphique
ggplot(data_sub, aes(x = tavg_other_mean, y = pourc_rev_conso)) +
  geom_point(alpha = 0.5, color = "blue") +  # Points de données
```

```
  geom_line(data = predicted_data, aes(y = consommation_linear), color = "red", size = 1,
  #geom_line(data = predicted_data, aes(y = consommation_poly), color = "green", size = 1
  labs(title = "Régression Linéaire et Quadratique",
       x = "tavg_other_mean ($)",
       y = "Revenu consacré à l'électricité (%)",
       caption = "Points de données et courbes de régression linéaire") +
  theme_minimal()
```

Variable dép : Conso élec

Revenu

[ ] ↳ *3 cellules masquées*

Logement

Réparations majeures requises

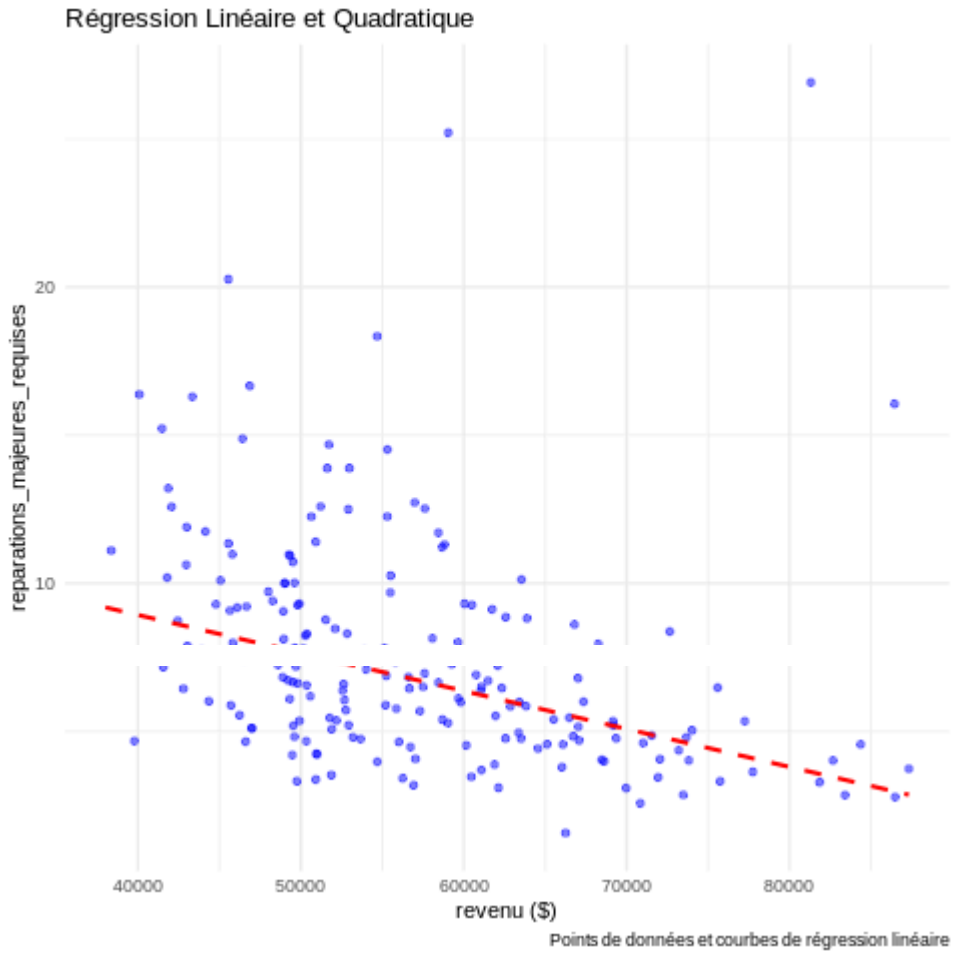
```
%%R

model_rep = lmRob(consommation_moy ~ reparations_majeures_requises, data = data_sub)
summary(model_rep)
```

```
00:00:00 left

Call:
lmRob(formula = consommation_moy ~ reparations_majeures_requises,
    data = data_sub)

Residuals:
    Min      1Q  Median      3Q     Max
-7854.2 -1515.2   107.4  2253.5 72517.2

Coefficients:
                             Estimate Std. Error t value Pr(>|t|)
(Intercept)                  21002.51     510.33  41.154  < 2e-16 ***
reparations_majeures_requises  -245.12      62.43  -3.926 0.000119 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
WARNING:rpy2.rinterface_lib.callbacks:R[write to console]: Error in signif(x$sigma, d
  unused argument (useS4 = FALSE)

---------------------------------------------------------------------------
RRuntimeError                             Traceback (most recent call last)
<ipython-input-22-cd4043f82692> in <cell line: 1>()
----> 1 get_ipython().run_cell_magic('R', '', '\nmodel_rep = lmRob(consommation_moy
~ reparations_majeures_requises, data = data_sub)\nsummary(model_rep)\n')

                                 8 frames
<decorator-gen-160> in R(self, line, cell, local_ns)

/usr/local/lib/python3.10/dist-packages/rpy2/rinterface.py in __call__(self, *args,
**kwargs)
    675             )
    676             if error_occured[0]:
--> 677                 raise embedded.RRuntimeError(_rinterface._geterrmessage())
    678         return res
    679

RRuntimeError: Error in signif(x$sigma, digits, digits, ) :
```

```
%%R

library(ggplot2)

# Ajuster les modèles
model_linear <- lmRob(reparations_majeures_requises ~ revenu , data = data_sub)  # Régres
#model_poly <- lmRob(consommation ~ poly(reparations_majeures_requises, 2), data = data_s

# Créer un dataframe pour les prédictions
predicted_data <- data.frame(revenu = seq(min(data_sub$revenu), max(data_sub$revenu), len

# Prédire les valeurs pour chaque modèle
predicted_data$consommation_linear <- predict(model_linear, newdata = predicted_data)  #
#predicted_data$consommation_poly <- predict(model_poly, newdata = predicted_data)  # Rég

# Créer le graphique
ggplot(data_sub, aes(x = revenu, y = reparations_majeures_requises)) +
  geom_point(alpha = 0.5, color = "blue") +  # Points de données
```

