HEC MONTRÉAL

Stacking pour les modèles Uplift

par

Thomas Plourde

Sciences de la gestion

(Option Intelligence d'affaires)

Mémoire présenté en vue de l'obtention du grade

de maîtrise ès sciences en gestion

(M.Sc.)

Décembre 2018

©Thomas Plourde, 2018

Résumé

La modélisation Uplift présente une problématique unique qui provient du fait que la variable-cible que l'on tente de modéliser est inconnue même sur le jeu d'entraînement. En effet, il est simplement impossible de connaître la réponse d'un même individu à la fois dans les groupes traitement et contrôle. En dépit de nombreuses recherches sur cette problématique, la modélisation Uplift présente toujours un potentiel intéressant pour améliorer sa performance.

Plusieurs solutions pour modéliser de telles situations en dépit de ce problème ont été proposées dans la littérature, chacune ayant des avantages et des inconvénients. L'approche à deux modèles tente de modéliser séparément sur les groupes traitement et contrôle la variable-réponse en fonction des variables explicatives. En théorie, la différence entre les prédictions de ces deux modèles correspond au lift. La méthode Class Transformation consiste pour sa part à transformer la variable-cible pour tenir compte de l'indicateur d'assignation de traitement, pour ensuite entraîner un algorithme d'apprentissage automatique standard qui modélise le lift. Une dernière méthode consiste à construire de nouveaux algorithmes capables de modéliser directement le lift.

Le présent projet de mémoire tente de développer et d'évaluer plusieurs approches pour combiner les trois méthodes présentées ci-dessus en utilisant une technique de stacking. L'hypothèse de départ est que cette méthode de combinaison permettra d'améliorer la fiabilité et la performance de modèles Uplift. Par le fait même, puisque le stacking requiert l'usage d'un estimateur du lift réel pour calibrer le modèle de second niveau, cette recherche contribue également à la quête d'un estimateur fiable du lift pour solutionner le problème Uplift.

Les résultats démontrent que l'approche combinatoire par stacking permet d'améliorer la stabilité de la performance de modèles Uplift sur une variété de jeux de données. Dans plusieurs cas, les deux approches de stacking, par strates et par appariement, ont une performance supérieure à chacun des modèles sous-jacents utilisés. Les gains de performance varient de négligeables sur certaines données, à substantiels (20+% supérieur) dans d'autres contextes.

Mots-clés : Levier, uplift, méthodes d'ensemble, apprentissage automatique, stacking

Abstract

Uplift modelling presents a unique challenge where the real quantity we are attempting to model is inherently unknown because an individual cannot be both in the treatment and control groups. Despite the solutions brought forward in the existing literature, there is still significant room for improvement to its performance.

Three main branches exist in Uplift modelling, each representing a distinct approach to finding a solution to the Uplift problem. The two-model approach aims to do so by modelling the response variable as a function of the covariates separately on the treatment and control groups. The Class Transformation approach modifies the response variable to account for the treatment effect, and models this modified variable as a function of the covariates. The last approach attempts to use customized machine learning algorithms to model Uplift directly.

Each of these approaches has distinct advantages and drawbacks. This project attempts to develop and evaluate several approaches to combining these three methods using stacked ensemble models. It is hypothesized that using such combinations would improve the performance and reliability of Uplift models and contribute to the quest for an ultimate solution to the Uplift problem.

The results show that the stacking approaches tested can significantly improve the stability of Uplift models in comparison to the underlying techniques. In many cases, an overall performance improvement has also been observed. The performance gains vary from negligible on some datasets to substantial (20+% improvement) on others. Further study will be required to ascertain what factors influence the size of the gains.

Keywords: Uplift, stacking, ensemble models, machine learning

Table des matières

Résumé 1
Abstract 3
Table des matières 5
Liste des tableaux et figures 9
Liste des abréviations 12
Remerciements 14
1. Introduction 16
2. Revue de littérature 18
2.1. Contexte 18
2.2. Modélisation Uplift 19
2.2.1. Approche à deux modèles 20
2.2.2. Approche *Class Transformation* 22
2.2.3. Approche de modélisation directe du lift 23
2.2.4. Évaluation de la performance 24
2.3. Stacking 27
2.3.1. Introduction 27
2.3.2. Considérations méthodologiques 28
3. Méthodologie 30
3.1. Approche générale 30
3.2. Simulation de données 30
3.2.1. Simulation du package Uplift (sim_pte) 31
3.2.2. Simulation avec variation de la probabilité de traitement 33
3.3. Préparation des jeux de données et validation croisée 33
3.4. Élaboration des modèles sous-jacents 35
3.4.1. Méthode à deux modèles 36
3.4.2. Méthode Class Transformation 36

3.4.3. Méthode de modélisation directe 37
3.5. Élaboration des modèles de second niveau 38
3.5.1. Problématique de la variable-cible 38
3.5.2. Méthode avec agrégation par strates 39
3.5.3. Méthode par matching 40
3.5.4. Méthode de combinaison 41
3.6. Évaluation de la performance 42
3.6.1. Mesures sur données pratiques 42
3.6.2. Mesures sur données simulées 43
4. Résultats sur données simulées 44
4.1. Notes sur les résultats 44
4.2. Variation de la taille de l'effet Uplift – binaire 44
4.2.1. Performance 44
4.2.2. Poids accordés aux modèles 47
4.3. Variation de la taille de l'effet Uplift - continue 49
4.3.1. Performance 49
4.3.2. Poids accordés aux modèles 52
4.4. Simulation avec variation de la propension de traitement – binaire 54
4.4.1. Performance 54
4.4.2. Poids accordés aux modèles 59
4.5. Simulation avec variation de la propension de traitement – continue 60
4.5.1. Performance 60
4.5.2. Poids accordés aux modèles 64
4.6. Résumé des résultats 65
5. Présentation du cas 66
5.1. Préparation des données 66
5.2. Sélection de variables et élaboration des modèles 68
6. Résultats du cas pratique 72
6.1. Poids accordés par les modèles de stacking 72

6.2. Performance 73
7. Discussion 77
8. Conclusion 84
Bibliographie 88
Annexe 1: Tableaux de résultats sur données simulées 91
Simulation du package Uplift – binaire 91
Simulation du package Uplift – continue 95
Modèle avec variation de la proportion traitement/contrôle – binaire 98
Modèle avec variation de la proportion traitement/contrôle - continue 101
Annexe 2: Résultats complets du cas 105
Annexe 3: Tableaux de poids des modèles sous-jacents 109
1. Variation de la propension de traitement - continue 110
2. Variation de la propension de traitement - Binaire 111
3. Variation de la taille d'effet Uplift - Binaire 112
4. Variation de la taille d'effet Uplift - Continue 114

Liste des tableaux et figures

Figure 2.1. Une courbe des gains utilisée pour évaluer un modèle Uplift 25
Figure 2.2. Le concept de base du stacking consiste à utiliser les prédictions de modèles sous-jacents comme entrées dans un modèle de second niveau pour obtenir une seule prédiction qui est une combinaison linéaire des modèles sous-jacents. 27
Figure 3.1. La méthodologie de base représentée graphiquement: les prédictions des trois approches sont combinées pour faire une seule prédiction. 30
Figure 3.2. Structure d'un jeu de données pour la validation croisée 34
Figure 3.3. Chaque modèle est entraîné deux fois, l'une par validation croisée, l'autre sans, c'est-à-dire sur l'échantillon complet. Les poids obtenus par validation croisée sur le jeu d'entraînement sont ensuite appliqués sur les prédictions des modèles entraînés 36
Figure 4.1. Erreur quadratique moyenne en fonction de la taille de l'effet Uplift, variable-cible binaire. 45
Figure 4.2. Qini en fonction de la taille de l'effet Uplift, variable-cible binaire. 46
Figure 4.3. Lift sur le premier décile en fonction de la taille d'effet Uplift, variable-cible binaire. 47
Figure 4.4. Poids attribués aux modèles selon la méthode d'appariement (matching), variable-cible binaire. 48
Figure 4.5. Poids accordés aux modèles sous-jacents selon la méthode par strates, variable-cible binaire 49
Figure 4.6. Erreur quadratique moyenne en fonction de la taille d'effet uplift, variable-cible continue 51
Figure 4.7. Lift sur le premier décile en fonction de la taille d'effet Uplift, variable-cible continue. 52
Figure 4.8. Poids attribués aux modèles sous-jacents selon la méthode d'appariement, varaible-cible continue. 53
Figure 4.9. Poids attribués aux modèles sous-jacents par la méthode strates, variable-cible continue. 54
Tableau 4.1. Erreurs quadratiques moyennes en fonction de la propension de traitement, variable-cible binaire. 55

Figure 4.10. Erreur quadratique moyenne en fonction de la propension de traitement, variable-cible binaire. 56
Figure 4.11. Coefficient Qini en fonction de la proportion de traitement, variable-cible binaire. 57
Figure 4.12. Lifts sur le premier décile en fonction de la propension de traitement, variable-cible binaire. 58
Figure 4.13. Poids attribués aux modèles sous-jacents en fonction de la propension de traitement selon la méthode d'appariement, variable-cible binaire. 59
Figure 4.14. Poids attribués aux modèles sous-jacents en fonction de la propension de traitement selon la méthode par strates, variable-cible binaire. 60

Tableau 4.2. Erreur quadratique moyenne en fonction de la propension de traitement, variable-cible continue..... 61
Figure 4.15. Erreur quadratique moyenne en fonction de la propension de traitement, variable-cible continue. 62
Figure 4.16. Lifts sur le premier décile en fonction de la propension de traitement, variable-cible continue..... 63
Figure 4.17. Poids des modèles en fonction de la propension de traitement selon la méthode par appariement, variable-cible continue 64
Figure 4.18. Poids des modèles en fonction de la propension de traitement selon la méthode par strates, variable-cible continue. 65
Tableau 5.1. Décomposition du jeu de données selon l'assignation de traitement et la réponse 68
Tableau 5.2. Répartition des variables en fonction de leur nombre d'apparitions dans des modèles différents 69
Tableau 6.1. Poids des modèles sous-jacents selon les deux méthodes de stacking dans le contexte de l'étude de cas 73
Figure 6.1. Gains cumulatifs en fonction du décile dans le contexte de l'étude de cas. 74
Figure 6.2. Gains marginaux par décile dans le contexte de l'étude de cas. 75
Tableau 6.2. Principales mesures de performance pour l'étude de cas 75

Liste des abréviations

EQM : Erreur quadratique moyenne

DGP : Data Generating Process, ou processus de génération de données

Remerciements

J’aimerais en premier lieu remercier mon directeur de mémoire, Denis Larocque, qui a cru en moi avant même mon inscription à la maîtrise et sans qui ce mémoire n’existerait tout simplement pas. Tout au long du processus, il a eu le don de m’accompagner là où c’était nécessaire et de transformer des problèmes qui semblaient à première vue insurmontables en de simples blocages temporaires. Ses conseils judicieux, au bon moment, ont fait en sorte de me permettre une progression constante à travers les hauts et les bas qui viennent inévitablement dans le cadre de la rédaction d’un travail de cette ampleur.

De même, il me serait impossible de taire la contribution de Tarek, qui m’a fourni conseils et accompagnement afin de réconcilier la recherche académique avec les aspects plus pratiques du projet. Son aide précieuse m’a permis d’éviter de nombreux pièges potentiels sur la route de la réalisation de ce mémoire, et de l’enrichir avec plusieurs éléments auxquels je n’aurais pas pensé sans son appui et son soutien constant.

Je dois également souligner ma reconnaissance envers Michel, qui a démontré un enthousiasme pour m’aider à réaliser ce projet dès que je l’ai contacté pour la première fois, et qui a travaillé pendant plusieurs mois pour me permettre de le réaliser dans sa forme actuelle.

En terminant, de simples mots ne suffisent pas à exprimer l’importance de mes parents dans ma vie et, par extension, dans la réalisation de ce projet. Leur soutien, tantôt moral, tantôt financier, psychologique en d’autres occasions...bref, leur soutien n’a aucune limite connue. Leur travail a permis de faire de moi la personne que je suis aujourd’hui, et c’est grâce à eux que je suis capable d’écrire ces lignes.

1. Introduction

L'optimisation du ciblage de la clientèle est l'un des problèmes-clés en marketing que tente de résoudre la science des données. La méthode la plus répandue actuellement est l'utilisation de modèles de propension, qui évaluent la probabilité naturelle qu'un client achète un produit au cours d'une période donnée, et ciblent la clientèle qui achètera vraisemblablement le produit avec des offres publicitaires. Cette méthode a sa raison d'être dans un contexte de marché compétitif où le risque existe qu'un client qui achètera le produit, acquière celui-ci auprès d'un compétiteur; cependant, elle a également le défaut de ne pas tenir compte de l'effet de l'offre, en plus de ne cibler que les clients qui achèteraient le produit coûte que coûte.

Une méthode plus sophistiquée consiste à enregistrer les réponses d'une campagne publicitaire réelle et utiliser celles-ci afin de prédire la probabilité qu'un client achète le produit, en prenant pour acquis qu'il est ciblé par ladite campagne publicitaire. Cette méthode est plus efficace que le modèle de propension, car elle cible les clients qui ont répondu favorablement à l'offre publicitaire, maximisant le taux de réponses positives de la campagne. Cependant, cette méthode cible toujours des clients qui sont des *sure things,* en plus des clients dits persuadables, qui répondraient favorablement à l'offre mais qui ne prendraient vraisemblablement pas le produit si on avait omis de lui faire une offre directe.

C'est cette dernière catégorie qui est le centre d'intérêt de la modélisation incrémentale, également connue sous l'appellation anglaise Uplift (par soucis de brièveté, ce dernier terme sera privilégié pour le reste de ce mémoire). En utilisant un groupe traitement et un groupe contrôle, la modélisation Uplift tente de maximiser l'impact de l'offre sur la probabilité que chaque client réponde favorablement. Les clients les plus « persuadables » sont retenus par le modèle, alors que ceux qui sont certains de prendre ou de ne pas prendre le produit, de même que ceux sur qui l'offre publicitaire a un impact négatif, sont écartés de la campagne.

La problématique centrale de la modélisation Uplift provient du fait que la valeur réelle de cet impact est totalement inconnue, même dans le cas du jeu de données d'entraînement, parce qu'un même client ne peut se retrouver à la fois dans les groupes de traitement et de contrôle. La littérature existante a détaillé trois grandes familles d'approches qui

solutionnent à leur manière cette problématique. Chacune de ces solutions a ses avantages et ses inconvénients et aucune ne solutionne de manière définitive le problème.

Ce mémoire s'inscrit dans cette quête pour trouver une solution fiable au problème Uplift. En effet, la présente recherche repose sur l'hypothèse qu'une approche combinatoire des trois méthodes existantes par stacking permettra d'améliorer la performance et la robustesse de modèles Uplift. Bien que d'autres méthodes d'ensemble aient été tentées par le passé, c'est la première fois que cette approche par stacking est tentée dans la littérature Uplift. Plusieurs manières de produire un estimateur du lift ont également été étudiées afin de servir de cible au modèle de second niveau, apportant une contribution secondaire à la recherche d'un estimateur fiable du lift réel.

Le présent mémoire est divisé en six chapitres. Le premier présentera une revue de la littérature existante dans les domaines de la modélisation Uplift et de la méthode par stacking. Le second chapitre détaillera les différents choix méthodologiques qui ont été faits dans le processus de réalisation de cette recherche, de même qu'une explication des raisons derrière chaque choix lorsque applicable. Le troisième chapitre présentera les résultats obtenus par les méthodes élaborées et comparera celles-ci avec chacune des trois méthodes classiques en modélisation Uplift. Le quatrième chapitre présentera un cas d'application pratique élaboré sur un jeu de données de production en collaboration avec une entreprise canadienne de même que les choix et modifications au modèle qui ont été nécessaires pour implanter la méthode sur ces données. Le cinquième chapitre évalue et compare les résultats sur ce contexte de données réelles, validant l'applicabilité de la solution obtenue. Le sixième et dernier chapitre consiste en une évaluation globale de l'étude et de ses limites et en une discussion de ses impacts possibles et de l'évolution à laquelle pourrait mener le concept dans de futures recherches.

2. Revue de littérature

2.1. Contexte

Au sein de la large famille de méthodes d'apprentissage automatique, la modélisation Uplift demeure une niche relativement restreinte, puisqu'elle ne s'applique qu'à un type de problématique bien spécifique qui implique l'évaluation et la maximisation de l'impact d'un traitement donné.

Bien que son application puisse paraître restreinte à première vue, le perfectionnement des techniques de modélisation Uplift a en réalité un impact significatif sur les champs d'activité sur lesquels la méthode peut être utilisée.

La politique en est l'un des exemples les plus marquants à ce jour. En effet, la campagne de réélection de Barack Obama à la présidentielle américaine de 2012 est notoire pour avoir utilisé des modèles Uplift pour identifier les indécis, et adapter les actions de campagne en fonction des ciblages générés par ces modèles (Siegel, 2013).

Un autre usage traditionnel de la modélisation Uplift se situe dans les domaines médical et pharmaceutique (Jaskowski & Jaroszwicz, 2012), où elle est connue sous le nom de modèle causal. L'usage de groupes traitement et contrôle est déjà largement répandu dans ce domaine par sa nature même où l'on tente de démontrer l'efficacité d'un traitement. La modélisation Uplift permet toutefois de distinguer les effets qui sont généralisés de ceux qui sont individuels, permettant de déceler si des facteurs connus influenceront l'efficacité du traitement.

Le marketing est un autre domaine où l'utilisation de modèles Uplift a énormément de potentiel, alors que celle-ci donne un outil permettant aux entreprises de maximiser l'impact de leurs campagnes publicitaires, tout en réduisant les coûts en ne ciblant que les clients sur qui la campagne aura le plus grand effet.

Les enjeux d'une amélioration de la performance de modèles Uplift sont donc significatifs, ceux-ci étant en emploi dans une variété de domaines qui tirent chacun un bénéfice lorsqu'ils parviennent à identifier de manière plus fiable l'effet individuel d'un même traitement.

2.2. Modélisation Uplift

La modélisation Uplift, une adaptation spécifique au contexte d'affaires du problème d'inférence causale d'un traitement, a été proposée par Victor S.Y. Lo (2002). Celui-ci avait alors identifié une faiblesse dans la méthode traditionnellement utilisée pour prédire l'effet d'un traitement sur la variable cible dans un contexte d'affaires.

En effet, la méthode traditionnelle consiste à maximiser E(Y), Y étant la variable-cible. Elle identifie donc les individus qui ont l'espérance la plus élevée, sans égard à l'efficacité du traitement en soi. Par exemple, si Y est la probabilité qu'un client donné achète un produit, un tel algorithme tentera d'identifier les clients qui ont la plus forte probabilité d'acheter le produit, tout simplement.

C'est précisément à cette approche que Lo s'objecte, puisqu'elle n'évalue pas l'effet du traitement lui-même. Du moment que la probabilité d'achat est élevée, l'effet du traitement pourrait être nul voire négatif et un tel algorithme ciblerait tout de même le client. Cette méthodologie ne cible pas de manière optimale les individus pour lesquels le traitement aura le plus d'impact.

La méthodologie Uplift tente de solutionner ce problème en tenant compte des effets hétérogènes, c'est-à-dire en évaluant l'effet du traitement en fonction des variables explicatives pour chaque individu. Avec suffisamment de données dans les groupes de traitement et de contrôle, il devient possible d'évaluer pour chaque individu la différence entre P(Y=1|contrôle) et P(Y=1|traitement). La différence entre ces deux quantités correspond à l'effet du traitement pour cet individu, qu'on appelle *uplift* ou plus simplement *lift.*

L'objectif des modèles Uplift est donc de découvrir le plus fidèlement possible la valeur du lift pour chaque observation, pour ensuite sélectionner seulement ceux pour lesquels le lift est le plus élevé. En tenant compte des variables explicatives, l'effet du traitement devient donc (Larocque, 2018) :

$$\tau(x) = E\big(Y \,|\, W = 1, X = x\big) - E\big(Y \,|\, W = 0, X = x\big)$$

où W est l'indicateur binaire de traitement et X est un vecteur de variables explicatives.