```
  geom_line(data = predicted_data, aes(y = consommation_linear), color = "red", size = 1,
  #geom_line(data = predicted_data, aes(y = consommation_poly), color = "green", size = 1
  labs(title = "Régression Linéaire et Quadratique",
       x = "revenu ($)",
       y = "reparations_majeures_requises",
       caption = "Points de données et courbes de régression linéaire") +
  theme_minimal()
```

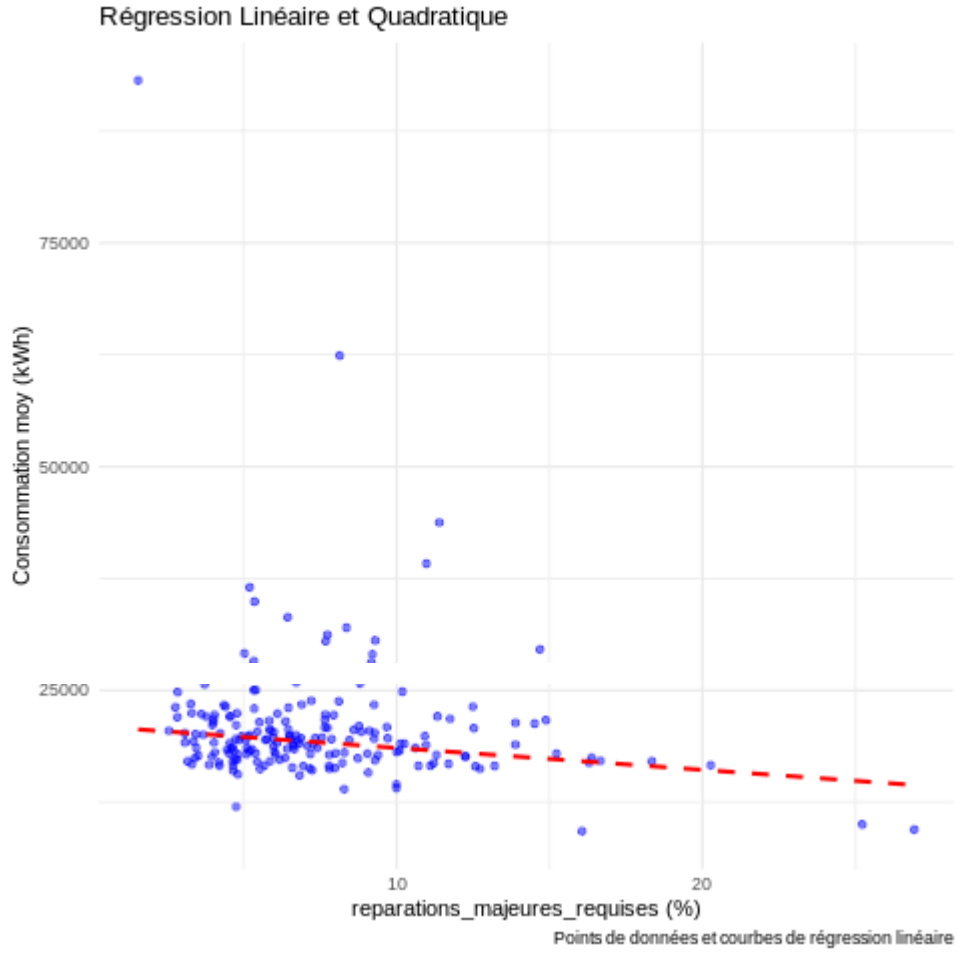
```
00:00:00 left
```




```
%%R

library(ggplot2)

# Ajuster les modèles
model_linear <- lmRob(consommation_moy ~ reparations_majeures_requises , data = data_sub)
#model_poly <- lmRob(consommation ~ poly(reparations_majeures_requises, 2), data = data_s

# Créer un dataframe pour les prédictions
predicted_data <- data.frame(reparations_majeures_requises = seq(min(data_sub$reparations

# Prédire les valeurs pour chaque modèle
predicted_data$consommation_linear <- predict(model_linear, newdata = predicted_data)  #
#predicted_data$consommation_poly <- predict(model_poly, newdata = predicted_data)  # Rég

# Créer le graphique
ggplot(data_sub, aes(x = reparations_majeures_requises, y = consommation_moy)) +
  geom_point(alpha = 0.5, color = "blue") +  # Points de données
  geom_line(data = predicted_data, aes(y = consommation_linear), color = "red", size = 1,
```

```
  #geom_line(data = predicted_data, aes(y = consommation_poly), color = "green", size = 1
  labs(title = "Régression Linéaire et Quadratique",
       x = "reparations_majeures_requises (%)",
       y = "Consommation moy (kWh)",
       caption = "Points de données et courbes de régression linéaire") +
  theme_minimal()
```

```
00:00:00 left
```



date de construction

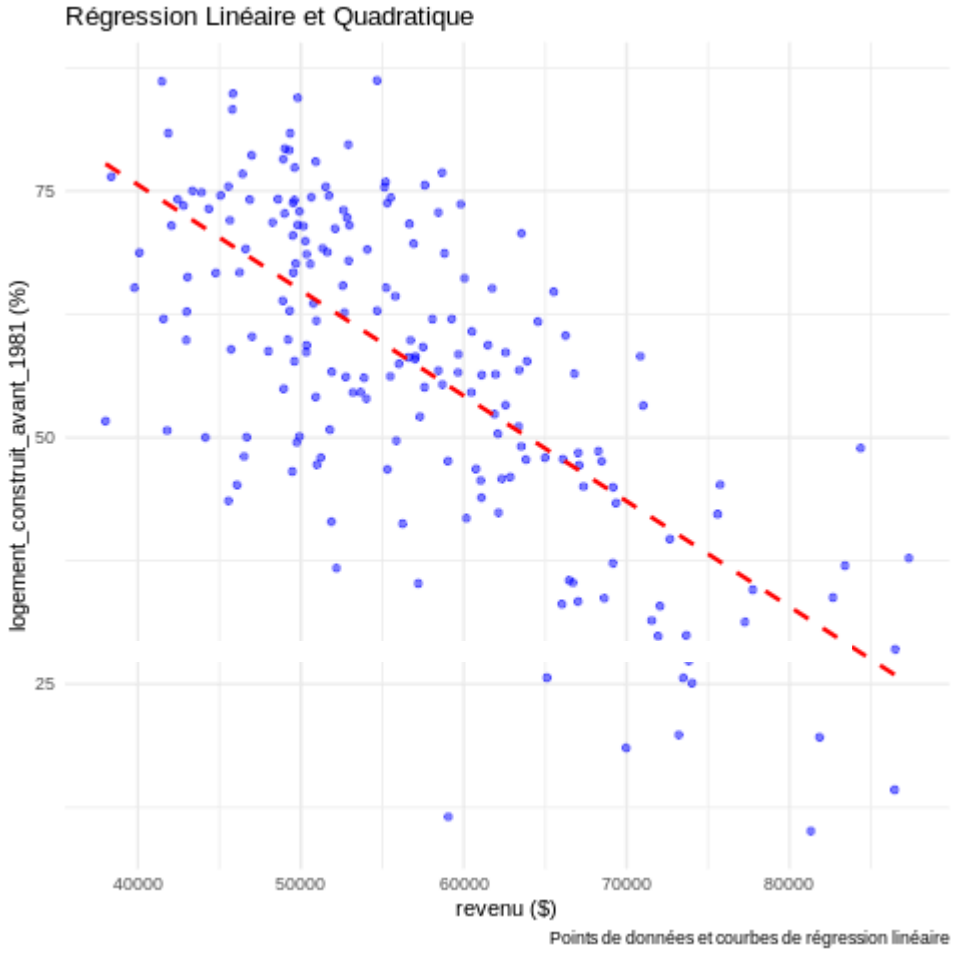
```
%%R

model_avant_1981 = lmRob(consommation_moy ~ logement_construit_avant_1981, data = data_su
summary(model_avant_1981)
```

```
00:00:00 left

Call:
lmRob(formula = consommation_moy ~ logement_construit_avant_1981,
    data = data_sub)

Residuals:
   Min     1Q Median     3Q    Max
-13590  -1260    252   1799  74083

Coefficients:
                             Estimate Std. Error t value Pr(>|t|)
(Intercept)                  23804.29     647.38   36.77  < 2e-16 ***
logement_construit_avant_1981   -78.73      10.77   -7.31 6.53e-12 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
WARNING:rpy2.rinterface_lib.callbacks:R[write to console]: Error in signif(x$sigma, d
  unused argument (useS4 = FALSE)

---------------------------------------------------------------------------
RRuntimeError                             Traceback (most recent call last)
<ipython-input-9-24bd44dece02> in <cell line: 1>()
----> 1 get_ipython().run_cell_magic('R', '', '\nmodel_avant_1981 =
lmRob(consommation_moy ~ logement_construit_avant_1981, data =
data_sub)\nsummary(model_avant_1981)\n')

                                 8 frames
<decorator-gen-160> in R(self, line, cell, local_ns)

/usr/local/lib/python3.10/dist-packages/rpy2/rinterface.py in __call__(self, *args,
**kwargs)
    675             )
    676             if error_occured[0]:
--> 677                 raise embedded.RRuntimeError(_rinterface._geterrmessage())
    678         return res
    679
```

```
%%R

library(ggplot2)

# Ajuster les modèles
model_linear <- lmRob(logement_construit_avant_1981 ~ revenu , data = data_sub)  # Régres
#model_poly <- lmRob(consommation ~ poly(reparations_majeures_requises, 2), data = data_s

# Créer un dataframe pour les prédictions
predicted_data <- data.frame(revenu = seq(min(data_sub$revenu), max(data_sub$revenu), len

# Prédire les valeurs pour chaque modèle
predicted_data$consommation_linear <- predict(model_linear, newdata = predicted_data)  #
#predicted_data$consommation_poly <- predict(model_poly, newdata = predicted_data)  # Rég

# Créer le graphique
ggplot(data_sub, aes(x = revenu, y = logement_construit_avant_1981)) +
  geom_point(alpha = 0.5, color = "blue") +  # Points de données
```

```
  geom_line(data = predicted_data, aes(y = consommation_linear), color = "red", size = 1,
  #geom_line(data = predicted_data, aes(y = consommation_poly), color = "green", size = 1
  labs(title = "Régression Linéaire et Quadratique",
       x = "revenu ($)",
       y = "logement_construit_avant_1981 (%)",
       caption = "Points de données et courbes de régression linéaire") +
  theme_minimal()
```

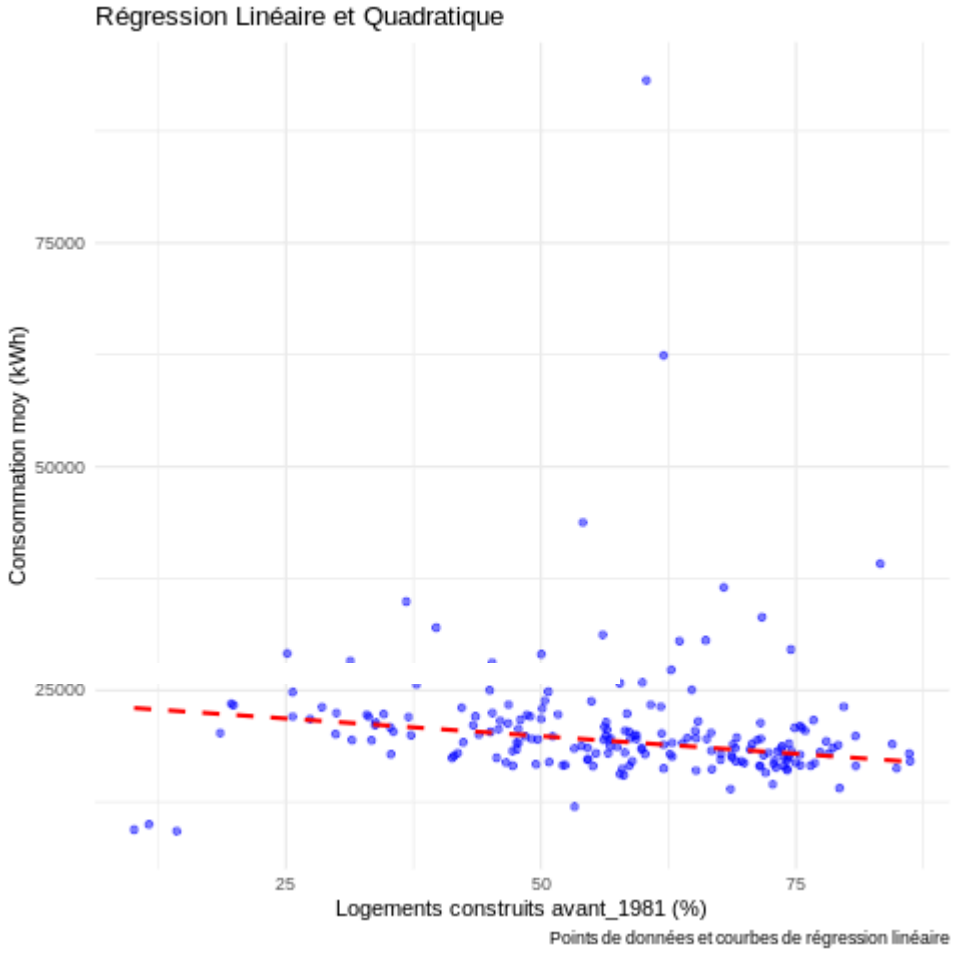
```
00:00:00 left
```




```
%%R

library(ggplot2)

# Ajuster les modèles
model_linear <- lmRob(consommation_moy ~ logement_construit_avant_1981, data = data_sub)
#model_poly <- lmRob(consommation ~ poly(revenu, 2), data = data_sub)  # Régression quadr

# Créer un dataframe pour les prédictions
predicted_data <- data.frame(logement_construit_avant_1981 = seq(min(data_sub$logement_co

# Prédire les valeurs pour chaque modèle
predicted_data$consommation_linear <- predict(model_linear, newdata = predicted_data)  #
#predicted_data$consommation_poly <- predict(model_poly, newdata = predicted_data)  # Rég

# Créer le graphique
ggplot(data_sub, aes(x = logement_construit_avant_1981, y = consommation_moy)) +
  geom_point(alpha = 0.5, color = "blue") +  # Points de données
  geom_line(data = predicted_data, aes(y = consommation_linear), color = "red", size = 1,
```

```
#geom_line(data = predicted_data, aes(y = consommation_poly), color = "green", size = 1
labs(title = "Régression Linéaire et Quadratique",
     x = "Logements construits avant_1981 (%)",
     y = "Consommation moy (kWh)",
     caption = "Points de données et courbes de régression linéaire") +
theme_minimal()
```

```
00:00:00 left
```



Climat

```
%%R

model_climat_rob = lmRob(consommation_moy ~ tavg_winter_mean, data = data_sub)
summary(model_climat_rob)
```

```
00:00:00 left

Call:
lmRob(formula = consommation_moy ~ tavg_winter_mean, data = data_sub)

Residuals:
     Min       1Q   Median       3Q      Max
-9486.13 -1764.26    32.53  2482.66 73794.42

Coefficients:
                 Estimate Std. Error t value Pr(>|t|)
(Intercept)      19555.03     766.72  25.505   <2e-16 ***
tavg_winter_mean    49.11     141.63   0.347    0.729
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
WARNING:rpy2.rinterface_lib.callbacks:R[write to console]: Error in signif(x$sigma, d
  unused argument (useS4 = FALSE)