Cela demeure vrai tant dans le cas d'une variable Y binaire que d'une variable continue; en effet, dans le cas d'une variable binaire, E(Y) correspondra à la probabilité que Y prenne la valeur 1.

Le problème qui se présente inévitablement dans la poursuite de cet objectif, est que le lift réel n'est pas directement observable, puisqu'un même individu ne peut pas à la fois appartenir aux groupes traitement et contrôle, ce qui rend plus difficile la modélisation et l'évaluation de la performance (Larocque, 2018).

Trois approches ont été proposées dans la littérature pour apporter une piste de solution à ce problème. Les trois sections suivantes seront consacrées à l'examen de ces trois approches.

2.2.1. Approche à deux modèles

L'approche la plus simple, et également la plus souvent proposée dans la littérature (Radcliffe, 2007) consiste à construire deux modèles différents, l'un utilisant seulement les observations du groupe traitement et l'autre utilisant celles du groupe contrôle. Ces deux modèles peuvent ensuite être utilisés pour estimer le lift. En théorie, la différence entre le modèle traitement et le modèle contrôle devrait représenter le lift, puisque la seule différence à l'entraînement des deux modèles est la valeur que prend la variable traitement.

Au-delà de sa simplicité, l'avantage de cette idée est qu'elle permet d'utiliser une variété de modèles prédictifs traditionnels sans nécessiter de modifications ou d'adaptations spécifiques à la problématique uplift. En effet, les principaux modèles d'apprentissage machine, tels que la régression, les arbres, les réseaux de neurones, et les forêts aléatoires, peuvent tous être implantés tels quels pour prédire les

valeurs de Y (Larocque, 2018). Cela permet également d'utiliser les algorithmes les plus récents, ce qui n'est pas le cas lorsque l'on modélise l'uplift directement.

En pratique, parce que les modèles sont imparfaits et entraînés sur des jeux de données indépendants, cette approche ajoute un niveau d'incertitude puisque la différence entre les deux populations n'est pas directement modélisée. Victor S.Y. Lo (2002) s'est inquiété de cette problématique, indiquant que les estimateurs du lift peuvent être sensibles à d'infimes variations dans les paramètres des modèles, qui ne sont pas prises en compte dans cette approche. C'est d'autant plus vrai lorsque l'effet uplift moyen est très petit, et donc plus difficile à saisir (Surry & Radcliffe, 2011).

Lo suggère plutôt de modifier cette approche pour n'entraîner qu'un seul modèle en utilisant une régression, ajoutant des termes d'interaction entre la variable traitement et les autres variables explicatives. Le fait que cette approche ne produise qu'un seul modèle permet de mitiger les craintes liées aux estimations indépendantes de l'uplift.

De manière générale, il a été démontré que l'approche à deux modèles est relativement performante, toutefois d'autres approches réussissent généralement à la surclasser (Gutierrez & Gerardy, Causal Inference and Uplift Modelling - A review of the literature, 2016). Sa performance est habituellement supérieure dans des contextes de données simulées, mais moins intéressante lorsque des données réelles sont utilisées (Surry & Radcliffe, 2011).

L'une des raisons avancées par ces mêmes auteurs pour expliquer ce phénomène, et fréquemment observées dans le contexte d'expérimentations, est que les variables qui influencent les résultats et celles qui influencent l'effet du traitement sont parfois différentes. Cela signifie que les modèles qui entrent dans la composition de cette approche, qui tentent de reproduire les résultats plutôt que le lift, capturent les mauvais éléments. Pour cette même raison, l'approche à deux modèles semble mieux performer lorsque la corrélation est élevée entre Y et le lift; les mêmes

éléments qui influencent la variable Y, influencent alors eux aussi le lift, facilitant la capture de l'effet par le modèle Uplift combiné (Soltys, Jaroszewicz, & Rzepakowski, 2014).

2.2.2. Approche *Class Transformation*

Une seconde approche, suggérée par Maciej Jaskowski et Szymon Jaroszewicz (2012), consiste à définir une variable-cible Y modifiée qui servira ensuite à entraîner le modèle.

Deux variantes de cette méthode ont été proposées. La première assume l'indépendance de l'assignation du traitement par rapport aux variables sous-jacentes, c'est-à-dire que toutes les observations ont la même probabilité d'être dans le groupe traitement, peu importe les valeurs des autres variables, et que la proportion traitement/contrôle est la même à 50%. Dans ce scénario, la nouvelle variable, Z, est définie de la manière suivante :

$$Si\ Y_i = 1\ et\ W_i = 1, alors\ Z_i = 1.$$
$$Si\ Y_i = 0\ et\ W_i = 0, alors\ Z_i = 1.$$
$$Si\ Y_i = 1\ et\ W_i = 0, alors\ Z_i = 0.$$
$$Si\ Y_i = 0\ et\ W_i = 1, alors\ Z_i = 0.$$

Ensuite, il est possible d'entraîner un modèle prédictif sur cette variable afin d'obtenir $P(Z_i=1 \mid X=x)$. Ces prédictions seront utilisées pour obtenir un estimateur de l'uplift de la manière suivante :

$$\tau(x) = 2P(Z_i = 1|\mathrm{X} = \mathrm{x}) - 1.$$

Cette méthode a été généralisée par Susan Athey et Guido W. Imbens (2015) afin de la rendre applicable même lorsque la probabilité qu'un sujet soit traité dépend directement des valeurs des variables explicatives.

La méthode suggérée consiste d'abord à entraîner un premier modèle qui permettra de prédire le score de propensité, noté p_w, qui reflète la probabilité qu'une observation

spécifique appartienne au groupe de traitement à partir des valeurs des variables sous-jacentes. La variable transformée dans ce cas-ci sera alors :

$$Y_i^* = Y_i * \left(\frac{W_i}{\hat{p}_w(x_i)} - \frac{1 - W_i}{1 - \hat{p}_w(x_i)}\right)$$

où Y_i est correspond à la variable-cible, W_i est l'indicateur de l'assignation de traitement, et x_i est le vecteur des variables explicatives pour le sujet i. $\hat{p}_w$ est une estimation de la fonction de propensité obtenue en parallèle. L'espérance de Y_i^* en fonction des variables explicatives est alors égale à l'effet uplift :

$$E[Y_i^* | X_i = x] = \tau(x).$$

Un second modèle prédictif viendra donc estimer Y_i^* en fonction des variables explicatives X_i pour obtenir l'effet du traitement pour chaque observation selon la méthode Class Transformation.

Tout comme la méthode à deux modèles, une fois les étapes préalables de transformation de la variable-cible complétées, cette méthode permet l'utilisation directe de modèles prédictifs traditionnels sans adaptation, permettant l'utilisation d'algorithmes de pointe.

2.2.3. Approche de modélisation directe du lift

La troisième approche est également celle sur laquelle la recherche la plus récente est concentrée. Elle consiste à construire de nouveaux algorithmes spécialement conçus dans le but de modéliser directement le lift.

De tels algorithmes ont été développés dans la littérature existante en modifiant des modèles de forêts aléatoires, de machines à support de vecteur (SVM), de K plus proches voisins et de boosting avec l'algorithme AdaBoost; (Zhao, Fang, & Simchi-Levi, 2017). Un algorithme nommé Contextual Treatment Selection (CTS) est également décrit par Zhao, Fang et Simchi-Levi. Celui-ci s'apparente à des forêts aléatoires en ce sens qu'il utilise des ensembles d'arbres décisionnels pour modéliser la relation tout en limitant le surapprentissage.

2.2.4. Évaluation de la performance

L'évaluation de la performance est un enjeu majeur de la recherche dans la modélisation Uplift. En effet, la valeur réelle du lift étant inconnue même sur les jeux de données utilisés pour entraîner les modèles, les mesures de performance traditionnelles ne sont pas utilisables dans le cadre de la méthodologie Uplift, sauf dans le cas spécifique de données simulées. Dans un contexte réel, il faut se rabattre sur des mesures alternatives.

Une méthode suggérée par Gerardy et Gutierrez (2016) consiste à employer la variable Y_i^* calculée dans le cadre de la méthode Class Transformation comme estimateur de l'effet de traitement. En effet, ceux-ci ont démontré qu'à la limite, l'espérance de Y_i^* est égale à l'espérance de l'effet moyen du traitement $\tau(x)$. On pourrait donc substituer Y_i^* pour la valeur réelle attendue dans le calcul de l'erreur quadratique moyenne.

En pratique, après avoir testé cette méthode dans le cadre de l'exploration préliminaire de ce projet, cette méthode n'a pas fonctionné. Une explication possible pour cela est que si en moyenne, ces deux valeurs sont égales, l'essai de cette méthode sur des données simulées où le lift réel est connu révèle que des écarts substantiels existent au niveau individuel entre les deux valeurs. Sur une base individuelle, la variable transformée n'agit donc pas comme l'estimateur attendu de l'effet de traitement. Toutefois, il n'est pas impossible que d'autres recherches démontrent qu'il est en fait possible d'utiliser la variable transformée comme cible avec une approche différente.

La courbe des gains et le coefficient Qini qui y est associé sont les deux mesures traditionnellement associées à l'évaluation de la performance de modèles Uplift.

On obtient la courbe des gains en appliquant pour chaque proportion de la population totale que l'on souhaite évaluer (traditionnellement par décile, mais la courbe peut être aussi précise qu'on le souhaite) la formule suivante, telle que rapportée par Pierre Gutierrez (2016), sur les données préalablement triées en fonction du score de lift attribué par le modèle :

$$Q = Y_{W=1}(\mathrm{k}) - Y_{W=0}(k) * N_{W=0}(k)/N_{W=1}(k)$$

où :

- $Y_{W=1}(k)$ est le nombre d'observations où Y=1 dans le groupe traitement parmi les k premières observations
- $Y_{W=0}(k)$ est le nombre d'observations où Y=1 dans le groupe contrôle parmi les k premières observations
- $N_{W=1}(k)$ est le nombre total d'observations du groupe traitement parmi les k premières observations
- $N_{W=0}(k)$ est le nombre total d'observations du groupe traitement parmi les k premières observations.

En évaluant cette fonction pour plusieurs valeurs différentes de k, on obtient la courbe des gains, qui s'évalue sensiblement de la même manière que la méthode traditionnelle de l'aire sous la courbe ROC: plus la courbe est élevée au-dessus de la droite représentant une sélection aléatoire, meilleur est le modèle.

Le coefficient Qini est pour sa part défini comme étant l'aire entre la courbe des gains du modèle et la droite représentant une sélection aléatoire (Gutierrez & Gerardy, 2016).

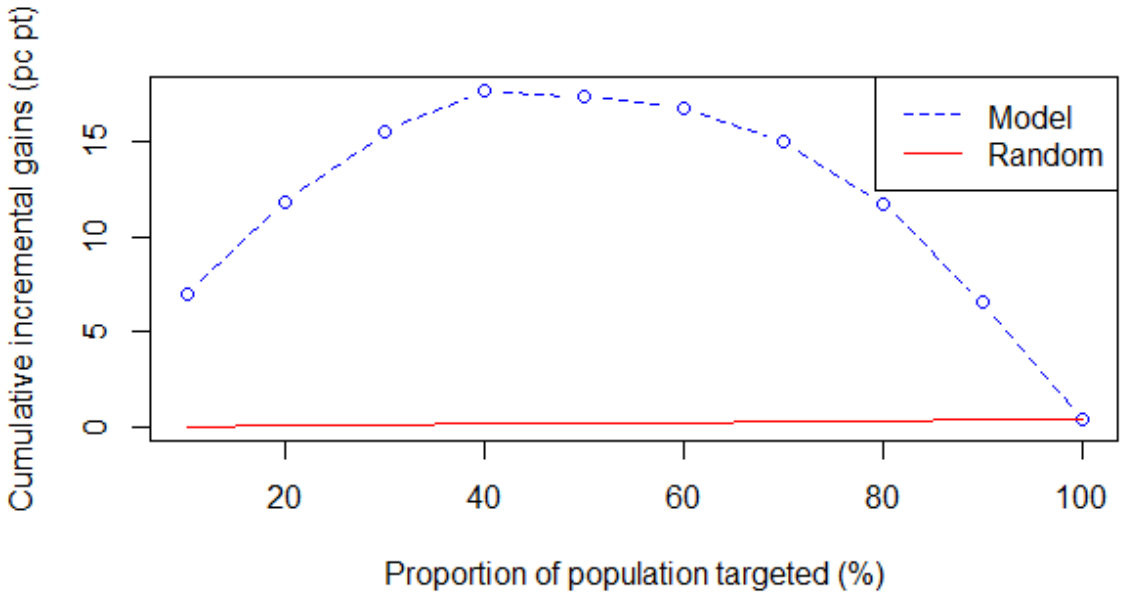

FIGURE 2.1. UNE COURBE DES GAINS UTILISÉE POUR ÉVALUER UN MODÈLE UPLIFT

En termes d'interprétation, la courbe des gains indique la proportion supplémentaire d'individus pour qui Y=1 identifiés par le modèle (ligne bleue dans la Figure 2.1) versus une sélection aléatoire (représentée par la ligne rouge, Random, dans la Figure 2.1), pour toutes

les proportions de ciblage de la population possibles. Plus la différence entre la courbe et la ligne représentant la sélection aléatoire est élevée, meilleure est la qualité du modèle. Cette méthode d'évaluation ressemble fortement à la courbe ROC utilisée dans d'autres types de problèmes d'apprentissage automatique.

Une autre approche couramment utilisée selon Gerardy et Gutierrez (2016), consiste à prédire l'uplift sur l'ensemble des observations qu'elles appartiennent au groupe traitement ou contrôle, puis trier les données par lift prédit pour ensuite faire une moyenne par décile dans les deux groupes. Cela donne une idée du gain d'uplift auquel on peut s'attendre pour chaque décile.

Le Qini et la courbe des gains demeurent utilisés dans le domaine de la modélisation Uplift seulement parce qu'aucune mesure plus fiable n'existe (Zhao, Fang, & Simchi-Levi, 2017) (Rzepakowski & Jaroszewicz, 2010). Le Qini conserve certains défauts, dont notamment le fait de ne mesurer qu'indirectement le lift. En effet, le Qini est une mesure d'ordre; un modèle pourrait être peu utile pour prédire la valeur réelle du lift, mais du moment qu'il ordonne correctement les observations, le coefficient Qini démontrera une performance élevée du modèle (Mesalles Naranjo, 2012). L'intérêt dans la recherche est élevé pour une mesure alternative au Qini.

D'un point de vue d'élaboration de nouvelles méthodologies pour implanter des modèles Uplift, l'utilisation de données simulées est fortement répandue puisqu'elle permet alors de connaître le lift réel et d'utiliser celui-ci conjointement avec des mesures de performance plus traditionnelles.

L'erreur quadratique moyenne entre les prédictions du modèle et la valeur réelle, utilisée conjointement avec le Qini pour découvrir si les deux mesures abondent dans le même sens, est l'une des méthodes les plus fréquemment observées dans la littérature.

Oscar Mesalles-Naranjo (2012) et Nicholas J. Radcliffe (2012) ont proposé une mesure alternative de performance appelée Moments d'Uplift qui n'est pas basée sur le rang et qui, selon les auteurs, est plus précise que les autres mesures de performance Uplift existantes. Cependant, le calcul de cette mesure de performance implique l'accès aux valeurs réelles du lift, ce qui élimine toute utilisation de cette mesure dans un contexte pratique, limitant celui-ci à l'élaboration de modèles sur des données simulées.

Dans le cas d'un modèle Uplift utilisé en production, la méthode la plus efficace pour faire le suivi de la performance consiste à utiliser un groupe contrôle sélectionné aléatoirement; la différence entre la proportion de Y=1 dans le groupe ciblé à l'aide du modèle et dans le groupe contrôle correspond à l'incrément apporté par le modèle prédictif utilisé.

2.3. Stacking

2.3.1. Introduction

Le *stacking*, également connu sous les noms *stacked generalization* et *stacked regression*, est une technique proposée initialement par David H. Wolpert (1992) qui consiste à utiliser un modèle de second niveau, auquel on donne pour variables explicatives les prédictions individuelles produites par plusieurs autres modèles prédictifs. Ce modèle de second niveau détermine ensuite les poids qui doivent être appliqués aux prédictions de chaque modèle de premier niveau pour minimiser l'erreur des prédictions par rapport à la valeur réelle attendue (Figure 2.2). Les tests effectués par Wolpert tendent à démontrer que, dans l'ensemble, le stacking permet de réduire l'erreur de prédiction globale du modèle.

FIGURE 2.2. LE CONCEPT DE BASE DU STACKING CONSISTE À UTILISER LES PRÉDICTIONS DE MODÈLES SOUS-JACENTS COMME ENTRÉES DANS UN MODÈLE DE SECOND NIVEAU POUR OBTENIR UNE SEULE PRÉDICTION QUI EST UNE COMBINAISON LINÉAIRE DES MODÈLES SOUS-JACENTS.

Wolpert a par ailleurs suggéré que des modifications mineures au concept de stacking permettent de l'utiliser dans des buts différents, par exemple comme une forme de validation croisée, en entraînant des modèles identiques sur des sous-ensembles de données différents puis en combinant les modèles par stacking. Cependant, cet usage ne s'est pas répandu dans la littérature; les auteurs subséquents ayant contribué à la recherche sur le stacking ont préféré explorer son usage en tant que méthode d'ensemble pour minimiser l'erreur de prédiction (Breiman, 1996); (Ting & Witten, 1997); (van der Laan, Polley, & Hubbard, 2007).

Le stacking peut être utilisé aussi bien dans une optique de régression (Breiman, 1996) que de classification (Leblanc & Tibshirani, 1994), et tout type de modèle prédictif peut être utilisé comme modèle de premier niveau. Pour le modèle de second niveau, les possibilités sont également très larges, mais la recherche jusqu'à maintenant s'est largement concentrée sur des méthodes de régression (linéaire, logistique, ridge) ou encore d'arbres (arbres de régression, forêts aléatoires) (van der Laan, Polley, & Hubbard, 2007).

Günes (2017) suggère qu'il est préférable d'utiliser des modèles très différents les uns des autres pour le stacking, puisque les modèles différents captureront des éléments différents du jeu de données sous-jacents, ce qui améliorera la performance du modèle combiné. Elle propose quatre méthodes pour créer des modèles sous-jacents différents: varier les algorithmes d'entraînement, modifier les hyperparamètres d'un même algorithme, employer des variables différentes en entrée, de même qu'utiliser des jeux d'entraînement différents, lorsque la quantité de données est suffisante pour le permettre.

Des études subséquentes par van der Laan, Polley et Hubbard (2007) ont démontré que les algorithmes de méta-apprentissage représentent une manière « asymptotiquement optimale » d'élaborer un modèle prédictif, c'est-à-dire que ces modèles ne feront jamais moins bien que le meilleur modèle sous-jacent. Une explication qu'ils ont apportée est qu'en l'absence d'une combinaison linéaire capable de réduire l'erreur par rapport au meilleur modèle sous-jacent, un tel modèle placerait tout le poids sur ledit algorithme et ignorerait les autres. Du moment qu'une combinaison capable de bâtir ces modèles existe, l'algorithme les sélectionnera et pourra ainsi améliorer sur la performance de tous les modèles pris individuellement. Concrètement, donc, le stacking ne devrait jamais faire pire que le meilleur modèle sous-jacent.

2.3.2. Considérations méthodologiques

L'une des problématiques fondamentales lorsqu'il est question de méthodes d'ensemble, et plus particulièrement de stacking, est le surapprentissage. Lorsque plusieurs modèles sont combinés et que la complexité augmente, le risque d'une mauvaise généralisation augmente significativement. Pour cette raison, Breiman (1996) insiste sur la nécessité d'utiliser la validation croisée lors de la détermination des poids pour le modèle de second niveau.

Une autre raison pour cette nécessité est que si le modèle de second niveau est entraîné sur les mêmes données que les modèles sous-jacents sans utiliser de validation croisée, le modèle aura systématiquement un biais pour le modèle sous-jacent le plus performant (Himmetoglu, 2017).

Breiman suggère donc d'entraîner ces modèles en utilisant la validation croisée *leave-one-out* et d'utiliser ensuite ces prédictions pour estimer les poids des modèles. Dans un monde idéal, c'est systématiquement cette méthode qui serait utilisée; toutefois, étant donnés les besoins de calcul élevés d'une telle méthodologie, la validation croisée K-groupes peut également être utilisée.