---------------------------------------------------------------------------
RRuntimeError                             Traceback (most recent call last)
<ipython-input-17-09ff8a121856> in <cell line: 1>()
----> 1 get_ipython().run_cell_magic('R', '', '\nmodel_climat_rob =
lmRob(consommation_moy ~ tavg_winter_mean, data =
data_sub)\nsummary(model_climat_rob)\n')

                                 8 frames
<decorator-gen-160> in R(self, line, cell, local_ns)

/usr/local/lib/python3.10/dist-packages/rpy2/rinterface.py in __call__(self, *args,
**kwargs)
    675             )
    676             if error_occured[0]:
--> 677                 raise embedded.RRuntimeError(_rinterface._geterrmessage())
    678         return res
    679
```

```
%%R

library(ggplot2)

# Ajuster les modèles
model_linear <- lmRob(consommation_moy ~ tavg_winter_mean, data = data_sub)  # Régressic
#model_poly <- lmRob(consommation ~ poly(tavg_winter_mean, 2), data = data_sub)  # Régre

# Créer un dataframe pour les prédictions
predicted_data <- data.frame(tavg_winter_mean = seq(min(data_sub$tavg_winter_mean), max(

# Prédire les valeurs pour chaque modèle
predicted_data$consommation_linear <- predict(model_linear, newdata = predicted_data)  #
#predicted_data$consommation_poly <- predict(model_poly, newdata = predicted_data)  # Re

# Créer le graphique
ggplot(data_sub, aes(x = tavg_winter_mean, y = consommation_moy)) +
  geom_point(alpha = 0.5, color = "blue") +  # Points de données
  geom_line(data = predicted_data, aes(y = consommation_linear), color = "red", size = 1
```

```
#geom_line(data = predicted_data, aes(y = consommation_poly), color = "green", size =
labs(title = "Régression Linéaire et Quadratique",
     x = "tavg_winter_mean ($)",
     y = "Revenu consacré à l'électricité (%)",
     caption = "Points de données et courbes de régression linéaire") +
```

```
00:00:00 left
```

drive + librairies

```
#-------------------------------------------------------------------------------
# Set up drive
#-------------------------------------------------------------------------------
from google.colab import drive
from google.colab import files
drive.mount('/content/gdrive')
```

```
Drive already mounted at /content/gdrive; to attempt to forcibly remount, call drive.mount("/content/gdrive", force_remount=True).
```

```
#-------------------------------------------------------------------------------
# Import libaries
#-------------------------------------------------------------------------------
import pandas as pd
import numpy as np

pd.set_option('display.max_columns', None)
pd.set_option('display.max_rows', None)


pd.options.display.float_format = '{:.2f}'.format


df = pd.read_csv(r'/content/gdrive/MyDrive/mémoire/Données/voisins_poor.csv', encoding= "utf-8")


df_imput = pd.read_csv(r'/content/gdrive/MyDrive/mémoire/Données/voisins_poor_imput.csv', encoding= "utf-8")
```

description

```
df.columns
```

```
Index(['Unnamed: 0', 'neighbor1', 'neighbor2', 'neighbor3', 'neighbor4',
       'neighbor5', 'neighbor6', 'neighbor7', 'neighbor8', 'neighbor9',
       'neighbor10', 'median_income', 'mean_income', 'median_consumption',
       'mean_consumption', 'c_k', 'Unnamed..0', 'code_geo', 'nom_geo',
       'population',
       'Logement.occupé.selon.période.de.construction...2016.à.2021....',
       'Population..2011',
       'Logements.privés.occupés.par.des.résidents.habituels',
       'Nombre.moyen.de.pièces.par.logement', 'Population..2016',
       'Superficie.des.terres.en.kilomètres.carrés',
       'Taille.moyenne.des.ménages.privés', 'Taux.de.chômage',
       'Total.des.logements.privés',
       'Type.de.construction.résidentielle...Maison.individuelle.non.attenante....',
       'Type.de.construction.résidentielle...Maison.jumelée....',
       'Type.de.construction.résidentielle...Maison.en.rangée....',
       'Type.de.construction.résidentielle...Appartement.ou.plain.pied.dans.un.duplex....',
       'Type.de.construction.résidentielle...Appartement.dans.un.immeuble.de.moins.de.cinq.étages....',
       'Type.de.construction.résidentielle...Appartement.dans.un.immeuble.de.cinq.étages.ou.plus....',
       'Logement.occupé.selon.période.de.construction...1960.ou.avant....',
       'Logement.occupé.selon.période.de.construction...1961.à.1980....',
       'Logement.occupé.selon.période.de.construction...1981.à.1990....',
       'Logement.occupé.selon.période.de.construction...1991.à.2000....',
       'Logement.occupé.selon.période.de.construction...2001.à.2005....',
       'Logement.occupé.selon.période.de.construction...2006.à.2010....',
       'Logement.occupé.selon.période.de.construction...2011.à.2015....',
       'reparations_majeures_requises',
       'Ménages.privés.selon.le.mode.d.occupation...Propriétaire....',
       'locataires',
       'Ménages.privés.selon.le.mode.d.occupation...Logement.fourni.par.le.gouvernement.local....',
       'Plus.haut.diplôme...Aucun.certificat..diplôme.ou.grade....',
       'Plus.haut.diplôme...Diplôme.d.études.secondaires.ou.attestation.d.équivalence....',
       'Plus.haut.diplôme...Certificat..diplôme.ou.grade.d.études.postsecondaires....',
       'personnes_agees',
       'Ménages.dont.30...ou.plus.du.revenu.est.consacré.aux.frais.de.logement.seulement....',
       'Revenu.médian.moyen.après.impôts', 'Revenu.médian.moyen.avant.impôts',
       'Revenu.médian.des.ménages.avant.impôts', 'revenu', 'region',
       'median_total_kwh_winter', 'mean_total_kwh_winter',
       'median_total_kwh_summer', 'mean_total_kwh_summer', 'consommation_med',
       'consommation_moy', 'code_postal', 'location', 'tavg_winter_mean',
       'tmin_winter_min', 'tmax_winter_max', 'tavg_other_mean',
       'tmin_other_min', 'tmax_other_max', 'logement_construit_avant_1981',
       'id', 'conso_moy_mensuelle', 'cout_electricite_menage_an',
       'pourcentage_revenus_electricite', 'imputé..', 'conso_moy_menage',
       'energetically_poor'],
      dtype='object')
```

```
df_imput.columns
```

```
Index(['Unnamed: 0', 'neighbor1', 'neighbor2', 'neighbor3', 'neighbor4',
       'neighbor5', 'neighbor6', 'neighbor7', 'neighbor8', 'neighbor9',
       'neighbor10', 'median_income', 'mean_income', 'median_consumption',
       'mean_consumption', 'c_k', 'code_geo', 'nom_geo', 'population',
       'Logement.occupé.selon.période.de.construction...2016.à.2021....',
       'Population..2011',
       'Logements.privés.occupés.par.des.résidents.habituels',
       'Nombre.moyen.de.pièces.par.logement', 'Population..2016',
       'Superficie.des.terres.en.kilomètres.carrés',
       'Taille.moyenne.des.ménages.privés', 'Taux.de.chômage',
       'Total.des.logements.privés',
       'Type.de.construction.résidentielle...Maison.individuelle.non.attenante....',
       'Type.de.construction.résidentielle...Maison.jumelée....',
       'Type.de.construction.résidentielle...Maison.en.rangée....',
       'Type.de.construction.résidentielle...Appartement.ou.plain.pied.dans.un.duplex....',
       'Type.de.construction.résidentielle...Appartement.dans.un.immeuble.de.moins.de.cinq.étages....',
       'Type.de.construction.résidentielle...Appartement.dans.un.immeuble.de.cinq.étages.ou.plus....',
       'Logement.occupé.selon.période.de.construction...1960.ou.avant....',
       'Logement.occupé.selon.période.de.construction...1961.à.1980....',
       'Logement.occupé.selon.période.de.construction...1981.à.1990....',
       'Logement.occupé.selon.période.de.construction...1991.à.2000....',
       'Logement.occupé.selon.période.de.construction...2001.à.2005....',
       'Logement.occupé.selon.période.de.construction...2006.à.2010....',
       'Logement.occupé.selon.période.de.construction...2011.à.2015....',
       'reparations_majeures_requises',
       'Ménages.privés.selon.le.mode.d.occupation...Propriétaire....',
       'locataires',
       'Ménages.privés.selon.le.mode.d.occupation...Logement.fourni.par.le.gouvernement.local....',
       'Plus.haut.diplôme...Aucun.certificat..diplôme.ou.grade....',
       'Plus.haut.diplôme...Diplôme.d.études.secondaires.ou.attestation.d.équivalence....',
       'Plus.haut.diplôme...Certificat..diplôme.ou.grade.d.études.postsecondaires....',
       'personnes_agees',
       'Ménages.dont.30...ou.plus.du.revenu.est.consacré.aux.frais.de.logement.seulement....',
       'Revenu.médian.moyen.après.impôts', 'Revenu.médian.moyen.avant.impôts',
       'Revenu.médian.des.ménages.avant.impôts', 'revenu', 'region',
       'median_total_kwh_winter', 'mean_total_kwh_winter',
       'median_total_kwh_summer', 'mean_total_kwh_summer', 'consommation_med',
       'consommation_moy', 'code_postal', 'location', 'tavg_winter_mean',
       'tmin_winter_min', 'tmax_winter_max', 'tavg_other_mean',
       'tmin_other_min', 'tmax_other_max', 'logement_construit_avant_1981',
       'id', 'conso_moy_menage', 'energetically_poor'],
      dtype='object')
```

```
# netotyage

df = df.rename(columns={"personnes_agees": "pourcentage des personnes de plus de 65 ans"})
df_imput = df_imput.rename(columns={"personnes_agees": "pourcentage des personnes de plus de 65 ans"})
```

```
len(df)
```

```
199
```

```
del df['cout_electricite_menage_an']
```

```
# Calculer "consommation par ménage par mois" for df
df['consommation par ménage par mois'] = (df['consommation_moy'] / ((df['Population..2011'] + df['population']) / 2) * df['Taille.moyenn

# Calculer "consommation par ménage par mois" for df_imput
df_imput['consommation par ménage par mois'] = (df_imput['consommation_moy'] / ((df_imput['Population..2011'] + df_imput['population'])
```

```
# fonction qui calcule le coût en électricité par ménage

import numpy as np
def cout_electricite_menage(df):

  conso_quotidienne = df['consommation par ménage par mois'] / 30
  cout = np.where(conso_quotidienne < 40,
                  conso_quotidienne * 0.06704,
                  (40 * 0.06704) + (conso_quotidienne - 40) * 0.10342)
  return cout

df['cout electricite par menage par an'] = cout_electricite_menage(df)*30*12
df_imput['cout electricite par menage par an'] = cout_electricite_menage(df_imput)*30*12
```

```
df_imput.head()
```

	Unnamed: 0	neighbor1	neighbor2	neighbor3	neighbor4	neighbor5	neighbor6	neighbor7	neighbor8	neighbor9	neighbor10	median_ir
0	0	1	337	154	168	166	279	313	194	212	207	557
1	1	2	66	201	84	121	101	272	76	175	198	549
2	2	3	28	51	139	92	340	44	193	16	52	499
3	3	4	244	36	357	189	305	18	307	168	212	513
4	4	5	43	192	45	257	21	53	14	163	137	491

```
# calculer une colonne qui me donne le pourcentage de revenus après impôts des ménages consacré à l'électricité

def pourcentage_revenus_electricite(df):
  return (df['cout electricite par menage par an'] / df['revenu']) * 100

df["pourcentage des revenus des ménages consacrés à l'electricite"] = pourcentage_revenus_electricite(df)
df_imput["pourcentage des revenus des ménages consacrés à l'electricite"] = pourcentage_revenus_electricite(df_imput)

df.head()
```

	Unnamed: 0	neighbor1	neighbor2	neighbor3	neighbor4	neighbor5	neighbor6	neighbor7	neighbor8	neighbor9	neighbor10	median_ir
0	0	1	83	173	107	120	116	180	66	24	91	565
1	1	2	24	103	13	120	114	189	116	107	48	523
2	2	3	29	106	31	143	16	36	10	74	58	491
3	3	4	37	177	57	7	58	27	192	21	15	492
4	4	5	55	172	193	75	170	72	65	95	70	584

```
# calculer la médiane de pourcentage_revenus_electricite

mediane_pourcentage_revenus = df["pourcentage des revenus des ménages consacrés à l'electricite"].median()
mediane_pourcentage_revenus_imput = df_imput["pourcentage des revenus des ménages consacrés à l'electricite"].median()

# Filtrage des municipalités dans df
df_filtre_df = df[df["pourcentage des revenus des ménages consacrés à l'electricite"] >= 2 * mediane_pourcentage_revenus]

# Filtrage des municipalités dans df_imput
df_filtre_df_imput = df_imput[df_imput["pourcentage des revenus des ménages consacrés à l'electricite"] >= 2 * mediane_pourcentage_rever

print(f"Médiane de pourcentage_revenus_electricite: {mediane_pourcentage_revenus}")

print("\nMunicipalités dans df avec un pourcentage_revenus_electricite au moins deux fois supérieur à la médiane:")
```

```
print(df_filtre_df[['nom_geo', "pourcentage des revenus des ménages consacrés à l'electricite"]])


print("\nMunicipalités dans df_imput avec un pourcentage_revenus_electricite au moins deux fois supérieur à la médiane:")
print(df_filtre_df_imput[['nom_geo', "pourcentage des revenus des ménages consacrés à l'electricite"]])
```

```
Médiane de pourcentage_revenus_electricite: 2.685486031553853

Municipalités dans df avec un pourcentage_revenus_electricite au moins deux fois supérieur à la médiane:
                nom_geo  \
81             Valcourt
85            Stanstead
86               Hatley
166      Mont-Tremblant
167  Notre-Dame-du-Laus
175    Lac-Sainte-Marie
198        Plessisville

     pourcentage des revenus des ménages consacrés à l'electricite
81                                                   10.11
85                                                    7.64
86                                                   13.84
166                                                   6.42
167                                                   5.73
175                                                   8.31
198                                                   6.02