Un autre problème soulevé par Breiman est la corrélation élevée entre les variables insérées comme entrées dans l'algorithme de méta-apprentissage. La corrélation élevée rend les modèles très sensibles à de petites variations dans les données, réduisant l'efficacité du modèle résultant. La réponse traditionnelle pour solutionner ce type de problème où les variables sont fortement corrélées est d'utiliser une régression ridge. Selon les études de Breiman, les résultats de cette méthode ne sont cependant pas totalement fiables.

Il suggère plutôt de remplacer la contrainte d'une régression ridge par une contrainte de non-négativité sur les poids attribués à chaque modèle, indiquant qu'en pratique, cette combinaison a toujours une erreur de prédiction plus faible que les modèles seuls; cependant, il note que cette assertion n'est pas prouvée. Witten et Ting (1997) indiquent pour leur part que la contrainte de non-négativité jugée cruciale par Breiman n'est d'aucune utilité dans un contexte de classification.

3. Méthodologie

3.1. Approche générale

Le concept de base qui sera exploré consiste à employer un modèle de stacking pour trouver une combinaison linéaire optimale des trois grandes approches de modélisation Uplift. Ainsi, le modèle de second niveau prend en entrée les prédictions des trois approches, soit la méthode deux modèles, Class Transformation et la modélisation directe, et attribue un poids à chacune d'entre elles, de manière à produire une prédiction unique combinant les prédictions de ces modèles sous-jacents (Figure 3.1).

FIGURE 3.1. LA MÉTHODOLOGIE DE BASE REPRÉSENTÉE GRAPHIQUEMENT: LES PRÉDICTIONS DES TROIS APPROCHES SONT COMBINÉES POUR FAIRE UNE SEULE PRÉDICTION.

3.2. Simulation de données

Les jeux de données compatibles avec la méthodologie Uplift se font excessivement rares (Soltys, Jaroszewicz, & Rzepakowski, 2014), et les jeux de données qui sont disponibles publiquement sont généralement peu adaptés à l'expérimentation de nouvelles méthodologies comme ce que tente de réaliser le présent mémoire.

En effet, ce type d'expérimentation requiert deux caractéristiques qui sont absentes de tels jeux de données: la taille du signal Uplift doit être contrôlée de manière à pouvoir tester les méthodes sur un large éventail de forces de signal possibles. De plus, le lift réel doit être connu de manière à pouvoir évaluer la performance du modèle de manière préliminaire avant de l'évaluer ensuite sur un jeu de données réel. Cette approche permet de valider que le Qini, utilisé sur les données réelles, est représentatif de la performance réelle du modèle dans un cadre contrôlé.

De même, l'hypothèse d'indépendance conditionnelle n'est pas systématiquement respectée dans des jeux de données utilisés en pratique.

Pour toutes ces raisons, l'usage de données simulées s'est imposé dans le contexte de ce projet. Plusieurs algorithmes ont été utilisés. Ils seront présentés dans les sections suivantes.

3.2.1. Simulation du package Uplift (sim_pte)

Pour les besoins de la présente étude, la fonction *sim_pte* contenue dans le package R *uplift* (Guelman, 2014) a été utilisée puisqu'elle produit aisément des jeux de données qui répondaient aux critères énoncés ci-dessus, en plus de permettre de contrôler le nombre d'observations, de modifier certains paramètres de la simulation, de permettre de produire de nouveaux jeux de données simulées de manière simple et rapide et de reproduire les mêmes jeux de données et les mêmes modèles si cela s'avérait nécessaire.

Elle produit des données dans le format (Y,W,X) nécessaire à l'entraînement et l'évaluation de modèles Uplift, Y étant une variable-cible binaire, W étant l'indicateur binaire traitement/contrôle et X étant une série de variables explicatives standardisées. Elle génère des données qui respectent l'hypothèse d'indépendance conditionnelle: l'assignation de traitement est indépendante des variables explicatives et les jeux de données sont complètement équilibrés entre groupes traitement et contrôle. Cette méthode est donc une excellente manière de vérifier, dans un premier temps, si la méthode fonctionne bien dans un monde parfait.

Avec cette première méthode, deux paramètres pouvaient être étudiés afin de constater leur effet sur la performance des différents modèles. D'abord, la taille de l'effet uplift comparativement aux effets d'interaction des variables explicatives est modifiée, permettant d'évaluer le fonctionnement de la méthode à diverses tailles d'effet uplift. Il était également possible de faire varier la corrélation entre les différentes variables explicatives; toutefois, dans le contexte où tous les modèles testés sont basés sur des forêts aléatoires dont l'algorithme performe naturellement une sélection de variables, ce paramètre était moins pertinent pour cette étude.

Les données sont générées de la manière suivante (Guelman, 2014):

$$Y = I(\sum_{j=1}^{p} \beta_j X_j + \sum_{j=1}^{p} \gamma_j X_j T + \sigma_0 \epsilon > 0)$$

où :

$$\gamma = \left(\frac{1}{2}, -\frac{1}{2}, \frac{1}{2}, -\frac{1}{2}, 0, \dots 0\right)$$

$$\beta = \frac{(-1)^{j+1} I(3 \leq j \leq 10)}{beta.den}.$$

Les variables explicatives X_j sont générées à l'aide d'une distribution normale multivariée avec une matrice de variance-covariance de type équicorrélation, et le terme d'erreur est généré selon une distribution normale de moyenne 0 et d'écart-type 1.

La fonction permet de faire varier la covariance entre les prédicteurs, la taille de l'effet Uplift en modifiant le paramètre beta.den, le terme d'erreur et le nombre de prédicteurs. X_j étant une matrice, sa taille variera en fonction du nombre de prédicteurs voulus.

Avant de prédire l'effet Uplift sur les données générées, il faut ramener l'indicateur de traitement à un format binaire {0,1} en changeant tous les cas où W=-1 pour W=0. Le jeu de données initial attribuait la valeur W=-1 aux observations de contrôle par défaut parce que certains algorithmes exigent cette nomenclature (Athey & Imbens, 2015). Ce n'est cependant pas le cas des algorithmes utilisés dans le contexte de ce mémoire, les données ont donc été ramenées à ce format binaire.

Une explication plus détaillée de la méthode de génération des données existe dans la documentation du package uplift (Guelman, 2014) et la méthodologie détaillée a été décrite par ses auteurs (Tian, Alizadeh, Gentiles, & Tibshirani, 2014).

Afin d'assurer que les résultats sont fiables et reproductibles plutôt que de ne représenter que les idiosyncrasies d'un seul jeu de données, la simulation a été tournée 100 fois avec les mêmes paramètres dans chaque cas. Chaque résultat indiqué dans les tableaux représente la performance moyenne des 100 modèles élaborés avec ces mêmes paramètres.

Une version modifiée de cette même simulation, produite par Denis Larocque et utilisant une loi normale plutôt qu'une loi binomiale pour générer la variable-cible, a également été utilisée pour tester la méthode avec une variable-cible continue.

3.2.2. Simulation avec variation de la probabilité de traitement

Un second processus de génération de données a été utilisé, cette fois pour évaluer l'impact d'une variation de la proportion d'observations de traitement et contrôle, et également de manière à tester la méthode sur des jeux de données générés de manière différente.

La fonction, préparée par Denis Larocque, génère une série de dix variables explicatives indépendantes et uniformément distribuées sur l'intervalle [0,1], dont seulement cinq sont réellement utilisées pour générer la variable-cible. Les cinq autres n'ont aucun lien et sont utilisées pour générer du bruit et confondre les algorithmes qui tentent d'expliquer le jeu de données.

En utilisant ces variables explicatives, on génère la quantité:

$$\frac{1}{1+e^{-0.3(-4+x1+x2+x3+x4+x5+0.5*treat+3(-1.5*treat*x1+treat*x2+treat*x4+treat*x5))}}$$

L'indicateur de traitement est généré au préalable en fonction du nombre d'observations désirées et de la proportion d'observations dans le groupe traitement spécifiée en entrée de la fonction.

Dans le cas d'une variable cible binaire, on utilise cette quantité comme étant la probabilité que Y prenne la valeur.

Dans le cas continu, on utilise cette quantité comme étant la moyenne de Y qu'on génère selon une distribution normale avec un écart-type de 0.1.

3.3. Préparation des jeux de données et validation croisée

Considérant la nécessité d'effectuer une validation croisée afin d'entraîner le modèle de second niveau qui déterminera les poids attribués aux prédictions de chaque modèle sous-jacent, ainsi que le besoin de séparer les jeux de données en traitement et contrôle pour

l'approche à deux modèles, la méthode requiert la préparation d'un nombre significatif de jeux de données différents.

Afin de permettre une validation suffisante tout en maintenant la complexité à un minimum, la validation croisée à 10-groupes a été retenue.

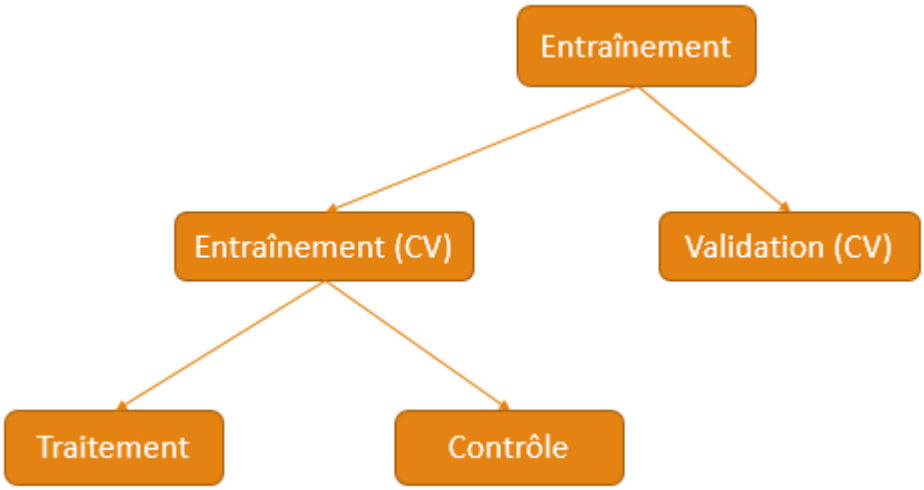

FIGURE 3.2. STRUCTURE D'UN JEU DE DONNÉES POUR LA VALIDATION CROISÉE

La préparation des jeux de données pour validation croisée se fait comme suit :

1. Trier le jeu d'entraînement de manière aléatoire
2. Produire 10 sous-ensembles du jeu d'entraînement, qui serviront de jeux de validation pour la validation croisée (10% chacun)
3. Produire 10 autres sous-ensembles du jeu d'entraînement, chacun excluant les observations contenues dans l'un des jeux produits à l'étape 2 (90% des données pour chacun)
4. Pour chaque jeu de données produit à l'étape 3, diviser en deux sous-ensembles, l'un contenant les observations se retrouvant dans le groupe traitement et l'autre avec les observations du groupe contrôle.

La méthode pratique d'implantation des jeux de données pour validation croisée est celle suggérée par Gasana (2018).

3.4. Élaboration des modèles sous-jacents

Dans le but de maintenir la comparabilité des différentes approches et de fixer le plus grand nombre de variables possible, il a été décidé que tous les modèles contenus dans ce mémoire soient de même type.

Bien que plusieurs options étaient possibles, les forêts aléatoires présentaient un certain nombre d'avantages qui en justifiaient le choix aisément :

- Étant une méthode d'ensemble, ils offrent une performance relativement bonne dans un large éventail de situations.
- Par leur nature, les forêts aléatoires font automatiquement une sélection de variables, permettant de faire une certaine abstraction de ce facteur afin de concentrer la recherche sur les comparaisons entre les différents modèles.
- Des algorithmes permettant leur usage sont déjà directement implantés sous forme de packages R, permettant leur implantation rapidement et simplement et ce, pour les trois approches existantes en modélisation Uplift.

Pour toutes ces raisons, les forêts aléatoires s'imposaient comme méthode de choix pour supporter le développement de l'étude.

Chaque modèle a été entraîné deux fois. L'une, sans validation croisée, c'est-à-dire avec toutes les observations, permet d'obtenir les prédictions finales qui sont directement faites sur le jeu de données de validation. La seconde, entraînée sur la base de la validation croisée, permet d'entraîner les modèles utilisés pour établir les poids donnés par le modèle de stacking (Figure 3.3). Cette manière permet d'éliminer le biais naturel vers le modèle sous-jacent le plus performant tel qu'énoncé par Breiman (1996), de même que limiter le surapprentissage. Cela assure ainsi que le modèle sera généralisable et performera de manière adéquate sur les données de validation.

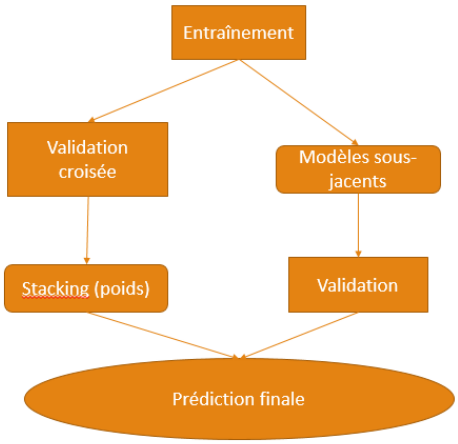

FIGURE 3.3. CHAQUE MODÈLE EST ENTRAÎNÉ DEUX FOIS, L'UNE PAR VALIDATION CROISÉE, L'AUTRE SANS, C'EST-À-DIRE SUR L'ÉCHANTILLON COMPLET. LES POIDS OBTENUS PAR VALIDATION CROISÉE SUR LE JEU D'ENTRAÎNEMENT SONT ENSUITE APPLIQUÉS SUR LES PRÉDICTIONS DES MODÈLES ENTRAÎNÉS

3.4.1. Méthode à deux modèles

Pour la méthode à deux modèles, c'est le package R (R Core Team, 2018) randomForest (Breiman, Cutler, Liaw, & Wiener, 2018) qui a été utilisé, avec les hyperparamètres par défaut suggérés par les auteurs. Bien que cette approche ne soit pas optimale d'un point de vue de performance brute, elle permet néanmoins de contrôler ces variables pour assurer la comparabilité des modèles et minimiser les facteurs qui pourraient nuire à l'évaluation subséquente de la performance du stacking comparativement aux modèles sous-jacents.

3.4.2. Méthode Class Transformation

La méthode Class Transformation étant elle aussi compatible avec des algorithmes standards, le package randomForest a également été utilisé pour l'entraînement de ces modèles, toujours avec les hyperparamètres par défaut définis par les auteurs.

Pour la transformation de la variable-cible, c'est la méthode de Athey et Imbens (2015) qui a été employée. Concrètement, la méthode assume donc que la proportion de traitement et de contrôle n'est pas égale à 50% et que la condition d'indépendance n'est pas respectée, et calcule donc systématiquement le score de propensité. Cette hypothèse se fait au coût d'une légère augmentation de l'erreur dans les cas où l'assignation de traitement est indépendante et égale à 50% du jeu de données puisqu'il y a une étape d'estimation supplémentaire qui ne serait pas nécessaire. Toutefois, dans un contexte de données réelles

collectées sur une base observationnelle, il est raisonnable de supposer le non-respect de la condition puisque le traitement est presque toujours dépendant des variables explicatives, sauf dans le cas d'études contrôlées conçues spécifiquement dans le but de respecter cette condition (Austin, 2011). Les choix méthodologiques propres à l'étude de cas sur des données réelles seront présentés séparément dans le chapitre 5.

En pratique, donc, dans le cas des simulations, un premier modèle prédictif estime la valeur de l'indicateur d'assignation de traitement en fonction des variables explicatives. Ce modèle est lui aussi basé sur des forêts aléatoires avec le package randomForest. La prédiction résultant de ce modèle est ensuite insérée dans la formule proposée par Jaskowski et Jaroszwicz (2012) pour créer la variable transformée finale qui sert de variable-cible dans le modèle prédictif proprement dit.

3.4.3. Méthode de modélisation directe

Contrairement aux deux autres méthodes, il est impossible d'utiliser un algorithme prédictif standard pour modéliser directement l'effet de traitement dans le cadre de modèles Uplift. Pour cette raison, cette méthode utilise un algorithme différent des deux autres approches.

Toujours sous R, deux algorithmes utilisant des forêts aléatoires ont été implantés pour ce type de problèmes, l'un dans le package Uplift (Guelman, 2014) avec la fonction upliftRF et l'autre dans le package grf, pour Generalized Random Forest, ou forêt aléatoire généralisée (Tibshirani, et al., 2018).

Les deux méthodes ont été employées dans le contexte de ce mémoire. Lorsque la variable-cible Y est binaire, la méthode upliftRF est significativement plus performante que celle de grf, la rendant intéressante à utiliser lorsque tel est le cas.

Par contre, cette méthode est limitée à ce cas précis et est incapable de gérer les données où la variable-cible est continue. Pour les jeux de données où Y est continue, le package grf et sa fonction causal.forest sont donc employés.

Dans les deux cas, les hyperparamètres par défaut suggérés par les auteurs sont maintenus, à l'exception du nombre d'arbres. En effet, la valeur par défaut est de 500 arbres pour toutes les méthodes sauf upliftRF, pour lequel 200 arbres sont utilisés. Pour rendre les modèles à armes égales, le nombre d'arbres est fixé à 500 pour tous les modèles.

3.5. Élaboration des modèles de second niveau

3.5.1. Problématique de la variable-cible

Lorsque vient le temps de mettre en place des modèles de second niveau afin de calibrer les poids accordés à chaque modèle sous-jacent, immédiatement le problème central de la modélisation Uplift refait surface et prend une importance d'autant plus capitale.

En effet, en ne connaissant pas la valeur réelle du lift pour chaque client, la même problématique s'en trouve formulée autrement dans un contexte de stacking : quelle est la variable-cible à optimiser avec le modèle de second niveau?

Le lift étant inconnu, il faut alors utiliser un approximateur ou une mesure de performance qui jouera le rôle de cible à optimiser en utilisant les poids des modèles sous-jacents dans le modèle de stacking.

Une première approche qui a été explorée consiste à utiliser la variable transformée Y_i^* dont la valeur a été calculée pour la méthode Class Transformation. En effet, Gerardy et Gutierrez (2016) ont démontré qu'à la limite, l'espérance moyenne de cette variable tend vers l'effet de traitement moyen $\tau(x)$. En pratique, cette variable s'est avérée être un estimateur d'une qualité discutable, ne permettant pas d'optimiser adéquatement la fonction de second niveau pour obtenir une combinaison fiable des modèles sous-jacents, peu importe la méthode de combinaison utilisée.

Une autre méthode étudiée consistait à utiliser le Qini comme fonction-cible à maximiser à l'aide d'un algorithme d'optimisation. Cette idée s'est également avérée futile puisque les algorithmes d'optimisation devenaient radicalement instables ou encore fournissaient des valeurs absurdes comme poids à accorder aux modèles. Le problème provient de la nature même du Qini comme mesure d'ordre. En effet, le Qini ne mesurant que l'ordre des observations, pas leur lift réel, une infinité de solutions sont possibles lorsque cette mesure est utilisée comme variable-cible. La méthode d'optimisation, donc, ne parvient pas à trouver des poids optimaux.

Les deux autres méthodes qui ont été tentées se sont avérées relativement performantes et ont été retenues pour les résultats qui suivront. Ces deux méthodes ayant été implantées, elles seront présentées dans les deux prochaines sections.

3.5.2. Méthode avec agrégation par strates

Une troisième méthodologie fut proposée par mon directeur Denis Larocque (communication personnelle, 2018). Elle consiste essentiellement à utiliser les prédictions de base des trois modèles en accordant à chacune un poids égal, puis ordonner les données selon cette estimation et stratifier les données en groupes de 40 individus (20 traitement et 20 contrôle) de manière à obtenir une estimation du lift moyen pour chaque groupe. En détails :

1. Estimer les trois modèles de premier niveau par validation croisée de manière à obtenir les lifts estimés pour chaque observation.
2. Calculer un estimateur préliminaire en utilisant des poids égaux pour chaque prédicteur, donc 1/3 pour chacun des trois modèles.
3. Ordonner les observations du groupe contrôle en fonction de l'estimateur préliminaire, du plus petit lift au plus grand.
4. Séparer les observations du groupe contrôle en groupes de 20 observations sur la base de l'ordonnancement de l'étape 3.
5. Ordonner les observations du groupe traitement en fonction de l'estimateur préliminaire, du plus petit lift au plus grand.
6. Séparer les observations du groupe traitement en groupes de 20 observations sur la base de l'ordonnancement de l'étape 5.
7. Pour chaque groupe, calculer la proportion de 1 se retrouvant dans traitement et dans contrôle respectivement.
8. Pour chaque groupe, soustraire la proportion de 1 dans le groupe contrôle de celle du groupe traitement. Cela deviendra l'estimateur du lift réel, pour ce groupe.
9. Calculer, par groupe, la prédiction moyenne de chacun des trois modèles obtenus par validation croisée.
10. Utiliser l'estimateur préliminaire créé plus tôt comme cible pour le modèle de stacking, où l'on minimise l'erreur entre cet estimateur et le lift estimé par le modèle combiné, au niveau de chaque groupe.

Une fois ces manipulations complétées, le modèle de stacking de second niveau sera obtenu en modélisant cet estimateur du lift réel par rapport aux prédictions moyennes par groupe. Le modèle donnera en sortie des poids qui seront ensuite appliqués aux prédictions des modèles de base sur le jeu de validation pour obtenir une prédiction individuelle finale par stacking.

Lorsque le nombre d'observations de traitement et de contrôle n'est pas équilibré, c'est en fonction du plus petit des deux groupes que l'on déterminera le nombre de strates et ainsi la taille des groupes, de sorte que chaque sous-groupe demeure significatif. Par exemple, dans le cas d'un jeu de données avec 10 000 observations dont 70% dans le groupe traitement, on pourrait choisir de créer 100 strates; dans le groupe traitement, chaque strate contiendrait 70 observations et dans le groupe contrôle, 30.

3.5.3. Méthode par matching

Une autre approche possible consiste à utiliser un algorithme d'appariement pour associer chaque observation de traitement avec une observation du groupe contrôle en fonction de leur ressemblance au niveau des variables explicatives.

On peut alors supposer que la différence entre le Y traitement et le Y contrôle représente le lift individuel pour cette observation; ce lift servira donc de cible pour le modèle de second niveau : il cherchera à prédire cette différence de Y en fonction des prédictions des trois modèles sous-jacents obtenus par validation croisée.

Par la suite, les prédictions des modèles sous-jacents sur le jeu de validation seront combinées en utilisant les poids obtenus en sortie de ce modèle de stacking pour produire la prédiction individuelle combinée.

L'algorithme d'appariement retenu est celui contenu dans le package Matching (Sekhon, 2018). Celui-ci utilise une grande quantité de méthodes d'appariement différentes qu'il combine pour ensuite associer chaque observation de traitement avec une observation du groupe contrôle. Pour trouver un match pour les observations du groupe contrôle, une petite manipulation a été effectuée pour inverser les groupes traitement et contrôle aux yeux de l'algorithme, puis les vecteurs correspondants ont été combinés pour former à nouveau un jeu de données complet. Le lift calculé ci-dessus a été ajouté au jeu de données initial, puis utilisé comme cible pour le modèle de second niveau.

3.5.4. Méthode de combinaison

Plusieurs fonctions ont été étudiées afin de combiner les trois modèles sous-jacents.

Une méthode intéressante qui a été retenue est celle d'utiliser un algorithme d'optimisation afin de trouver les poids à accorder à chaque modèle sous-jacent afin de minimiser l'erreur quadratique moyenne entre la prédiction combinée et l'estimateur du lift réel.

Dans le cas de la méthode par matching, la fonction serait donc la suivante :

$$min \frac{1}{n} \sum_{i=1}^{n} ((x_1 * \widehat{\tau_{1i}} + x_2 * \widehat{\tau_{2i}} + \ x_3 * \widehat{\tau_{3i}}) - \tau_i)^2$$

où τ_i est la valeur du lift obtenue par matching et $\widehat{\tau_{1i}}, \widehat{\tau_{2i}} et\ \widehat{\tau_{3i}}$ sont les prédictions individuelles des trois modèles sous-jacents.

Dans le cas de la méthode par strates, la formule est légèrement différente :

$$min \frac{1}{k} \sum_{i=1}^{k} ((x_1 * \widehat{\tau_{1k}} + x_2 * \widehat{\tau_{2k}} + \ x_3 * \widehat{\tau_{3k}}) - \tau_k)^2$$

où τ_k est la valeur moyenne observée par groupe et $\widehat{\tau_{1k}}, \widehat{\tau_{2k}} et\ \widehat{\tau_{3k}}$ sont les prédictions des trois modèles sous-jacents agrégées au niveau de chaque groupe.

Dans les deux cas, x_1, x_2 et x_3 correspondent aux poids de chaque modèle et donc les trois cibles à optimiser.

En pratique, l'erreur quadratique moyenne a été calculée à l'aide de la fonction MSE présente dans le package MLmetrics (Yan, 2016).

Deux méthodes d'optimisation ont été tentées sur cette fonction. La première est la méthode Broyden-Fletcher-Goldfarb-Shanno (BFGS) telle qu'implantée dans la fonction optim incluse de base avec R (R Core Team, 2018). Cette méthode est implantée sans contrainte, laissant à l'algorithme la latitude de déterminer la valeur qui minimise l'EQM.

La seconde méthode est un algorithme d'optimisation globale utilisant la méthode d'évolution différentielle élaborée par Storn et Price (1997) telle qu'implantée dans le package DEoptim (Ardia, Mullen, Peterson, Ulrich, & Boudt, 2016). La méthode recherche

par 200 itérations le minimum de la fonction et fournit les trois paramètres qui la minimisent. Ces trois paramètres sont les poids de chaque modèle. Pour cette seconde méthode cependant, une contrainte de non-négativité a été appliquée à chaque paramètre, tel que suggéré par Breiman (1996). La fonction requiert également un maximum, qui a été fixé à 10 de manière à ce que les poids ne l'atteignent jamais en pratique (puisque les poids auront tendance à se trouver autour de 1 car l'algorithme tente de faire une prédiction à partir de prédictions).

Une troisième méthode mise en place consiste à entraîner une régression linéaire tentant de prédire l'estimateur en fonction des trois prédictions produites par les modèles sous-jacents. L'ordonnée à l'origine de la régression est fixée à zéro afin d'obtenir des paramètres qui reflètent l'importance relative de chaque modèle. Cette régression est entraînée à l'aide de la fonction lm incluse de base dans le logiciel R (R Core Team, 2018) :

$$\widehat{\tau_i} = \beta_1 * \hat{\tau}_{1i} + \beta_2 * \hat{\tau}_{2i} + \beta_3 * \hat{\tau}_{3i} + \varepsilon.$$

3.6. Évaluation de la performance

Dans la mesure où la méthode a été testée sur, d'une part, des jeux de données simulées et, d'autre part, des campagnes marketing réellement exécutées par une entreprise, il convient de faire la distinction entre ces deux types de données au niveau de l'évaluation de la performance. En effet, le fait que la fonction qui sous-tend la relation entre les variables explicatives et le lift est connue sur un ensemble de données simulées permet de mesurer beaucoup plus précisément la performance de la technique sur ces données, ce qui peut ensuite servir de baromètre pour les modèles élaborés dans le cadre de l'étude de cas.

3.6.1. Mesures sur données pratiques

Deux mesures de performance principales ont été employées dans le cadre de l'étude de cas. Dans un premier temps, le Qini est mesuré et évalué pour chaque modèle sous-jacent et chaque modèle de stacking. Le Qini étant traditionnellement utilisé pour mesurer la performance de modèles Uplift et pouvant être calculée aisément tant sur une simulation que des données réelles, celui-ci est la principale méthode qui permettra de comparer la performance des différents modèles entre eux. La représentation graphique de la courbe

des gains sera également utilisée pour l'évaluation du modèle et l'interprétation des résultats.

Le lift brut sera également calculé sur le premier décile des données identifiées par le modèle comme seconde mesure de performance. La raison est fortement basée sur la réalité: la plupart des entreprises qui utilisent la méthodologie Uplift tentent explicitement de cibler les premiers déciles afin de maximiser le retour sur investissement de leurs campagnes. Entre deux modèles dont la performance globale est équivalente, on choisira généralement celui qui offre la meilleure performance sur les premiers déciles.

Toutes ces mesures de performance sont calculées sur des données de validation distinctes du jeu d'entraînement, de manière à avoir une représentation fidèle de la performance réelle en limitant le surapprentissage capté dans les mesures de performance.

3.6.2. Mesures sur données simulées

Le cas particulier des données simulées présente un avantage qui est absent des données réelles. En effet, la fonction qui lie le lift aux variables explicatives peut être contrôlé, et celui-ci peut par conséquent être connu. Dans ce contexte, il est donc possible d'évaluer directement l'erreur quadratique moyenne (EQM) du modèle élaboré par rapport au lift réel pour cette observation. Il s'agit de la mesure la plus fiable disponible dans ce cas.

L'EQM est donc calculée pour chaque modèle et utilisée dans un premier temps pour évaluer l'efficacité des modèles simulés. Un parallèle est également fait entre le Qini et l'EQM afin de déterminer s'il est systématiquement possible de conclure qu'une EQM plus faible correspond à un Qini plus élevé. Cela donnera une meilleure idée de la fiabilité des résultats du modèle dans un contexte d'affaires et permettra de comparer plus aisément les résultats entre les différents jeux de données.

4. Résultats sur données simulées

4.1. Notes sur les résultats

Les résultats présentés ici ont fait l'objet d'une sélection préalable. En effet, les méthodes d'optimisation sans contrainte et de régression pour le stacking ayant été nettement inférieures à la méthode d'optimisation avec contrainte de non-négativité, et ce, dans toutes les situations rencontrées, il s'est avéré peu intéressant de les analyser dans le texte. Par souci de transparence, les résultats complets sont cependant disponibles en annexe au mémoire, tant pour les simulations que pour l'étude de cas.

Tout au long des simulations, 100 jeux de données différents sont générés pour chaque valeur de chaque paramètre afin de valider l'efficacité sur une large quantité de jeux de données générés aléatoirement et d'éviter toute anomalie due à un petit nombre de simulations.

Les jeux de données générés ont toujours la même taille peu importe la méthode de génération des données, soit de 6 000 observations pour les données d'entraînement et 10 000 observations pour les données de test. Les résultats présentés sont donc la moyenne de ces 100 simulations, pour chaque valeur de chaque paramètre étudié

4.2. Variation de la taille de l'effet Uplift – binaire

4.2.1. Performance

L'analyse de l'erreur quadratique moyenne sur ce jeu de données (présenté dans la section 3.2.1) démontre une différence significative entre les résultats lorsque le multiplicateur de l'effet Uplift est faible et ceux lorsqu'il est élevé. En effet, l'erreur est très élevée lorsque le multiplicateur est de 1, puis diminue rapidement lorsqu'on l'augmente avant d'atteindre un plateau autour de 3. Ce phénomène est observé sur tous les modèles à l'exception notable du modèle Class Transformation (Figure 4.1).

Au chapitre des différences entre les modèles sous-jacents, lorsque l'effet est faible, les méthodes deux modèles et de modélisation directe sont très similaires, mais la méthode directe prend l'avantage lorsque l'effet Uplift est plus grand. La méthode Class

Transformation est pour sa part constamment inférieure aux deux autres modèles sous-jacents.

Dans la plupart des scénarios simulés, le stacking par matching performe de manière très similaire à la modélisation directe. Une exception notable est observée lorsque l'effet est très petit; dans ce cas, le matching détient un net avantage.

Quant au stacking par strates, lorsque l'effet est petit, il fait sensiblement moins bien que la plupart des modèles, mais au fur et à mesure que l'effet Uplift augmente, il améliore sa performance pour devenir le meilleur modèle dans des cas où l'effet Uplift est plus grand.

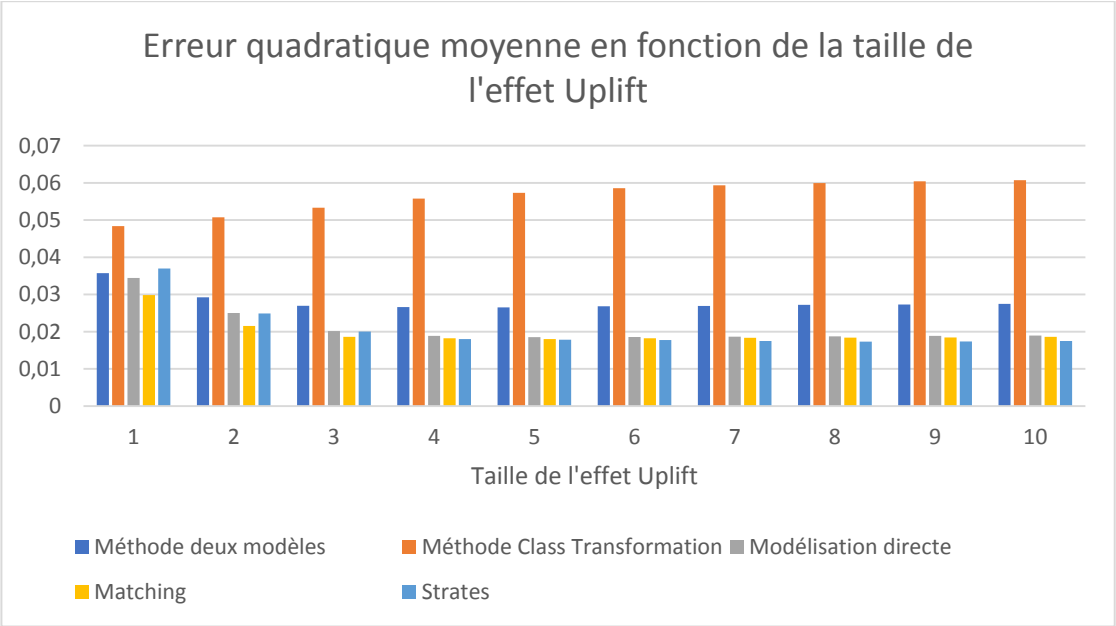

FIGURE 4.1. ERREUR QUADRATIQUE MOYENNE EN FONCTION DE LA TAILLE DE L'EFFET UPLIFT, VARIABLE-CIBLE BINAIRE.

Les mêmes constats se retrouvent de manière similaire lorsque le Qini devient la mesure de performance étudiée. Lorsque l'on part d'un très petit effet Uplift, l'amélioration de performance est significative au départ, puis atteint rapidement un plateau où les gains sont minimes (Figure 4.2).

Les trois modèles sous-jacents performent de manières comparables lorsque l'effet Uplift est grand (à l'exception de Class Transformation qui est toujours plus faible), mais la

variation entre ceux-ci est plus grande lorsque l'effet est plus faible. Ici également, la méthode par matching est très près de la performance de la modélisation directe, étant parfois légèrement plus faible, parfois légèrement plus élevée. La différence entre les deux méthodes n'est pas significative.

On retrouve ici également le phénomène où les deux approches de stacking donnent des résultats similaires sur la quasi-totalité des simulations. L'exception se trouve lorsque le multiplicateur du signal Uplift est faible; dans ce contexte, le matching détient un certain avantage et devient le meilleur modèle.

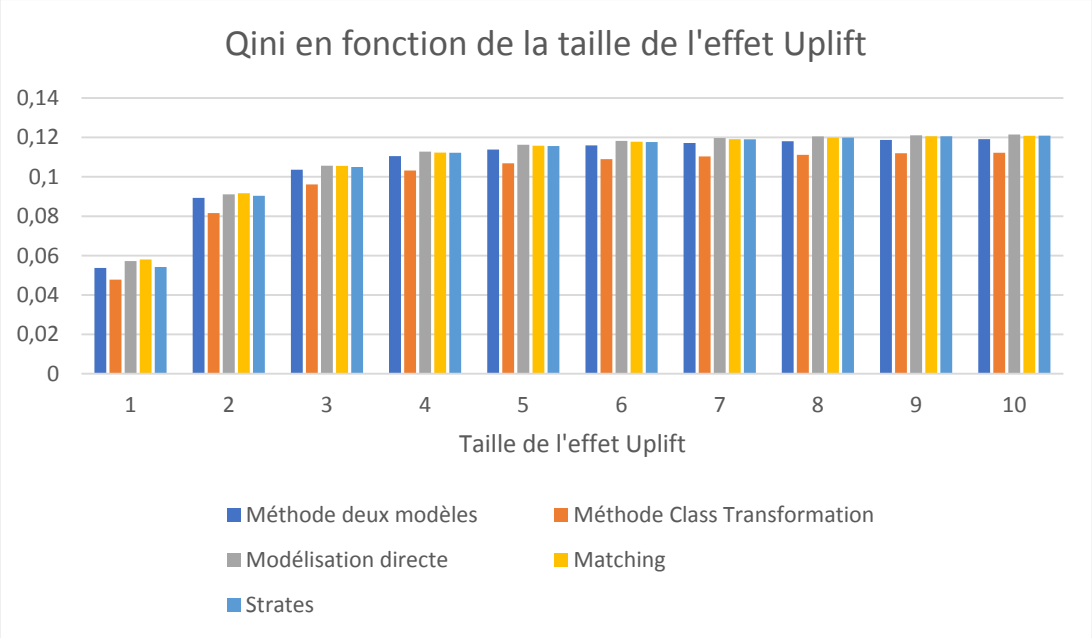

FIGURE 4.2. QINI EN FONCTION DE LA TAILLE DE L'EFFET UPLIFT, VARIABLE-CIBLE BINAIRE.

Sur la base du lift observé sur le premier décile de chaque modèle, les deux méthodes de stacking sont là encore équivalentes, obtenant sensiblement les mêmes lifts observés sauf lorsque l'effet est très petit, où le matching est le meilleur modèle (Figure 4.3).

Lorsque l'effet est élevé, ni l'une ni l'autre des méthodes de stacking ne parviennent à faire mieux que la modélisation directe, qui est le meilleur modèle dans l'ensemble pour cette mesure.

Il est difficile de déclarer un modèle supérieur sur cette simulation. Dans l'ensemble, les conclusions dépendent largement du contexte de l'analyse, plusieurs modèles étant supérieurs dans des contextes différents. Il est cependant intéressant de noter que lorsque

le matching fait relativement bien, la méthode par strates fait moins bien et vice-versa. Les deux méthodes peuvent donc être intéressantes à utiliser dans des cas différents, en fonction de la taille de l'effet du traitement. En effet, le matching est ici supérieur lorsque l'effet est petit alors que la méthode strates est faible; l'inverse est vrai à plus grande taille d'effet Uplift.

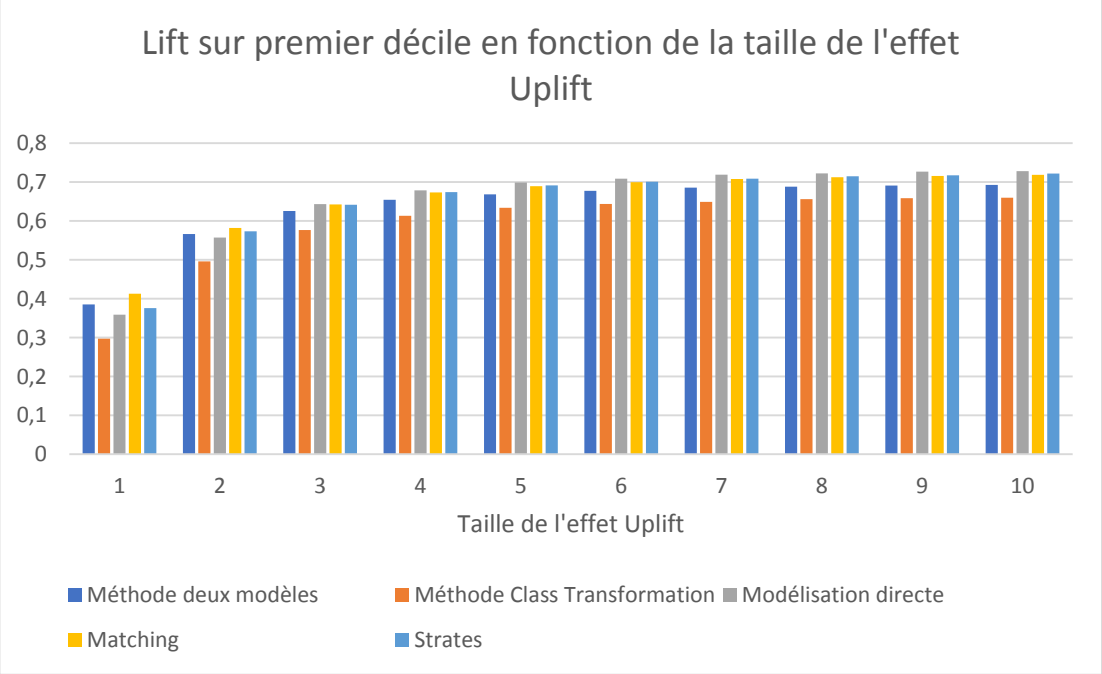

FIGURE 4.3. LIFT SUR LE PREMIER DÉCILE EN FONCTION DE LA TAILLE D'EFFET UPLIFT, VARIABLE-CIBLE BINAIRE.