Municipalités dans df_imput avec un pourcentage_revenus_electricite au moins deux fois supérieur à la médiane:
                     nom_geo  \
141  Saints-Martyrs-Canadiens
151                 Valcourt
156                Stanstead
157                   Hatley
159                   Austin
245    Notre-Dame-de-la-Merci
246               Saint-Donat
298            Wentworth-Nord
301           Ivry-sur-le-Lac
303            Mont-Tremblant
305        Notre-Dame-du-Laus
315                 Lac-Simon
317             Lac-des-Plages
318                    Duhamel
322          Lac-Sainte-Marie
363              Plessisville

     pourcentage des revenus des ménages consacrés à l'electricite
141                                                   6.89
151                                                  10.11
156                                                   7.64
157                                                  13.84
159                                                   6.23
245                                                   8.72
246                                                   8.50
298                                                   7.91
301                                                   9.49
303                                                   6.42
305                                                   5.73
315                                                   7.68
317                                                   6.97
318                                                   6.81
322                                                   8.31
363                                                   6.02
```

```
df.columns
```

```
Index(['Unnamed: 0', 'neighbor1', 'neighbor2', 'neighbor3', 'neighbor4',
       'neighbor5', 'neighbor6', 'neighbor7', 'neighbor8', 'neighbor9',
       'neighbor10', 'median_income', 'mean_income', 'median_consumption',
       'mean_consumption', 'c_k', 'Unnamed..0', 'code_geo', 'nom_geo',
       'population',
       'Logement.occupé.selon.période.de.construction...2016.à.2021....',
       'Population..2011',
       'Logements.privés.occupés.par.des.résidents.habituels',
       'Nombre.moyen.de.pièces.par.logement', 'Population..2016',
       'Superficie.des.terres.en.kilomètres.carrés',
       'Taille.moyenne.des.ménages.privés', 'Taux.de.chômage',
       'Total.des.logements.privés',
       'Type.de.construction.résidentielle...Maison.individuelle.non.attenante....',
       'Type.de.construction.résidentielle...Maison.jumelée....',
       'Type.de.construction.résidentielle...Maison.en.rangée....',
       'Type.de.construction.résidentielle...Appartement.ou.plain.pied.dans.un.duplex....',
       'Type.de.construction.résidentielle...Appartement.dans.un.immeuble.de.moins.de.cinq.étages....',
       'Type.de.construction.résidentielle...Appartement.dans.un.immeuble.de.cinq.étages.ou.plus....',
       'Logement.occupé.selon.période.de.construction...1960.ou.avant....',
       'Logement.occupé.selon.période.de.construction...1961.à.1980....',
       'Logement.occupé.selon.période.de.construction...1981.à.1990....',
       'Logement.occupé.selon.période.de.construction...1991.à.2000....',
       'Logement.occupé.selon.période.de.construction...2001.à.2005....',
```

```
       'Logement.occupé.selon.période.de.construction...2006.à.2010....',
       'Logement.occupé.selon.période.de.construction...2011.à.2015....',
       'reparations_majeures_requises',
       'Ménages.privés.selon.le.mode.d.occupation...Propriétaire....',
       'locataires',
       'Ménages.privés.selon.le.mode.d.occupation...Logement.fourni.par.le.gouvernement.local....',
       'Plus.haut.diplôme...Aucun.certificat..diplôme.ou.grade....',
       'Plus.haut.diplôme...Diplôme.d.études.secondaires.ou.attestation.d.équivalence....',
       'Plus.haut.diplôme...Certificat..diplôme.ou.grade.d.études.postsecondaires....',
       'pourcentage des personnes de plus de 65 ans',
       'Ménages.dont.30...ou.plus.du.revenu.est.consacré.aux.frais.de.logement.seulement....',
       'Revenu.médian.moyen.après.impôts', 'Revenu.médian.moyen.avant.impôts',
       'Revenu.médian.des.ménages.avant.impôts', 'revenu', 'region',
       'median_total_kwh_winter', 'mean_total_kwh_winter',
       'median_total_kwh_summer', 'mean_total_kwh_summer', 'consommation_med',
       'consommation_moy', 'code_postal', 'location', 'tavg_winter_mean',
       'tmin_winter_min', 'tmax_winter_max', 'tavg_other_mean',
       'tmin_other_min', 'tmax_other_max', 'logement_construit_avant_1981',
       'id', 'conso_moy_mensuelle', 'pourcentage_revenus_electricite',
       'imputé..', 'conso_moy_menage', 'energetically_poor',
       'consommation par ménage par mois',
       'cout electricite par menage par an',
       'pourcentage des revenus des ménages consacrés à l'electricite'],
      dtype='object')


# Calculer "Consommation d'électricité par ménage par an" for df
df["Consommation d'électricté par ménage par an (kWh)"] = df['consommation par ménage par mois'] * 12

# Calculer "Consommation d'électricité par ménage par an" for df_imput
df_imput["Consommation d'électricté par ménage par an (kWh)"] = df_imput['consommation par ménage par mois'] * 12


# nettoyage

# Renommer colonnes df
df = df.rename(columns={
    "population": "Population en 2021",
    "revenu": "Revenu médian des ménages après impôts ($)",
    "Taille.moyenne.des.ménages.privés": "Taille moyenne des ménages",
    "pourcentage des revenus des ménages consacrés à l'electricite": "Pourcentage des revenus des ménages consacrés à l'électricité",
    'cout electricite par menage par an': "Cout de l'électricité par ménage par an",
    'logement_construit_avant_1981': 'Pourcentage de logements construits avant 1981',
    'reparations_majeures_requises': 'Pourcentage de logement néccessitant des réparations majeures',
    'tavg_winter_mean': 'Température moyenne en hiver (°C)',
    "pourcentage des personnes de plus de 65 ans": "Pourcentage des personnes âgées de plus de 65 ans"
})

# Renommer colonnes df_imput
df_imput = df_imput.rename(columns={
    "population": "Population en 2021",
    "Taille.moyenne.des.ménages.privés": "Taille moyenne des ménages",
    "revenu": "Revenu médian des ménages après impôts ($)",
    "pourcentage des revenus des ménages consacrés à l'electricite": "Pourcentage des revenus des ménages consacrés à l'électricité",
    'cout electricite par menage par an': "Cout de l'électricité par ménage par an",
    'logement_construit_avant_1981': 'Pourcentage de logements construits avant 1981',
    'reparations_majeures_requises': 'Pourcentage de logement néccessitant des réparations majeures',
    'tavg_winter_mean': 'Température moyenne en hiver (°C)',
    "pourcentage des personnes de plus de 65 ans": "Pourcentage des personnes âgées de plus de 65 ans"
})


# nettoyage

df = df[df["Revenu médian des ménages après impôts ($)"] <= 90000]

df_imput = df_imput[df_imput["Revenu médian des ménages après impôts ($)"] <= 90000]


# pstatistiques descriptives

def descriptive_stats_table(df, variables):
  stats_table = pd.DataFrame(columns=['Variable', 'Moyenne', 'Médiane', 'Ecart-Type', 'Minimum', 'Maximum'])

  for variable in variables:
    if variable in df.columns:
      distribution = df[variable].describe()
      mean = df[variable].mean()
      median = df[variable].median()
      std_dev = df[variable].std()
      num_values = df[variable].count()
      minimum = df[variable].min()  # Call the min function
      maximum = df[variable].max()  # Call the max function

      stats_table = pd.concat([stats_table, pd.DataFrame({'Variable': [variable],
                                                          'Moyenne': [mean],
```