Sur cette méthode de simulation, le matching a donc un net avantage de performance, particulièrement lorsque l'effet Uplift est très petit et donc plus difficile à déceler par chaque modèle. Par la suite, lorsque l'effet est plus grand, il devient difficile de distinguer la performance de la modélisation directe, du matching et de la méthode par strates, les trois ayant une performance totale similaire.

4.2.2. Poids accordés aux modèles

Sur cette méthode de génération des données, la méthode par appariement partage le poids le plus élevé entre les méthodes deux modèles et modélisation directe (Figure 4.4). En effet, lorsque l'effet Uplift est très faible, la méthode deux modèles est la plus importante

dans le modèle, puis son impact diminue graduellement lorsque le multiplicateur augmente entre 2 et 9. L'inverse est vrai de la modélisation directe dont le poids augmente graduellement sur la même période.

Un point d'inflexion intéressant apparaît lorsque le multiplicateur est de 7. En effet, les poids de ces deux modèles sont à toutes fins pratiques identiques pour ce multiplicateur, et la modélisation directe dépasse ensuite la méthode deux modèles lorsque le multiplicateur est plus élevé.

La méthode Class Transformation est pour sa part plus faible et très stable peu importe la valeur du multiplicateur, son poids étant toujours autour de 0.2 sans égard aux autres variations avec cette méthode.

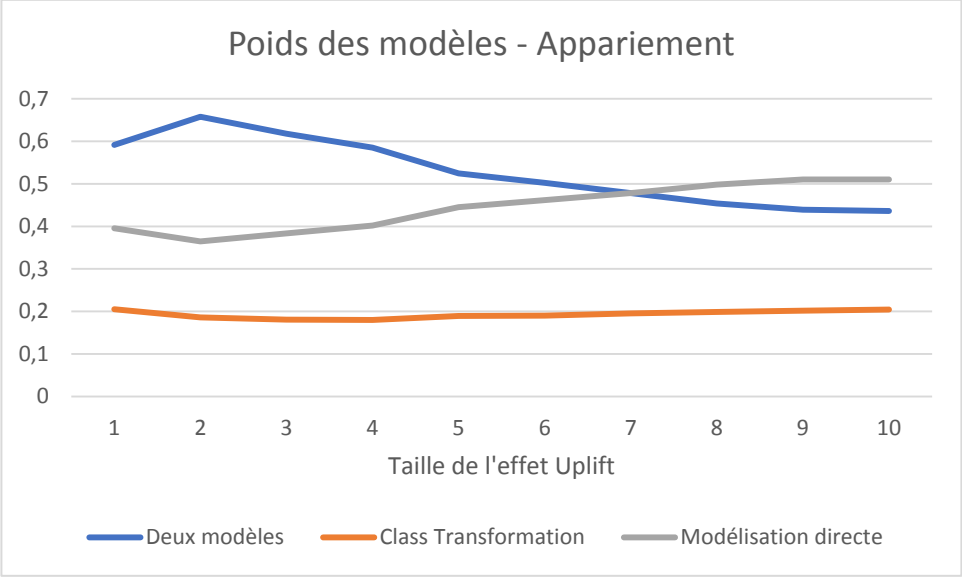

FIGURE 4.4. POIDS ATTRIBUÉS AUX MODÈLES SELON LA MÉTHODE D'APPARIEMENT (MATCHING), VARIABLE-CIBLE BINAIRE.

Les poids sont très différents lorsque la méthode par strates est utilisée (Figure 4.5). En effet, cette méthode se décline en deux étapes : lorsque le multiplicateur du lift est très faible (1 ou 2), la méthode deux modèles prend beaucoup de poids et la méthode de transformation de classe arrive tout juste derrière; la modélisation directe a un poids très faible au départ.

Son poids augmente cependant très rapidement lorsque le multiplicateur augmente pour devenir le modèle le plus important à partir d'un multiplicateur de 3. Les deux autres

continuent de diminuer en importance pour n'apporter qu'une contribution marginale, avec des poids allant de 0.13 à 0.20, lorsque le signal Uplift est très grand.

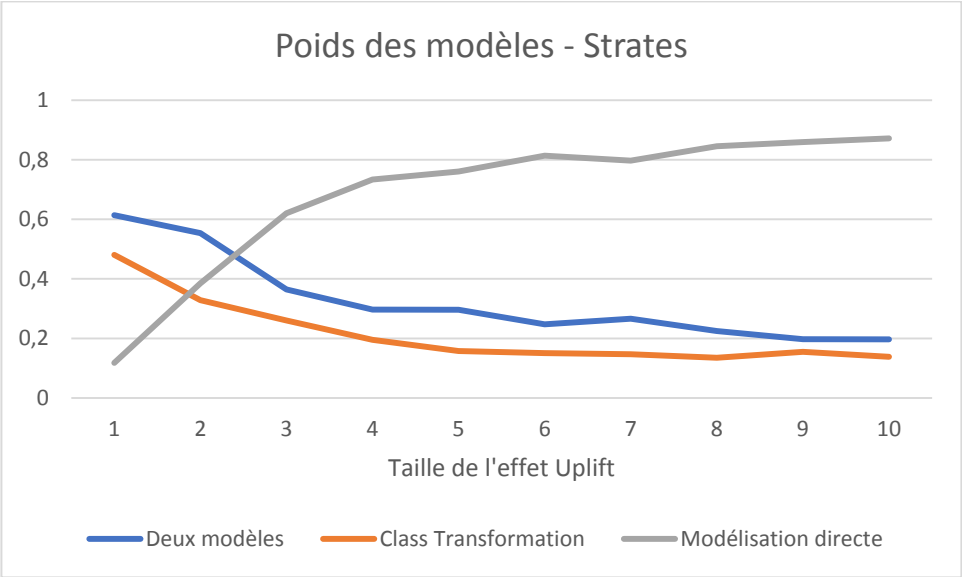

FIGURE 4.5. POIDS ACCORDÉS AUX MODÈLES SOUS-JACENTS SELON LA MÉTHODE PAR STRATES, VARIABLE-CIBLE BINAIRE

4.3. Variation de la taille de l'effet Uplift - continue

4.3.1. Performance

Lorsque l'effet Uplift est très faible, il est apparent que tous les modèles sous-jacents ont de la difficulté à capturer les éléments significatifs dans les données. La performance des modèles est en effet très faible lorsque le multiplicateur est de 1, mais augmente très rapidement lorsqu'on l'augmente à partir de ce point pour ensuite atteindre un plateau avec les tailles d'effet plus élevées (Figure 4.6).

Les différences de performance entre les modèles traditionnels sont significatives, Class Transformation ayant toujours une erreur quadratique moyenne nettement plus élevée que la méthode deux modèles, elle-même moins performante que la modélisation directe lorsque la taille de l'effet est petite. Lorsque l'effet est plus grand, la méthode deux modèles prend un léger avantage. Toutefois, la direction générale de l'effet d'une variation de la taille du signal est relativement constante d'un modèle à l'autre: l'erreur diminue lorsque l'effet uplift augmente.

Au niveau des modèles de stacking, les résultats sont intéressants également lorsque le signal est très faible. La méthode par strates suit exactement la tendance des modèles sous-jacents, avec une erreur nettement plus élevée que lorsque le signal Uplift est plus élevé, bien qu'elle soit tout de même inférieure à tous les modèles sous-jacents et améliore donc la performance.

La méthode par matching est clairement la plus intéressante sur ce jeu de données à variable-cible continue, cependant. Malgré un léger désavantage de performance par rapport à la méthode par strates lorsque l'effet dépasse 3, son avantage est en contrepartie significatif lorsque le signal uplift est très petit. Non seulement le matching parvient-il à diminuer l'erreur de manière globale par rapport à toutes les autres méthodes, le gain de performance sur le meilleur modèle sous-jacent variant entre 8% et 36%, cet effet s'en trouve amplifié lorsque l'effet Uplift a un multiplicateur faible. La stabilité du modèle par matching est donc un point significatif à soulever, puisque sa performance est fortement moins affectée par une variation de la taille de l'effet Uplift; le modèle est capable de prédictions relativement fiables dans toutes circonstances, contrairement aux modèles existants dont la performance se dégrade lorsque l'effet est petit.

Les méthodes par matching et par strates sont relativement équivalentes lorsque l'effet Uplift est de taille moyenne, leur erreur étant très proche l'une de l'autre alors que les deux méthodes offrent toujours un gain de performance sur chacun des modèles sous-jacents. Lorsque l'effet est plus élevé, la méthode par strates prend un léger avantage sur la méthode par matching.

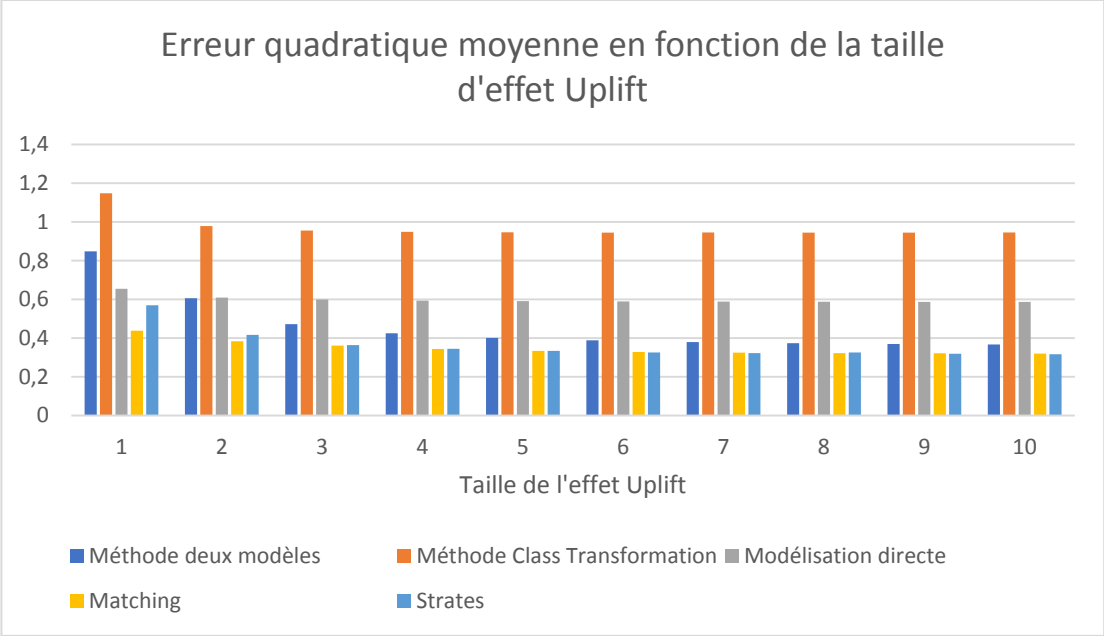

FIGURE 4.6. ERREUR QUADRATIQUE MOYENNE EN FONCTION DE LA TAILLE D'EFFET UPLIFT, VARIABLE-CIBLE CONTINUE

L'effet de la variation de la taille du signal Uplift sur les lifts observés dans le premier décile des résultats est similaire (Figure 4.7). Au départ, l'effet d'une augmentation de la force du signal est important sur la plupart des modèles, puis atteint un plateau autour d'un multiplicateur de 3.

La méthode de modélisation directe fait exception; la taille de l'effet Uplift ne semble pas affecter sa capacité à identifier l'effet Uplift dans le premier décile puisque celui-ci varie très peu. C'est également le cas de la méthode de stacking par matching, qui demeure toujours très près de la performance de la modélisation directe sur cette base. Le matching est légèrement supérieur dans la plupart des cas, mais la différence est minime.

Le stacking par strates suit tout juste derrière, les résultats étant conformes à ce qui a été observé sur la base de l'EQM. Ce modèle performe mieux que la méthode deux modèles et Class Transformation, mais suit tout de même la tendance à la dégradation de la performance sur des jeux de données avec un petit effet Uplift, alors que la méthode par matching mitige cet effet. De manière générale, bien qu'intéressante d'un point de vue comparatif avec les modèles sous-jacents, la méthode par strates est surclassée par le matching dans le contexte de cette simulation.

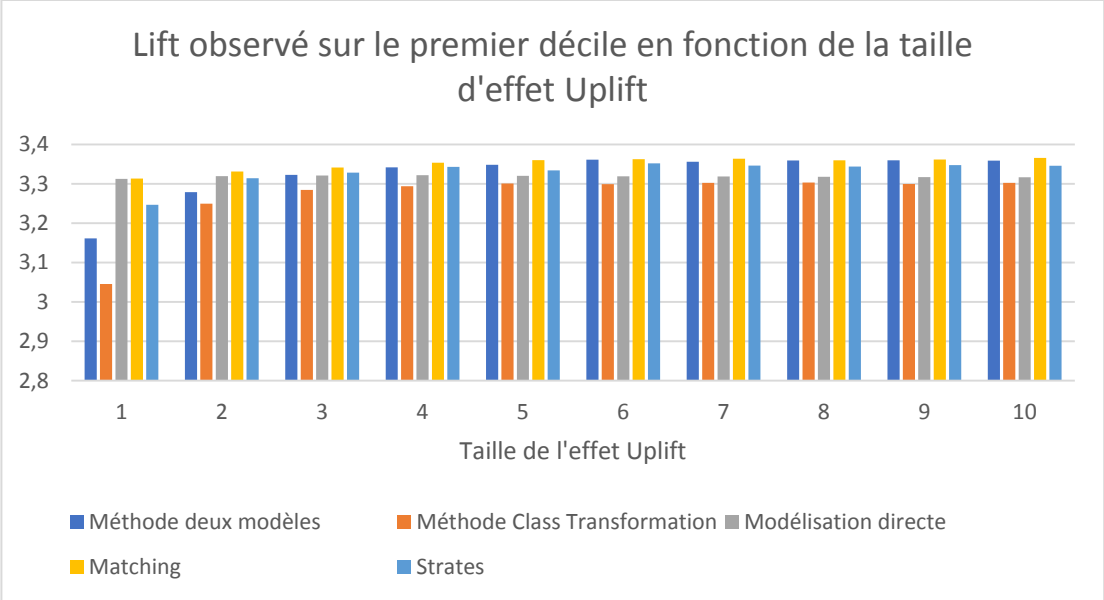

FIGURE 4.7. LIFT SUR LE PREMIER DÉCILE EN FONCTION DE LA TAILLE D'EFFET UPLIFT, VARIABLE-CIBLE CONTINUE.

Dans ce contexte de variable-cible continue, l'apport global de performance dû au stacking est nettement plus prononcé que dans le cas binaire. On observe toujours cette même tendance où le stacking apporte des gains plus prononcés lorsque l'effet Uplift est petit. Sur la base de l'EQM, seule la méthode deux modèles s'approche de la performance de l'une ou l'autre des méthodes de stacking lorsque l'effet est plus grand, et aucune ne s'en approche lorsque l'effet est petit.

Dans l'ensemble, l'apport du stacking est plus prononcé dans le cas continu que dans le cas binaire étudié précédemment.

4.3.2. Poids accordés aux modèles

Avec cette méthode de génération de données, il y a une plus grande variabilité d'une simulation à une autre lorsque les mêmes paramètres sont conservés. Il est donc plus difficile d'en tirer des conclusions définitives que sur les autres jeux de données. Cependant, certaines tendances s'en dégagent tout de même.

On peut observer qu'avec la méthode d'appariement, les poids sont très près les uns des autres lorsque le lift est petit, entre 40 et 50% pour les trois modèles sous-jacents (Figure 4.8). Le poids de la modélisation directe est le plus important lorsque le lift est de 1, suivi de la méthode deux modèles et de la transformation de classe en troisième.

Par la suite, la modélisation directe diminue rapidement de poids pour devenir marginale voire inexistante, alors que la transformation de classe augmente rapidement au début pour devenir le modèle le plus important lorsque le multiplicateur est de 2 ou 3, et diminue ensuite légèrement. La méthode deux modèles voit pour sa part une augmentation graduelle de son poids lorsque le multiplicateur augmente. Elle est la méthode qui a le plus de poids à partir d'un multiplicateur de 3.

Il est intéressant de noter le poids généralement élevé accordé au modèle de transformation de classe sur ce DGP, malgré sa performance individuelle médiocre. Il semble que l'algorithme de stacking soit en mesure de tirer une information intéressante de ce modèle en dépit de sa performance difficile.

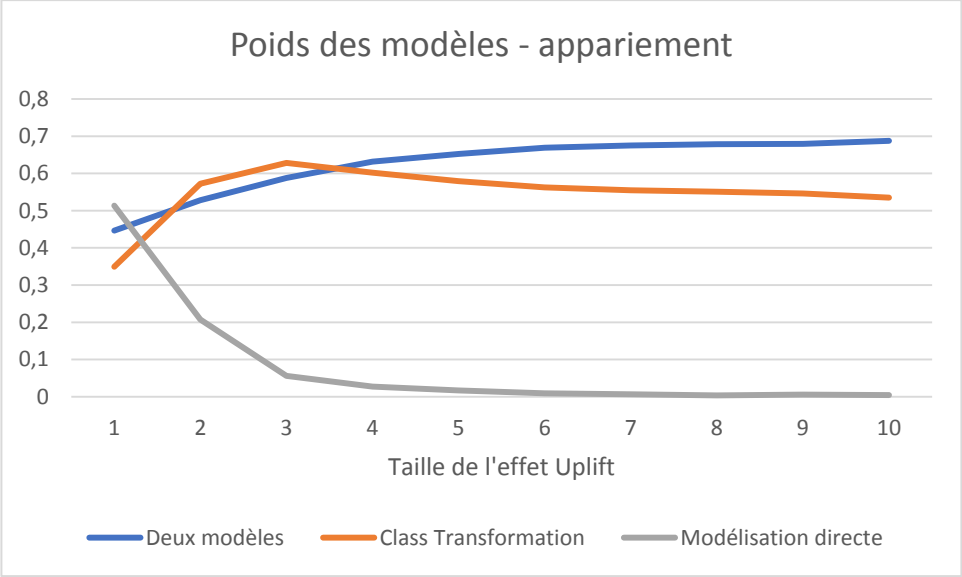

FIGURE 4.8. POIDS ATTRIBUÉS AUX MODÈLES SOUS-JACENTS SELON LA MÉTHODE D'APPARIEMENT, VARAIBLE-CIBLE CONTINUE.

Le même constat peut également être tiré lorsque la méthode par strates est utilisée pour combiner les modèles. En effet, à l'exception de lorsque le multiplicateur est de 1, le modèle de transformation de classe est toujours le premier modèle en importance selon cette méthode avec des poids stables entre 0.6 et 0.8 sans tendance claire (Figure 4.9).

La modélisation directe vient en deuxième, ayant un poids élevé lorsque l'effet uplift est petit, mais il diminue rapidement en poids pour ensuite s'échanger de place avec la méthode deux modèles lorsque le lift est élevé. En effet, ces deux méthodes ont des poids alternant autour de 0.3 à 0.4 lorsque le lift est élevé. La méthode deux modèles commence toutefois plus difficilement avec le poids le plus faible lorsque le lift est petit.

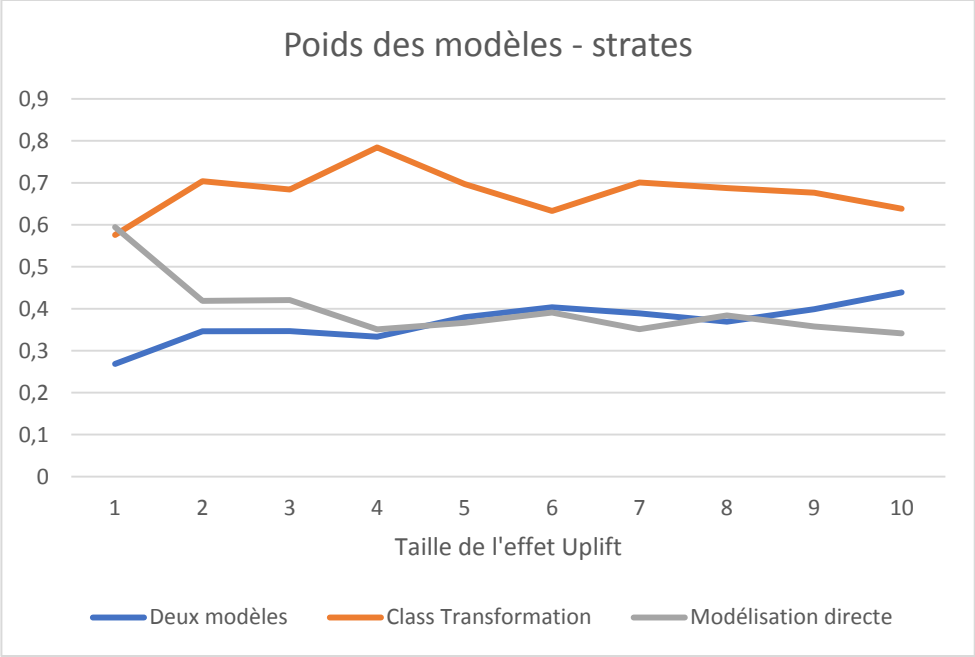

FIGURE 4.9. POIDS ATTRIBUÉS AUX MODÈLES SOUS-JACENTS PAR LA MÉTHODE STRATES, VARIABLE-CIBLE CONTINUE.

4.4. Simulation avec variation de la propension de traitement – binaire

4.4.1. Performance

Ce chapitre présente les résultats de la simulation où varie la propension de traitement, expliquée en détails dans la section 3.2.2.

Au premier coup d'œil, il est déjà apparent que les trois modèles sous-jacents réagissent de manière totalement différente à une variation de la proportion d'observations de traitement et de contrôle (Figure 4.10 et Tableau 4.1). En effet, lorsque c'est la méthode deux modèles qui est employée, l'erreur quadratique moyenne semble légèrement plus faible lorsque la proportion est près de 50% et augmente lorsque la proportion de traitement est moins équilibrée. La méthode Class Transformation ne fonctionne essentiellement pas lorsque la proportion de traitement est trop faible, l'EQM atteignant plus de 0.18 pour se rapprocher des autres lorsque la proportion de traitement augmente. Quant à la méthode de modélisation directe, son EQM semble augmenter légèrement et de manière très graduelle lorsque la proportion augmente. Globalement, c'est la modélisation

directe qui a l'erreur la plus faible, suivie de près par la méthode deux modèles. La méthode Class n'est généralement pas compétitive sur ce jeu de données.

TABLEAU 4.1. ERREURS QUADRATIQUES MOYENNES EN FONCTION DE LA PROPENSION DE TRAITEMENT, VARIABLE-CIBLE BINAIRE.

	Méthode deux modèles	Méthode Class Transformation	Modélisation directe	Appariement	Strates
20%	0,015034	0,185595	0,008592	0,011421	0,007928
30%	0,014579	0,110354	0,008801	0,011884	0,008348
40%	0,014009	0,060114	0,009104	0,011432	0,008637
50%	0,014012	0,030096	0,009444	0,011236	0,008802
60%	0,014063	0,017788	0,009706	0,01062	0,008963
70%	0,014293	0,020797	0,009539	0,009982	0,008482
80%	0,015171	0,041274	0,010607	0,010421	0,008897
Moyenne	**0,014452**	**0,066574**	**0,009399**	**0,010999**	**0,00858**

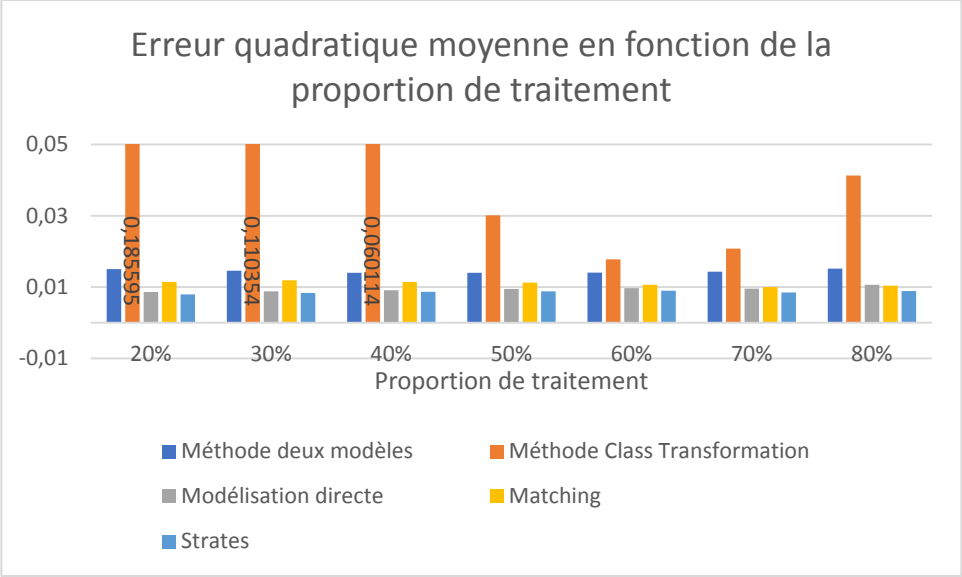

FIGURE 4.10. ERREUR QUADRATIQUE MOYENNE EN FONCTION DE LA PROPENSION DE TRAITEMENT, VARIABLE-CIBLE BINAIRE.

La méthode de stacking par appariement apporte une grande stabilité au niveau de l'erreur quadratique, celle-ci variant entre 1% et 1.2% pour toutes les proportions traitement/contrôle testées. Il s'agit d'une bonne performance globale et cette erreur se situant constamment entre la modélisation directe et la méthode deux modèles, sauf dans le cas où la proportion de données de traitement atteignait 80%; dans cette situation la méthode d'appariement est la meilleure. En termes de tendance, une faible tendance à la baisse de l'erreur est observée lorsque la proportion augmente; c'est donc dire que ce modèle fait de meilleures prédictions lorsque plus d'individus sont traités.

Ce constat explique largement les variations dans les poids attribués à chaque modèle, qui seront explorés dans la section suivante (Section 4.4.2, Figure 4.13). Lorsque que le poids de la modélisation directe diminue, son erreur augmente, et l'inverse est vrai de la méthode Class Transformation dont l'erreur tend à diminuer.

Pour le stacking par strates, la performance nette saute aux yeux au premier regard. En effet, l'erreur est plus faible que la modélisation directe, meilleur modèle sous-jacent. Cette approche est donc la plus performante dans l'ensemble sur la base de l'erreur quadratique moyenne. Elle est également relativement stable, bien que l'erreur augmente légèrement avec la proportion de traitement, tout comme la modélisation directe. Les poids en révèlent la raison: sur cette méthode de génération de données, le modèle accorde généralement

des poids nuls aux méthodes deux modèles et Class Transformation, plaçant tout son poids sur la dernière approche. Le poids qui est accordé à celle-ci est généralement autour de 0.92, indiquant que la modélisation directe surestime les lifts réels sur cette simulation. Le poids global inférieur à 1 explique donc l'erreur plus faible malgré qu'un seul modèle sous-jacent soit considéré en réalité. L'algorithme d'optimisation détecte cette tendance et corrige en plaçant des poids légèrement inférieurs à 1.

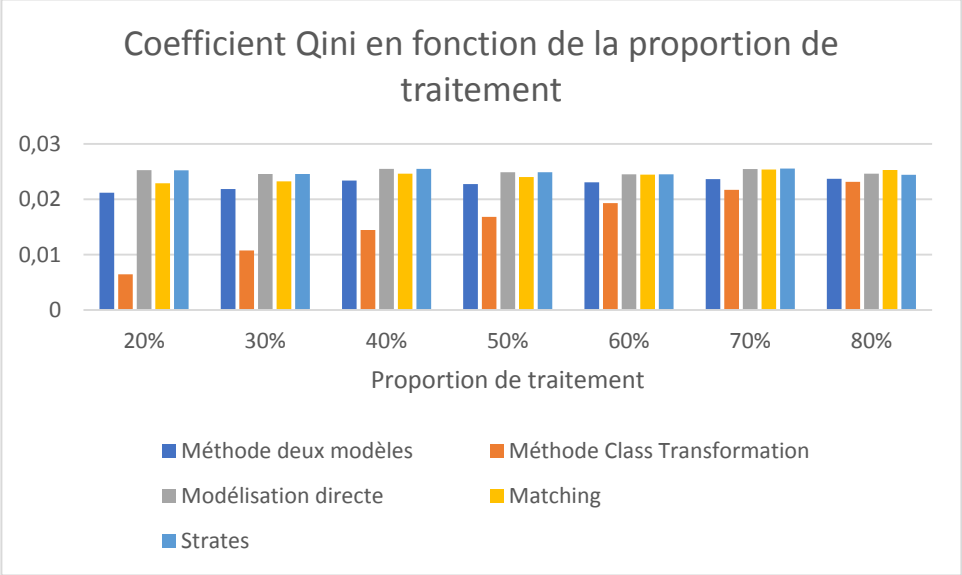

FIGURE 4.11. COEFFICIENT QINI EN FONCTION DE LA PROPORTION DE TRAITEMENT, VARIABLE-CIBLE BINAIRE.

Les mêmes tendances se retrouvent sur le Qini, où les modèles conservent tous leur rang respectif en termes de performance (Figure 4.11). Les tendances sont exactement les mêmes qu'avec l'EQM en fonction de la proportion. Sur ce DGP, le Qini apparaît donc comme une mesure fiable de la performance et refléte de manière acceptable l'erreur. À noter, cependant, qu'étant donné que le Qini est une mesure de rang, l'amélioration de l'erreur due à la sous-pondération dans l'approche par strates ne se reflète pas avec cette mesure : la performance de la méthode par strates est à toutes fins pratiques identique à celle de la modélisation directe, avec un avantage de 0,00003 pour modélisation directe sur la méthode par strates.

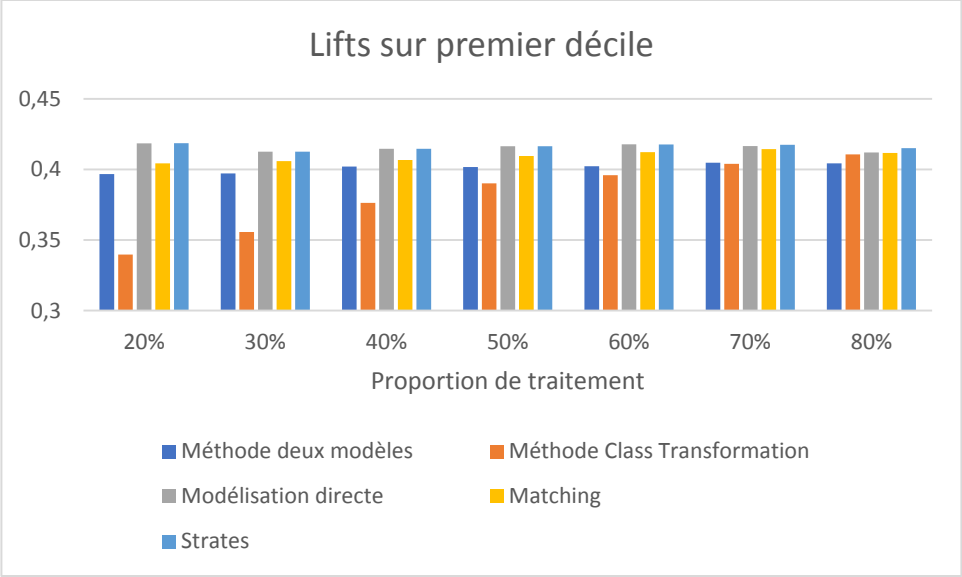

FIGURE 4.12. LIFTS SUR LE PREMIER DÉCILE EN FONCTION DE LA PROPENSION DE TRAITEMENT, VARIABLE-CIBLE BINAIRE.

Sur le premier décile, les lifts observés pour les méthodes de stacking sont très près du meilleur modèle ou encore le dépassent sur toutes les proportions de traitement testées (Figure 4.12). La technique apporte donc une certaine stabilité à ce niveau. La modélisation directe prend la position de tête dans la plupart des cas, mais le stacking vient très près. La méthode deux modèles fait légèrement moins bien, et la méthode Class Transformation suit la tendance observée avec le Qini et le MSE: performance déplorable lorsque la proportion de traitement est faible, puis performe relativement bien pour les proportions élevées.

Dans l'ensemble, le matching n'est pas à son meilleur avec cette méthode de génération de données. La modélisation directe performe extrêmement bien dans ce contexte, et le matching ne parvient pas à améliorer sa performance.

La méthode par strates améliore de manière marginale l'EQM, mais les poids révèlent cependant que ce n'est pas parce qu'une combinaison de modèles améliore la performance, puisque tout le poids est alors mis sur la modélisation directe. L'amélioration marginale de la performance provient alors de l'apport de l'algorithme d'optimisation qui ajuste légèrement le poids du modèle de manière à minimiser l'écart.

La valeur du stacking dans ce contexte n'est donc pas nécessairement significative dans son ensemble dans ce contexte précis.

4.4.2. Poids accordés aux modèles

Sur cette simulation avec la méthode d'appariement, les poids accordés à chaque modèle changent très peu en fonction de la proportion traitement/contrôle lorsque la méthode par matching est employée (Figure 4.13). Le modèle met principalement l'emphase sur l'approche à deux modèles, dont le poids tourne systématiquement autour de 0.6. La méthode de modélisation directe vient ensuite avec des poids entre 0.3 et 0.4, alors que le modèle Class Transformation ne dépasse jamais les 0.15. Cela reflète la performance des modèles sous-jacent, avec plus de poids sur le meilleur et moins de poids sur le moins performant. Une tendance est toutefois observable dans les poids, où le poids de la méthode Class Transformation augmente graduellement avec la proportion de données de traitement jusqu'à 60%, puis diminue ensuite. L'effet inverse est observable dans les courbes des deux autres modèles.

La Figure 4.13 présente les poids moyens de chaque modèle sous-jacent sur les 100 simulations par proportion de traitement. Il permet donc de déceler les variations dans la manière que l'algorithme sélectionne ses combinaisons.

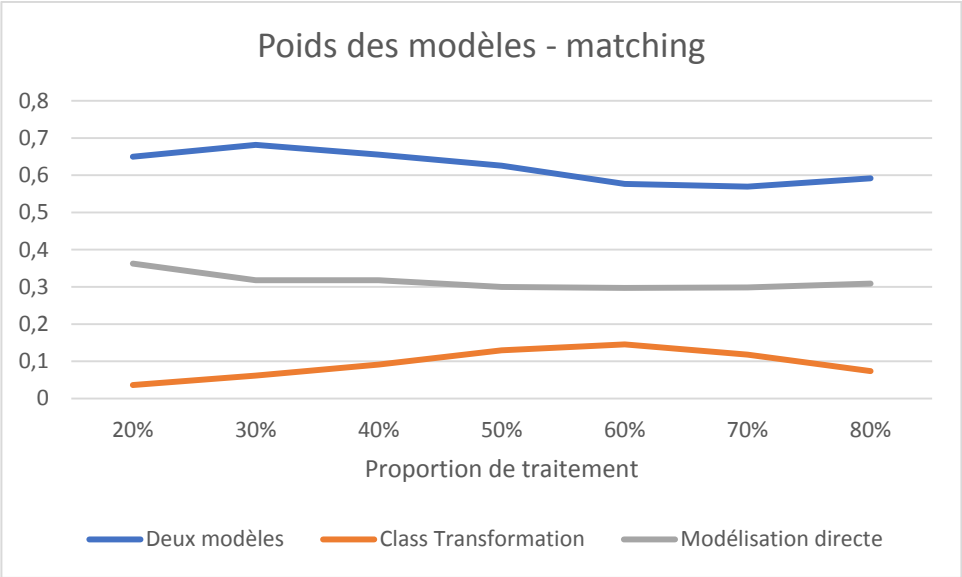

FIGURE 4.13. POIDS ATTRIBUÉS AUX MODÈLES SOUS-JACENTS EN FONCTION DE LA PROPENSION DE TRAITEMENT SELON LA MÉTHODE D'APPARIEMENT, VARIABLE-CIBLE BINAIRE.

La tendance observée est toute autre lorsque la méthode par strates est utilisée pour le stacking (Figure 4.14). Ici, le modèle qui reçoit la majorité du poids est systématiquement la

méthode de modélisation directe. En fait, entre 20 et 60% de données de traitement, le poids accordé aux deux autres modèles est nul et toute l'emphase est mise sur la modélisation directe. Les poids donnés par l'algorithme d'optimisation sont utilisés directement, sans transformation, et il n'y a pas de contrainte indiquant que les poids doivent atteindre 1. Cela fait en sorte que si l'algorithme d'optimisation estime que les modèles sous-jacents sur- ou sous-estiment le lift réel, les poids totaux seront inférieurs ou supérieurs pour donner un ajustement supplémentaire. Après 60%, le poids alloué à celui-ci chute drastiquement pour faire place aux autres modèles, principalement le Class Transformation.

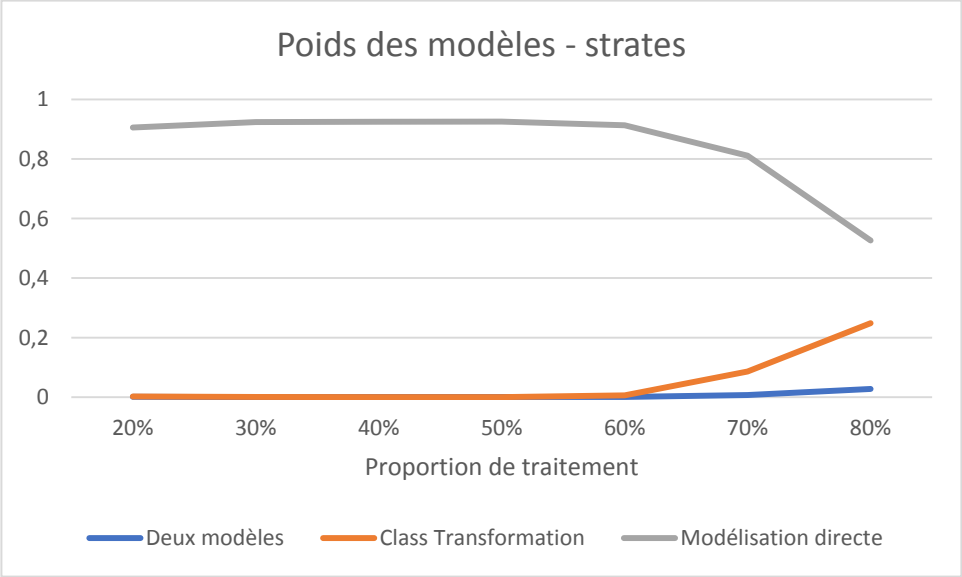

FIGURE 4.14. POIDS ATTRIBUÉS AUX MODÈLES SOUS-JACENTS EN FONCTION DE LA PROPENSION DE TRAITEMENT SELON LA MÉTHODE PAR STRATES, VARIABLE-CIBLE BINAIRE.

4.5. Simulation avec variation de la propension de traitement – continue

4.5.1. Performance

Dans le cas d'une variable-cible continue, la situation est inversée entre les deux approches de stacking: la méthode par matching présente désormais une meilleure performance que la méthode par strates (Tableau 4.2 et Figure 4.15). Dans tous les cas, le matching réduit significativement l'erreur quadratique moyenne par rapport au meilleur modèle sous-jacent, et la méthode par strates en fait autant dans trois des sept scénarios.