```
                                                'Médiane': [median],
                                                'Ecart-Type': [std_dev],
                                                'Nombre de valeurs non-nulles': [num_values],
                                                'Minimum': [minimum],
                                                'Maximum': [maximum]})],
                                          ignore_index=True)
    else:
      print(f"Variable '{variable}' not found in the DataFrame.")

  return stats_table

# Specify the variables you want to analyze
variables_to_analyze = ['Population en 2021',
                        "Revenu médian des ménages après impôts ($)",
                       'Taille moyenne des ménages',
                       "Pourcentage des revenus des ménages consacrés à l'électricité",
                        "Cout de l'électricité par ménage par an",
                        "Consommation d'électricté par ménage par an (kWh)",
                        'Pourcentage de logements construits avant 1981',
                         'Pourcentage de logement néccessitant des réparations majeures',
                        'Température moyenne en hiver (°C)',
                         'Pourcentage des personnes âgées de plus de 65 ans']

# Generate and print the descriptive statistics table
descriptive_table = descriptive_stats_table(df, variables_to_analyze)
descriptive_table
```

```
<ipython-input-20-e00d854f2f2f>:16: FutureWarning: The behavior of DataFrame concatenation with empty or all-NA entries is deprecate
  stats_table = pd.concat([stats_table, pd.DataFrame({'Variable': [variable],
```

	Variable	Moyenne	Médiane	Ecart-Type	Minimum	Maximum	Nombre de valeurs non-nulles
0	Population en 2021	12307.91	2323.00	39457.36	285.00	438366.00	195.00
1	Revenu médian des ménages après impôts ($)	56631.71	55200.00	10204.01	38016.00	87320.00	195.00
2	Taille moyenne des ménages	2.25	2.25	0.20	1.85	2.70	195.00
3	Pourcentage des revenus des ménages consacrés ...	3.02	2.69	1.33	1.33	13.84	195.00
4	Cout de l'électricité par ménage par an	1693.33	1520.07	793.45	832.32	9171.20	195.00
5	Consommation d'électricté par ménage par an (kWh)	21433.91	19763.55	7677.44	12415.34	93744.70	195.00
6	Pourcentage de logements construits avant 1981	57.89	58.63	15.13	18.54	86.21	195.00
7	Pourcentage de logement néccessitant des répar...	7.31	6.59	3.32	1.56	20.27	195.00
8	Température moyenne en hiver (°C)	-5.18	-5.15	1.66	-10.89	3.18	195.00
9	Pourcentage des personnes âgées de plus de 65 ans	12.76	12.46	3.70	6.13	21.97	195.00

```
descriptive_table_imput = descriptive_stats_table(df_imput, variables_to_analyze)
descriptive_table_imput
```

```
<ipython-input-20-e00d854f2f2f>:16: FutureWarning: The behavior of DataFrame concatenation with empty or all-NA entries is deprecate
  stats_table = pd.concat([stats_table, pd.DataFrame({'Variable': [variable],
```

	Variable	Moyenne	Médiane	Ecart-Type	Minimum	Maximum	Nombre de valeurs non-nulles
0	Population en 2021	12280.59	2037.50	45445.32	277.00	549459.00	356.00
1	Revenu médian des ménages après impôts ($)	56482.27	54916.00	9691.62	38016.00	87320.00	356.00
2	Taille moyenne des ménages	2.25	2.25	0.21	1.75	2.85	356.00
3	Pourcentage des revenus des ménages consacrés ...	3.07	2.70	1.34	1.33	13.84	356.00
4	Cout de l'électricité par ménage par an	1712.03	1525.42	754.77	741.73	9171.20	356.00
5	Consommation d'électricté par ménage par an (kWh)	21610.70	19815.24	7309.11	11063.98	93744.70	356.00
6	Pourcentage de logements construits avant 1981	58.58	59.91	14.17	18.54	86.21	356.00
7	Pourcentage de logement néccessitant des répar...	7.23	6.55	3.11	1.35	20.27	356.00
8	Température moyenne en hiver (°C)	-5.15	-5.21	1.92	-10.90	3.81	356.00
9	Pourcentage des personnes âgées de plus de 65 ans	12.99	12.61	3.89	3.15	27.87	356.00

```
# tableau descriptive_table avec une ligne pour df afin de comparer 2M et k-NN

municipalites_df = ["Saint-Épiphane", "Louiseville", "Berthierville", "Saint-Hyacinthe"]
municipalites_df_2M = ["Otter Lake", "Lac-Sainte-Marie", "Notre-Dame-du-Laus", "Boileau", "Mont-Tremblant", "Plessisville", "Valcourt",

# Vérification de la présence des municipalités
missing_muni_df = set(municipalites_df) - set(df['nom_geo'])
```

```
missing_muni_df_2M = set(municipalites_df_2M) - set(df['nom_geo'])

if missing_muni_df:
    print(f"Municipalités non trouvées dans df: {missing_muni_df}")
if missing_muni_df_2M:
    print(f"Municipalités non trouvées dans df_2M: {missing_muni_df_2M}")

# Création d'un tableau descriptif pour les municipalités sélectionnées
descriptive_table_final = pd.DataFrame()

# Filtrer les données pour les municipalités ciblées dans df
filtered_df = df[df['nom_geo'].isin(municipalites_df)]

# Calcul des statistiques descriptives pour les municipalités sans valeurs imputées
for variable in variables_to_analyze: # Nomme ces variables selon celles présentes dans ton DataFrame
    values = filtered_df[variable].agg(['mean', 'median', 'std', 'min', 'max', 'count'])

    # Création de la première ligne du tableau
    new_row = {
        'Variable': variable,
        'Moyenne': values['mean'],
        'Médiane': values['median'],
        'Ecart-Type': values['std'],
        'Minimum': values['min'],
        'Maximum': values['max'],
        'Nombre de valeurs non-nulles': values['count']
    }

    descriptive_table_final = pd.concat([descriptive_table_final, pd.DataFrame([new_row])])

    # Ajout de la ligne "(imputé)" pour les municipalités sélectionnées dans df_imput
    new_row_2M = new_row.copy()
    new_row_2M['Variable'] = f"{variable} (2M)"

    # Agrégation pour les municipalités de df_imput
    values_2M = df[df['nom_geo'].isin(municipalites_df_2M)][variable].agg(['mean', 'median', 'std', 'min', 'max', 'count'])
    new_row_2M.update({
        'Moyenne': values_2M['mean'],
        'Médiane': values_2M['median'],
        'Ecart-Type': values_2M['std'],
        'Minimum': values_2M['min'],
        'Maximum': values_2M['max'],
        'Nombre de valeurs non-nulles': values_2M['count']
    })

    descriptive_table_final = pd.concat([descriptive_table_final, pd.DataFrame([new_row_2M])])