TABLEAU 4.2. ERREUR QUADRATIQUE MOYENNE EN FONCTION DE LA PROPENSION DE TRAITEMENT, VARIABLE-CIBLE CONTINUE

	Méthode deux modèles	Méthode Class Transformation	Modélisation directe	Matching	Strates
20%	0,002397	0,175988	0,003502	0,002313	0,002375
30%	0,002005	0,100687	0,002824	0,001938	0,002014
40%	0,001819	0,051684	0,002544	0,001767	0,001908
50%	0,001673	0,022388	0,002409	0,001633	0,001772
60%	0,001558	0,010418	0,002422	0,001536	0,001558
70%	0,001502	0,015088	0,002634	0,001489	0,001493
80%	0,001499	0,036856	0,003154	0,001488	0,001491
Moyenne	0,001769	0,059016	0,002784	0,001738	0,001802

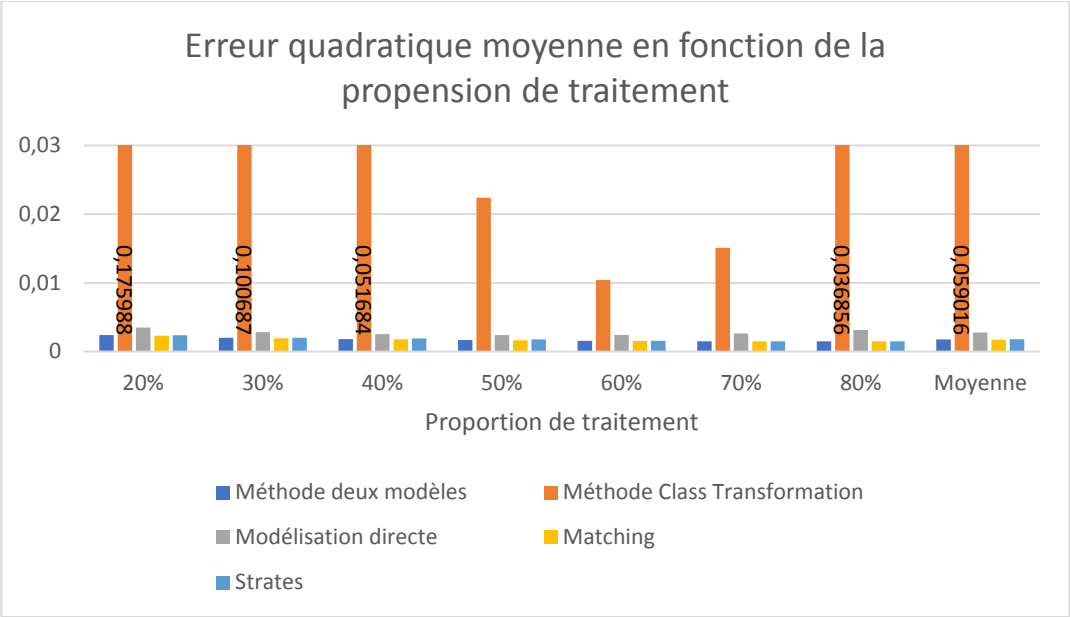

FIGURE 4.15. ERREUR QUADRATIQUE MOYENNE EN FONCTION DE LA PROPENSION DE TRAITEMENT, VARIABLE-CIBLE CONTINUE.

Les modèles sous-jacents performent de manière similaire à ce qu'ils ont fait sous une variable binaire, à l'exception de l'approche deux modèles dont l'erreur diminue drastiquement pour devenir le meilleur modèle, légèrement supérieur à la modélisation directe. Son erreur tend également à diminuer lorsque la proportion de traitement augmente, indiquant vraisemblablement que la méthode est plus efficace pour prédire les observations traitées que celles du groupe contrôle. La Class Transformation commence toujours avec une erreur énorme pour ensuite la voir diminuer lorsque la proportion de traitement est augmentée, atteindre une erreur minimale à 60% de traitement pour ensuite réaugmenter légèrement. La modélisation directe suite la même tendance, performant mieux lorsque les proportions groupes traitement et contrôle sont balancés.

Les deux modèles de stacking réagissent de la même manière que la méthode deux modèles lorsque le pourcentage de traitement varie, diminuant graduellement plus celui-ci est élevé.

Dans l'ensemble, sur ce DGP, les deux approches ont des résultats très similaires, autour de 0.0018 d'erreur quadratique moyenne, mais le matching fait légèrement mieux. Cette performance est comparable avec celle de la méthode deux modèles utilisée seule, toutefois, le matching étant légèrement meilleur et la méthode strates étant légèrement moins performante. Cela va de soi lorsque sont considérés les poids, les deux méthodes

privilégiant fortement la méthode deux modèles sur ce DGP. Il s'agit d'un effet attendu, la méthode deux modèles étant celle avec la meilleure erreur sur ce DGP.

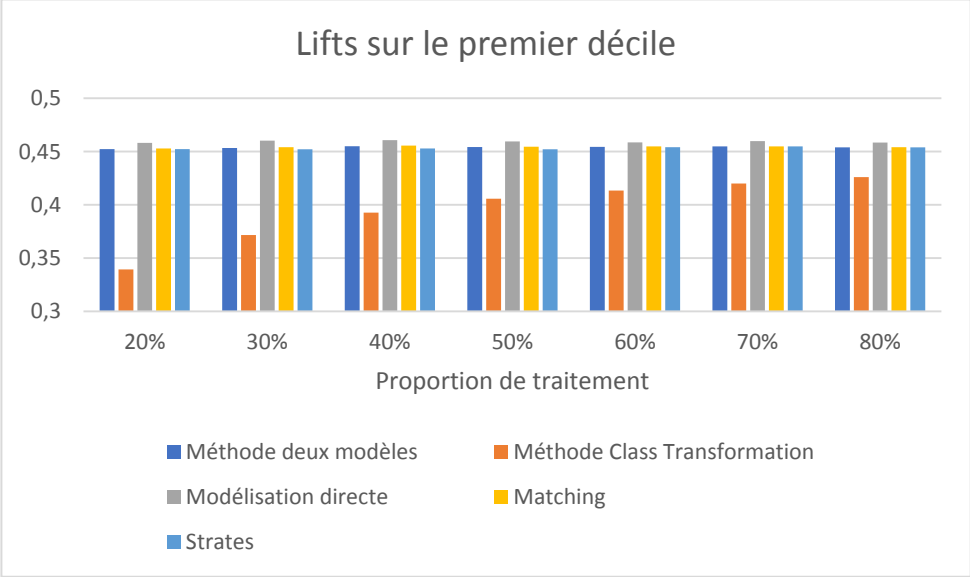

FIGURE 4.16. LIFTS SUR LE PREMIER DÉCILE EN FONCTION DE LA PROPENSION DE TRAITEMENT, VARIABLE-CIBLE CONTINUE.

On retrouve la même tendance du côté du lift sur le premier décile (Figure 4.16). Le stacking par matching et par strates ont tous deux des lifts comparables au meilleur modèle, toutefois le modèle par modélisation directe avec upliftRF obtient des lifts observés légèrement supérieurs sur le premier décile, à 0,5% supplémentaires. La méthode Class Transformation varie significativement sur ce plan, mais tous les autres modèles sont plutôt stables dans leur capacité de maximiser le lift sur le premier décile.

Sur cette méthode de génération de données, de manière générale, le même constat s'impose que sur la méthode précédente. Le stacking par matching apporte une diminution marginale de l'erreur par rapport à la méthode deux modèles qui est le meilleur modèle sous-jacent, alors que la méthode par strates ne parvient pas à l'égaler. La proportion de traitement influence très peu ce constat, tant les poids que la performance relative des modèles étant similaires dans tous les cas évalués.

4.5.2. Poids accordés aux modèles

Contrairement à l'effet observé sur la simulation avec variable-cible binaire, les deux méthodes de stacking sont presque parfaitement en accord sur les poids optimaux à utiliser pour maximiser le gain lorsque la variable est continue. En effet, en utilisant l'une ou l'autre des méthodes, la majorité du poids est donné à la méthode deux modèles, des poids mineurs étant accordés à chacune des deux autres méthodes (Figures 4.17 et 4.18). Dans les deux cas également, on retrouve une surpondération, la somme des poids excédant fréquemment 1.

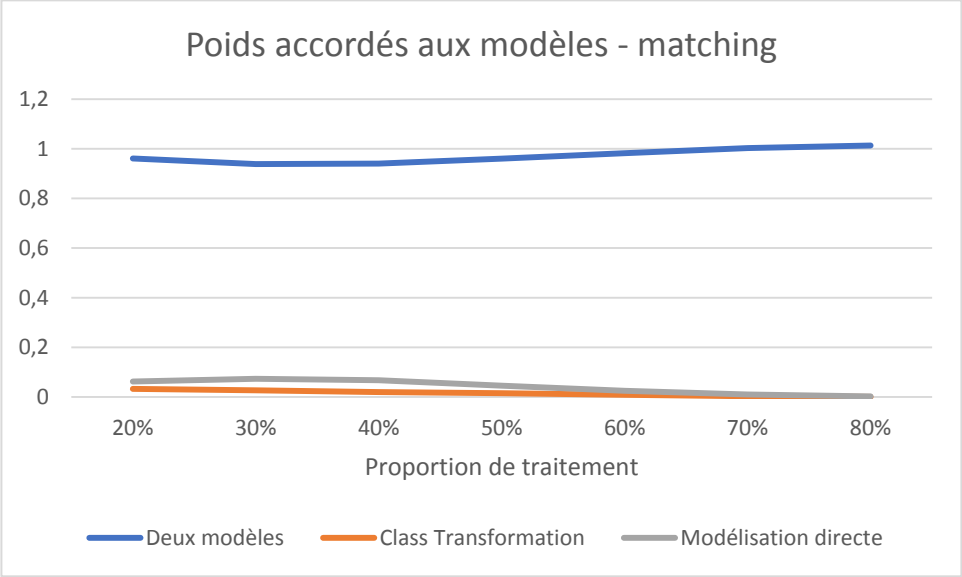

FIGURE 4.17. POIDS DES MODÈLES EN FONCTION DE LA PROPENSION DE TRAITEMENT SELON LA MÉTHODE PAR APPARIEMENT, VARIABLE-CIBLE CONTINUE

On peut toutefois observer que la méthode par strates demeure plus aggressive dans la modification des poids, avec des variations plus rapides lorsque la proportion de traitement change. Elle est également plus encline à fixer certains poids à zéro, alors que la méthode par matching conserve toujours un certain poids sur chaque modèle, aussi minime soit-il.

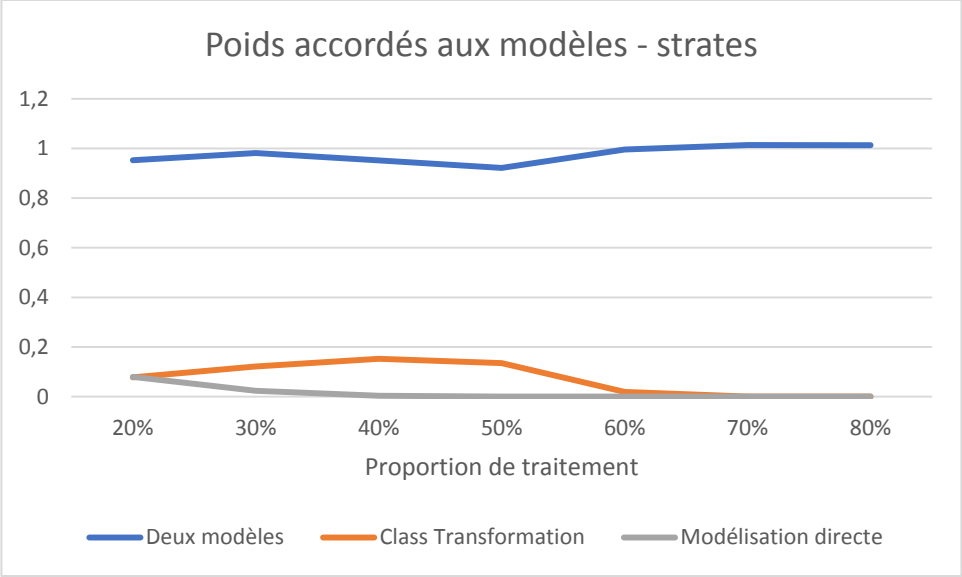

FIGURE 4.18. POIDS DES MODÈLES EN FONCTION DE LA PROPENSION DE TRAITEMENT SELON LA MÉTHODE PAR STRATES, VARIABLE-CIBLE CONTINUE.

4.6. Résumé des résultats

Dans l'ensemble, les résultats des simulations démontrent un avantage au stacking par rapport à tous les modèles individuels dans la plupart des simulations étudiées. Dans certains cas, cependant, l'un ou l'autre des modèles individuels demeure supérieur au stacking.

Les deux méthodes de stacking, par strates et par matching, ont un avantage l'une par rapport à l'autre dans certaines situations. Il est donc impossible de dire, sur la base de ces simulations, si une méthode ou une autre est supérieure de manière définitive.

Cependant, la valeur du stacking lui-même dans le contexte de modèles Uplift est bien démontrée dans les résultats des simulations.

Il reste toutefois à découvrir si cet avantage se transpose dans des situations de données réelles. C'est l'objectif des deux chapitres qui suivront.

5. Présentation du cas

Les données utilisées pour le cas d'application réelle de cette étude ont été gracieusement fournies pour les fins de cette étude par une entreprise canadienne qui a cependant refusé d'être identifiée. En vertu d'une entente de confidentialité avec l'entreprise et afin de protéger les données de ses clients, ni le contenu, ni les libellés des variables contenues dans le modèle ne peuvent être révélés. Seules les mesures de performance finales de même que des informations sommaires sur le jeu de données peuvent donc être publiées.

Le jeu de données complet comporte trois campagnes publicitaires distinctes portant sur des produits connexes. Les campagnes ont été complétées entre Janvier 2017 et Juin 2018. Combinées, ces trois campagnes totalisent 165 950 observations avec 330 variables explicatives pour chacune d'entre elles. Il est possible de les considérer de manière distincte afin de tirer avantage des différences subtiles entre un produit ou un autre, ou encore de ne les considérer que comme une seule vaste campagne.

Pour les fins de ce mémoire, il a été choisi de combiner toutes les données et de les considérer comme une seule et même campagne. La raison est simple et principalement motivée par une considération technique. En effet, étant donné le faible nombre de réponses positives, particulièrement au sein du groupe contrôle, certains modèles sous-jacents s'avéraient relativement instables lorsque la quantité de données est plus faible. La quantité accrue de données permet d'améliorer la stabilité et la fiabilité de chacun des modèles pour les fins d'analyse et de comparaison de la performance entre les modèles.

5.1. Préparation des données

La constitution du jeu de données a nécessité la mise en commun de trois sources de données différentes. L'une contient les informations connues sur chaque client et constitue la base de données explicatives par mois; la seconde contribue la liste de clients qui ont été contactés dans le cadre de la campagne ainsi que la liste des clients du groupe contrôle. La troisième source présente les résultats associés à chaque individu contacté dans le cadre de la campagne publicitaire.

La constitution du jeu de données final a requis le traitement des données d'abord sur une base mensuelle avant de les combiner. Cette approche était nécessaire pour plusieurs raisons :

- Le dossier-client utilisé doit correspondre au mois précédant celui où il a été contacté
- La variable-cible correspond à la détention du produit à la fin du mois suivant le contact
- Certaines variables explicatives ont été calculées sous forme de variations entre les mois.
- Certains clients ont été contactés plusieurs fois pendant une même campagne, et d'autres ont été contactés dans le cadre de plus d'une des trois campagnes. Les règles d'affaires de l'entreprise font en sorte de placer un client à qui on a présenté une offre publicitaire sur une liste noire pendant deux mois suivant le contact. Après cette période, un même client peut être recontacté.

Le fichier de campagne est ensuite fractionné en de multiples tables différentes contenant chacune les clients contactés pour un mois donné.

La constitution du jeu de données par mois consiste alors en une jointure de trois tables: la liste de campagne constituée au préalable, les variables explicatives pour le mois précédant le contact, et la détention du produit pour le mois suivant le contact, puisque celle-ci a été définie comme variable-cible.

La prochaine étape consiste à combiner à nouveau les données en une seule table pour toute la campagne, en prenant soin de n'inclure que la première fois qu'un même client a été contacté. Concrètement, cette étape a été réalisée par itérations à partir de Janvier 2017, le premier mois de la campagne. Une table temporaire a été créée contenant tous les clients contactés en Février qui n'apparaissent pas en Janvier, puis ses observations ajoutées à celles de Janvier. Le processus a été répété de la même manière pour tous les mois de la campagne de manière à obtenir une liste complète de tous les clients contactés, chaque client n'apparaissant qu'une seule fois.

Certaines variables avaient des données manquantes en très petite quantité, soit moins de 500 observations problématiques et, dans l'ensemble, moins de 1500 observations avaient des valeurs manquantes pour l'une ou l'autre des variables présentes dans le fichier. Étant donné leur petit nombre, elles ont été retirées. Pour plusieurs des valeurs manquantes, il s'agissait par ailleurs d'une situation où le client avait quitté l'entreprise, de telle sorte qu'il ne se trouvait pas dans les données des mois subséquents.

Dans l'ensemble, les données obtenues étaient très propres. Au-delà de ces valeurs manquantes, très peu de manipulations ont été nécessaires avant de les utiliser. À la fin de toutes les manipulations, le jeu de données contenait 106 338 observations dont les réponses étaient comme suit :

TABLEAU 5.1. DÉCOMPOSITION DU JEU DE DONNÉES SELON L'ASSIGNATION DE TRAITEMENT ET LA RÉPONSE

	Traitement	Contrôle
0	92 194	10 515
1	3 510	119
Total	95 704	10 634
Taux de réponse positive	3,67%	1,12%

En soustrayant le taux de réponse du groupe contrôle de celui du groupe traitement, on note alors un effet moyen du traitement de 2,55%. Il sera utile de connaître cette mesure afin de comparer l'effet du traitement sur le premier décile identifié avec l'effet moyen, donnant ainsi une mesure de la capacité du modèle à maximiser l'effet du traitement.

5.2. Sélection de variables et élaboration des modèles

Dans un premier temps, une sélection manuelle de variables a été effectuée. Cette première étape était nécessaire car certaines variables avaient des valeurs entièrement homogènes sur l'ensemble du jeu de données et d'autres n'avaient aucune signification ni aucune utilité dans le contexte de la problématique d'affaires étudiée. D'autres étaient par leur nature très corrélées et similaires en signification, et une sélection préliminaire a également été effectuée à ce niveau. De plus, l'inclusion de l'ensemble des 330 variables explicatives aurait

nécessité un temps de calcul impossible à réaliser en pratique, en plus d'être grandement inefficace d'un point de vue prédictif. Cette première sélection a permis de réduire à 95 le nombre de variables explicatives.

Par la suite, la fonction niv, pour Net Information Value, du package Uplift (Guelman, 2014), a été utilisée sur l'ensemble du jeu de données pour donner une vision d'ensemble et une estimation préliminaire de l'importance de chaque variable. Celle-ci est une version du critère de valeur informationnelle fréquemment utilisé en modélisation, modifiée pour être applicable dans le contexte Uplift. L'implantation utilisée est incluse dans le package Uplift (Guelman, 2014), et calcule le critère sur des échantillons bootstrap. Ensuite, pour chaque modèle, les variables ont été insérées une par une en fonction de leur valeur informationnelle nette ajustée, validant la performance à l'aide du Qini pour chaque modèle entraîné. Lorsque la performance est améliorée, on conserve la variable, si elle est réduite, on la retire et on répète avec la variable suivante.

Cette méthode a été répétée quatre fois, de manière à sélectionner des variables explicatives optimales pour chaque méthode (deux modèles sur traitement et contrôle, Class Transformation et modélisation directe). Les modèles sous-jacents ont donc été élaborés indépendamment, avec variables explicatives différentes, de manière à maximiser la performance de chacun sur la base du Qini.