# Affichage du tableau final
descriptive_table_final
```

	Variable	Moyenne	Médiane	Ecart-Type	Minimum	Maximum	Nombre de valeurs non-nulles
0	Population en 2021	17450.25	5863.00	26658.77	836.00	57239.00	4.00
0	Population en 2021 (2M)	2435.45	1035.00	3274.85	351.00	10992.00	10.00
0	Revenu médian des ménages après impôts ($)	45001.88	45299.50	5046.49	39794.00	49614.50	4.00
0	Revenu médian des ménages après impôts ($) (2M)	51273.40	50848.00	6840.32	42997.50	66254.00	10.00
0	Taille moyenne des ménages	2.04	2.00	0.15	1.90	2.25	4.00
0	Taille moyenne des ménages (2M)	2.11	2.15	0.18	1.85	2.35	10.00
0	Pourcentage des revenus des ménages consacrés ...	2.49	2.56	0.47	1.91	2.93	4.00
0	Pourcentage des revenus des ménages consacrés ...	7.37	6.22	2.79	5.08	13.84	10.00
0	Cout de l'électricité par ménage par an	1103.48	1131.02	119.88	935.51	1216.38	4.00
0	Cout de l'électricité par ménage par an (2M)	3924.90	3269.73	2131.98	2207.82	9171.20	10.00
0	Consommation d'électricté par ménage par an (kWh)	15696.21	16001.64	1232.68	13954.56	16827.02	4.00
0	Consommation d'électricté par ménage par an (k...	43016.55	36681.54	20614.80	26413.56	93744.70	10.00
0	Pourcentage de logements construits avant 1981	66.06	63.61	9.33	57.74	79.29	4.00
0	Pourcentage de logements construits avant 1981...	58.60	61.18	12.92	36.78	83.27	10.00
0	Pourcentage de logement néccessitant des répar...	6.66	5.99	2.50	4.67	10.00	4.00
0	Pourcentage de logement néccessitant des répar...	8.06	8.66	3.24	1.56	11.89	10.00
0	Température moyenne en hiver (°C)	-4.60	-4.32	0.95	-5.97	-3.81	4.00
0	Température moyenne en hiver (°C) (2M)	-5.22	-5.49	0.99	-6.81	-3.44	10.00
0	Pourcentage des personnes âgées de plus de 65 ans	14.69	14.40	2.94	11.65	18.31	4.00
0	Pourcentage des personnes âgées de plus de 65 ...	16.75	15.55	3.49	12.37	21.68	10.00

```
# tableau descriptive_table avec une ligne pour df_imput afin de comparer 2M et k-NN

municipalites_df = ["Saint-Épiphane", "Sainte-Apolline-de-Patton", "Lac-Bouchette", "Berthierville", "Saint-Hyacinthe", "Béthanie"]
municipalites_df_2M = ["Otter Lake", "Lac-Sainte-Marie", "Notre-Dame-du-Laus", "Boileau", "Mont-Tremblant", "Plessisville", "Valcourt"
                       "Hatley", "Stanstead", "Sainte-Praxède", "Saint-Donat", "Saints-Martyrs-Canadiens", "Lac-Simon", "Lac-des-Plage
                       "Notre-Dame-de-la-Merci", "Duhamel", "Wentworth-Nord", "Austin"]

# Vérification de la présence des municipalités
missing_muni_df = set(municipalites_df) - set(df_imput['nom_geo'])
missing_muni_df_2M = set(municipalites_df_2M) - set(df_imput['nom_geo'])

if missing_muni_df:
    print(f"Municipalités non trouvées dans df: {missing_muni_df}")
if missing_muni_df_2M:
    print(f"Municipalités non trouvées dans df_2M: {missing_muni_df_2M}")

# Création d'un tableau descriptif pour les municipalités sélectionnées
descriptive_table_final = pd.DataFrame()

# Filtrer les données pour les municipalités ciblées dans df
filtered_df = df_imput[df_imput['nom_geo'].isin(municipalites_df)]

# Calcul des statistiques descriptives pour les municipalités sans valeurs imputées
```

```
for variable in variables_to_analyze: # Nomme ces variables selon celles présentes dans ton DataFrame
    values = filtered_df[variable].agg(['mean', 'median', 'std', 'min', 'max', 'count'])

    # Création de la première ligne du tableau
    new_row = {
        'Variable': variable,
        'Moyenne': values['mean'],
        'Médiane': values['median'],
        'Ecart-Type': values['std'],
        'Minimum': values['min'],
        'Maximum': values['max'],
        'Nombre de valeurs non-nulles': values['count']
    }

    descriptive_table_final = pd.concat([descriptive_table_final, pd.DataFrame([new_row])])

    # Ajout de la ligne "(imputé)" pour les municipalités sélectionnées dans df_imput
    new_row_2M = new_row.copy()
    new_row_2M['Variable'] = f"{variable} (2M)"

    # Agrégation pour les municipalités de df_imput
    values_2M = df_imput[df_imput['nom_geo'].isin(municipalites_df_2M)][variable].agg(['mean', 'median', 'std', 'min', 'max', 'count']
    new_row_2M.update({
        'Moyenne': values_2M['mean'],
        'Médiane': values_2M['median'],
        'Ecart-Type': values_2M['std'],
        'Minimum': values_2M['min'],
        'Maximum': values_2M['max'],
        'Nombre de valeurs non-nulles': values_2M['count']
    })

    descriptive_table_final = pd.concat([descriptive_table_final, pd.DataFrame([new_row_2M])])

# Affichage du tableau final
descriptive_table_final
```

	Variable	Moyenne	Médiane	Ecart-Type	Minimum	Maximum	Nombre de valeurs non-nulles
0	Population en 2021	10749.83	1009.50	22823.95	350.00	57239.00	6.00
0	Population en 2021 (2M)	1993.53	1049.00	2589.31	277.00	10992.00	18.00
0	Revenu médian des ménages après impôts ($)	46164.58	46001.75	4687.42	41544.00	52256.00	6.00
0	Revenu médian des ménages après impôts ($) (2M)	51359.00	50848.00	6219.78	42997.50	66254.00	18.00
0	Taille moyenne des ménages	2.12	2.08	0.22	1.90	2.50	6.00
0	Taille moyenne des ménages (2M)	2.01	1.98	0.18	1.75	2.35	18.00
0	Pourcentage des revenus des ménages consacrés ...	2.46	2.32	0.52	1.91	3.26	6.00
0	Pourcentage des revenus des ménages consacrés ...	7.41	6.93	2.11	5.08	13.84	18.00
0	Cout de l'électricité par ménage par an	1128.17	1182.64	202.98	849.17	1401.89	6.00
0	Cout de l'électricité par ménage par an (2M)	3880.88	3508.49	1582.35	2207.82	9171.20	18.00
0	Consommation d'électricté par ménage par an (kWh)	15846.36	16500.82	2164.86	12666.60	18620.75	6.00
0	Consommation d'électricté par ménage par an (k...	42590.89	38990.15	15300.23	26413.56	93744.70	18.00
0	Pourcentage de logements construits avant 1981...	55.75	54.40	10.42	36.78	83.27	18.00
0	Pourcentage de logement néccessitant des répar...	10.51	10.31	4.47	4.82	17.62	6.00
0	Pourcentage de logement néccessitant des répar...	8.15	8.08	3.46	1.56	16.25	18.00
0	Température moyenne en hiver (°C)	-5.67	-5.21	1.80	-8.57	-3.81	6.00
0	Température moyenne en hiver (°C) (2M)	-4.16	-5.10	3.33	-9.45	3.81	18.00
0	Pourcentage des personnes âgées de plus de 65 ans	14.92	14.40	3.32	11.65	19.41	6.00
0	Pourcentage des personnes âgées de plus de 65 ...	18.45	18.86	3.76	12.37	26.27	18.00