À terme, le modèle combiné par stacking inclut 18 variables explicatives, réparties comme indiqué dans le Tableau 5.2 :

TABLEAU 5.2. RÉPARTITION DES VARIABLES EN FONCTION DE LEUR NOMBRE D'APPARITIONS DANS DES MODÈLES DIFFÉRENTS

Nombre d'apparitions dans un modèle	Variables
4	2

3	2
2	7
1	7

Les variables les plus efficaces pour expliquer sont immédiatement apparentes, puisque deux des 18 variables apparaissent dans tous les modèles, et deux autres apparaissent dans 3/4. Sept autres variables sont présentes dans deux modèles, alors que les sept restantes n'apparaissent que dans un seul modèle.

Il est donc évident qu'une grande portion de la valeur explicative de chaque modèle provient des mêmes variables. Cependant, une petite portion de celle-ci diffère d'un modèle à l'autre, rendant plutôt intéressante la combinaison de ces modèles puisque chacun capture certains aspects légèrement différents de la variance totale.

6. Résultats du cas pratique

6.1. Poids accordés par les modèles de stacking

Parmi les six modèles de second niveau élaborés, seuls deux seront présentés dans cette section et la section suivante sur la performance. En effet, les modèles avec optimisation sans contrainte et par régression se sont avérés nettement moins intéressants au niveau des résultats, et étaient donc moins pertinents à exposer formellement et à analyser.

Il s'avère effectivement préférable dans tous les cas d'utiliser un algorithme d'optimisation avec une contrainte de non-négativité, puisqu'il n'y a aucune situation dans laquelle la contrainte détériore les résultats, alors que dans les situations où l'optimum implique l'utilisation de poids négatifs, la performance du modèle ne se réplique pas du tout sur l'ensemble de validation. Ces résultats abondent donc dans le même sens que Breiman (1996) sur la nécessité d'utiliser des contraintes de non-négativité dans un contexte de stacking.

Par souci de transparence, les résultats de tous les modèles se trouvent tout de même dans les tableaux à l'annexe 2.

Le Tableau 6.1 présente donc les poids accordés par les deux modèles finaux, par matching et par strates, tous deux selon la méthode d'optimisation avec contrainte de non-négativité.

TABLEAU 6.1. POIDS DES MODÈLES SOUS-JACENTS SELON LES DEUX MÉTHODES DE STACKING DANS LE CONTEXTE DE L'ÉTUDE DE CAS

Méthode	Poids deux modèles	Poids Class Transformation	Poids modélisation directe (upliftRF)
Matching+optimisation avec contrainte de non-négativité	0,2164	0,0955	0,709
Strates+optimisation avec contrainte de non-négativité	0,269	0,000	0,689

Il y a une certaine similitude dans les poids qui sont attribués par les différents modèles, notamment au niveau du poids élevé accordé à la modélisation directe par upliftRF. En effet, les deux approches donnent un poids élevé à la modélisation directe (0,709 et 0,689), puis une valeur moindre pour l'approche à deux modèles (0,2164 et 0,269). Une différence notable se situe au niveau de la méthode Class Transformation, puisque la méthode par strates lui donne un poids entièrement nul, alors que le matching lui donne un poids de 0,116.

C'est précisément dans ce type de cas que la contrainte de non-négativité prenait toute son importance, puisque le poids de ce dernier modèle aurait autrement été négatif, ce qui aurait réduit la performance globale du modèle.

6.2. Performance

La Figure 6.1 présente la courbe des gains utilisée pour calculer le Qini, mais présentée sous forme de diagramme en bâton permettant de comparer la performance relative des différents modèles par décile. Le 10e décile a été retiré, puisqu'il implique de cibler l'ensemble des individus; la performance y est donc identique pour tous les modèles.

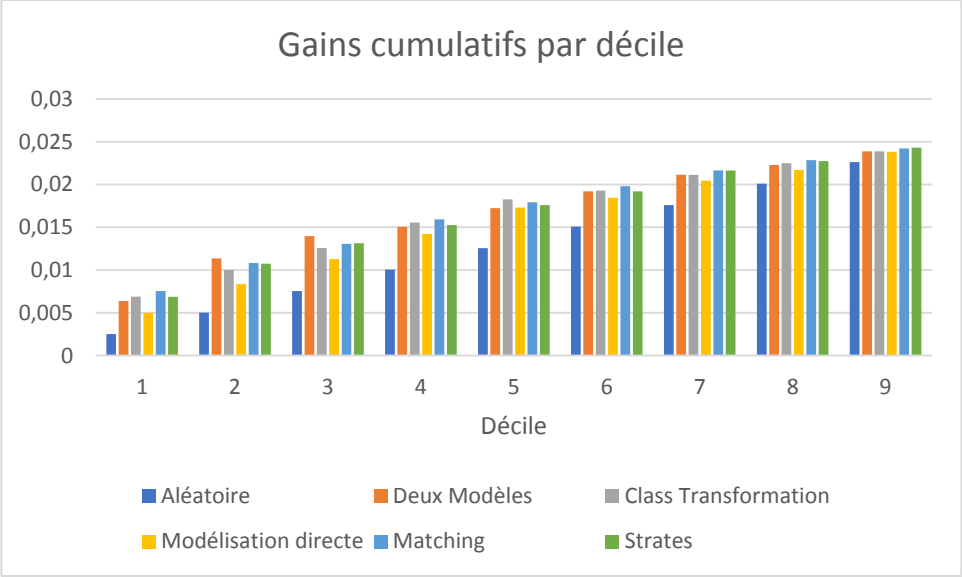

FIGURE 6.1. GAINS CUMULATIFS EN FONCTION DU DÉCILE DANS LE CONTEXTE DE L'ÉTUDE DE CAS.

On peut remarquer que les modèles de stacking se distinguent par leurs gains cumulatifs plus élevés que les autres sur la quasi-totalité des déciles. Les gains représentent la proportion supplémentaire de y=1 capturée dans chaque décile. Des gains accrus dans les premiers déciles sont donc souhaitables puisqu'ils indiquent que le modèle capture mieux les clients que le traitement affectera positivement. Une explication plus détaillée se trouve dans la section 2.2.4.

Au niveau des modèles individuels, il est possible de remarquer que ce n'est pas le même modèle qui est le meilleur sur tous les déciles. En effet, sur le premier décile, c'est la Class Transformation qui est le meilleur modèle, puis la méthode deux modèles prend le dessus pour quelques déciles avant de retomber sous le premier. Il est intéressant de noter que bien que la méthode de modélisation directe performe moins bien de manière générale, le poids qui y est associé par stacking est élevé. Le modèle apporte donc une valeur informationnelle différente des deux autres que le stacking est en mesure de capturer en dépit de sa performance individuelle moins intéressante. Chaque modèle de premier niveau semble donc avoir ses forces et ses faiblesses distinctes.

Une manière de valider cette assertion est de générer le même graphique des gains, en utilisant les gains marginaux plutôt que ceux cumulatifs.

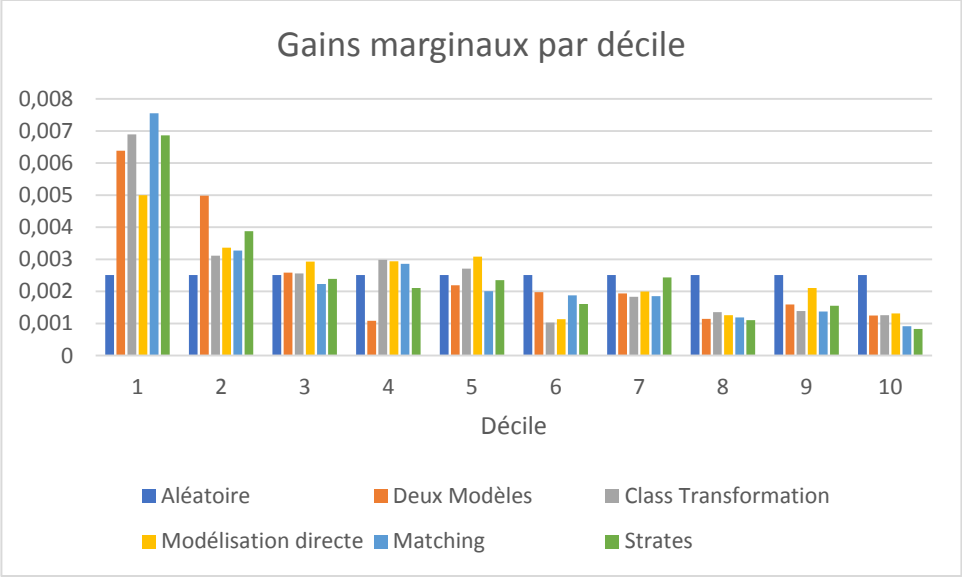

FIGURE 6.2. GAINS MARGINAUX PAR DÉCILE DANS LE CONTEXTE DE L'ÉTUDE DE CAS.

En observant la Figure 6.2, qui présente les gains non cumulatifs, on peut remarquer que la majorité des gains des modèles de stacking proviennent du premier décile, où le gain est supérieur à celui de tous les modèles sous-jacents. Le matching parvient à capturer un gain supplémentaire sur le premier décile par rapport à la méthode par strates, qui semble placer ces observations supplémentaires dans le second décile. Une autre portion des gains provient de la stabilité accrue par la combinaison. Par exemple, pour le 4e décile, la performance de la méthode deux modèles chute drastiquement après avoir bien fait sur les premiers déciles, mais les deux autres modèles font mieux; la combinaison par stacking fait en sorte que ces deux facteurs s'opposent et une performance respectable en résulte pour les modèles combinés.

Le tableau 6.2 présente le coefficient Qini global de chaque modèle, leur lift respectif sur le premier décile ainsi qu'une valeur représentant les gains relatifs sur le premier décile par rapport à une sélection aléatoire.

TABLEAU 6.2. PRINCIPALES MESURES DE PERFORMANCE POUR L'ÉTUDE DE CAS

Modèle	Qini	Lift sur premier	Lift sur premier

		décile	décile / aléatoire
Deux modèles	0,00355	6,49%	2,22x
Class Transformation	0,00348	7,10%	2,43x
Modélisation directe (upliftRF)	0,00263	5,12%	1,75x
Matching	0,00382	7,70%	2,64x
Strates	0,00362	6,94%	2,38x

Il est possible de remarquer au niveau de la performance avec le Qini que les méthodes deux modèles et Class Transformation sont très près autour de 0,0035, alors que la modélisation directe vient derrière à 0,00263. Sur le premier décile, cependant, Class Transformation a un net avantage à 7,1%.

Les modèles de stacking sont prometteurs sur ce jeu de données. En effet, tous améliorent significativement le Qini en comparaison avec les modèles sous-jacents utilisés individuellement, le matching augmentant celui-ci de 8% par rapport à l'approche deux modèles, le plus performant des trois modèles sous-jacents, alors que la méthode par strates avec contraintes de non-négativité l'augmente de 2%.

Cette amélioration se reflète également sur le premier décile, le modèle par strates étant légèrement inférieur au meilleur modèle (Class Transformation) mais grandement supérieur aux résultats des deux autres modèles, représentant 2,38 fois le lift d'une sélection aléatoire. La méthode par matching est pour sa part le meilleur modèle sur la base de cette mesure; son lift sur le premier décile de 7,70% représente 2,64 fois le lift aléatoire.

7. Discussion

Les résultats collectés démontrent sans aucun doute qu'il y a une valeur à employer une technique de stacking dans le contexte de la modélisation Uplift et ce, tant dans un environnement de simulation contrôlé que dans un contexte d'affaires réel utilisant des données marketing collectées par une entreprise. Les deux approches étudiées présentent chacune certains avantages et inconvénients et leur performance relative varie en fonction du contexte.

La méthode par matching apporte une grande stabilité dans la performance que l'on ne retrouve pas dans les autres modèles, ce qui est un avantage significatif dans le contexte où l'un des problèmes fondamentaux de la modélisation Uplift réside dans son instabilité par rapport à de nouvelles données. Dans les simulations, le modèle de matching termine quasi-systématiquement en deuxième position parmi tous les modèles étudiés, tant sur la base du Qini que de l'erreur quadratique moyenne, et permet de stabiliser les résultats lorsque les données changent en modifiant les poids en fonction de la performance relative des modèles sous-jacents. Sur le jeu de données réelles étudié, la méthode par matching performe mieux que toutes les autres approches utilisées individuellement.

La méthode par strates est également intéressante pour une toute autre raison. En effet, sur les simulations, cette méthode parvient à avoir la plus petite erreur quadratique moyenne par rapport au lift réel dans une grande majorité des cas. Cette méthode semble donc estimer le lift de la manière la plus fiable. Toutefois, ces gains ne se reflètent pas toujours sur le Qini, celui-ci étant une mesure d'ordre, et le modèle semble avoir une tendance à placer du poids seulement sur le meilleur modèle sur certains jeux de données.

Dans certains contextes, cette méthode s'apparente donc plutôt à une sélection de modèles qu'à une combinaison de modèles proprement dite. Puisque l'erreur quadratique moyenne ne peut pas être calculée lorsque le lift réel n'est pas connu et que le Qini ne suit pas toujours la même tendance, il est impossible de dire si cet avantage sur l'EQM se transpose lorsque le modèle est élaboré sur un jeu de données réel.

Par ailleurs, il semble que la méthode par strates fonctionne mieux que la méthode par matching lorsque la variable-cible est binaire. L'inverse est vrai lorsque la variable-cible est continue. Les deux méthodes de stacking sont significativement plus performantes que les modèles sous-jacents dans le contexte d'une variable-cible continue, le gain de performance à utiliser le stacking étant nettement supérieur à celui observé lorsque la variable-cible est binaire. C'est le cas sur les deux méthodes de génération de données utilisées.

De même, les simulations avec variation de la taille de l'effet Uplift ont démontré que la méthode par matching tend à mieux performer lorsque l'effet Uplift est moins significatif, alors que la méthode par strates est souvent supérieure lorsque celui-ci est élevé. Ce constat parvient possiblement à fournir une autre explication à la bonne performance du matching sur les données réelles d'entreprise, puisque l'effet total du traitement était très faible dans le cadre de l'étude, à 2,4% d'effet Uplift moyen.

Il y a donc plusieurs éléments de contexte, dont le type de variable-cible, l'effet moyen du traitement sur le jeu de données à analyser et l'objectif final visé par l'analyse, qui détermineront la méthode à privilégier. Au fil des entraînements, il est devenu apparent que l'une et l'autre des méthodes est parfois plus performante que l'autre en fonction du contexte. Il est donc impossible de déterminer de manière catégorique si l'une ou l'autre est plus performante dans l'ensemble. Les raisons de ces variations sont nébuleuses et il faudrait des recherches plus approfondies afin de déterminer les facteurs qui influencent la performance relative des deux méthodes.

Heureusement, il n'est pas nécessaire de choisir. En effet, les deux méthodes utilisent les mêmes modèles sous-jacents et il est ainsi possible de n'entraîner ceux-ci qu'une seule fois pour élaborer les deux modèles de stacking. Les deux méthodes d'optimisation nécessitent ensuite un temps de calcul négligeable (de l'ordre de quelques secondes tout au plus sur un ordinateur relativement récent en 2018), rendant possible et souhaitable d'entraîner les deux et de sélectionner ensuite en fonction des résultats obtenus selon les données, les modèles spécifiques et les objectifs visés par l'analyse.

La recherche a également permis de constater qu'il est nécessaire d'employer une contrainte de non-négativité pour que les résultats soient fiables et constants. Si cette nécessité n'est démontrée que dans le cas de la méthode par strates, il n'y a néanmoins aucun coût de performance ni de temps de calcul à utiliser une telle contrainte et les

résultats sont toujours équivalents ou supérieurs. Les algorithmes d'optimisation sans contrainte et de régression linéaire devraient donc être abandonnés puisque la performance du modèle avec contrainte est systématiquement supérieure lorsque l'un des poids serait négatif, ou égale lorsque tous les poids optimaux sont positifs dans tous les cas.

Même dans le cas du matching où ce problème n'a pas été rencontré, les performances sont totalement similaires et il est suffisant de ne générer qu'un seul modèle de second niveau parmi ceux testés. Les modèles de régression et d'optimisation sans contrainte ne fonctionnent simplement pas à la hauteur du modèle avec contrainte de non-négativité.

Dans les tests effectués, les approches de stacking sont toutes deux parvenues à tirer avantage des particularités de chaque modèle sous-jacent et à moduler les poids en fonction de la performance de chaque modèle à chaque itération. Ainsi, lorsqu'un jeu de données particulier donne du fil à retordre à l'un ou l'autre des modèles, la performance globale ne souffre pas autant puisque l'algorithme de second niveau réduit alors le poids du modèle qui performe mal.

Une différence notable existe entre la performance du stacking sur les données simulées et sur les données réelles testées dans le cadre de ce mémoire, puisque le Qini n'est que peu ou pas amélioré sur les données simulées alors qu'un gain plus substantiel a été observé sur les données réelles.

Plusieurs explications sont possibles pour ce phénomène. D'une part, il n'y avait pas de sélection de variables sur les simulations; toutes les variables étant incluses, il est possible que les modèles expliquent la même variabilité et que les gains étaient donc minimes ou inexistants. Toutefois, dans le contexte réel, il était impossible d'inclure toutes les variables, tant pour des raisons de performance du modèle que pour des considérations liées au temps de calcul, et une même variable n'a pas le même impact sur des modèles différents. Concrètement, à performance égale, chaque modèle explique un aspect différent du jeu de données, et il est alors logique qu'une combinaison linéaire optimisée permette d'aller chercher une portion des avantages de chacun pour obtenir une prédiction globale améliorée.

Ce phénomène est également apparent lorsque l'on observe la performance par décile, puisque les trois modèles sont chacun plus performants sur des déciles différents, et les

méthodes combinatoires améliorent la performance sur chacun des déciles. Il est possible, donc, qu'il soit souhaitable de rechercher des modèles très différents plutôt qu'une performance globale supérieure pour les modèles sous-jacents. Cette possibilité avait été suggérée par Günes (2017) et semble se refléter dans ces résultats. Il faudrait cependant étudier un plus grand nombre de modèles sur plusieurs jeux de données différents pour conclure définitivement.

Bien que la performance globale soit améliorée avec le stacking, celui-ci a certains désavantages qui rendent plus difficile son application dans la réalité, notamment des contraintes de temps d'élaboration et de temps de calcul. La nécessité d'élaborer quatre modèles différents, soit deux pour l'approche deux modèles, un pour la Class Transformation et un pour la méthode directe, consomme un temps très élevé; par définition, environ quatre fois le travail nécessaire pour préparer un seul modèle. À cela, il faut également ajouter le modèle d'estimation du score de propensité qui est un préalable à la méthode Class Transformation lorsque l'assignation de traitement n'est pas aléatoire, portant à cinq le nombre total de modèles élaborés.

Ce temps d'élaboration n'est qu'une facette de la problématique temporelle liée à l'utilisation du stacking; le temps de calcul en est une autre. La nécessité d'utiliser la validation croisée pour obtenir les poids multiplie autant de fois le temps de calcul nécessaire. À titre d'exemple, une seule itération du modèle complet sur le jeu de données pratique présenté dans ce mémoire requiert environ huit heures de calcul.

Bien que la performance soit adéquate malgré le fait que l'on cherche à identifier une classe rare dans le cas du jeu de données réelles, il peut être souhaitable de réaliser un sous-échantillonnage des observations où y=0. Cette méthode a été testée dans le cadre de l'étude de cas et a résulté en une performance similaire tout en réduisant le temps de calcul d'un facteur de 10. Ces résultats n'ont pas été inclus dans ces pages puisque cet échantillonnage introduit un certain biais dans les données, mais en pratique l'approche peut être intéressante à explorer.

Le stacking n'est donc pas une solution miracle qui conviendra à toutes les utilisations. Le gain de performance peut être intéressant dans certains contextes, mais l'investissement nécessaire pour y arriver est significatif et il est difficile voire impossible de savoir au préalable si les gains seront significatifs ou plutôt négligeables. La performance du modèle

combiné est également dépendant, par définition, de la capacité des modèles sous-jacents à capturer le signal Uplift.

Un autre problème potentiel provient du fait que le modèle combiné s'apparente beaucoup à une boîte noire; il devient difficile d'expliquer comment les variables explicatives influencent réellement le lift. L'algorithme LIME (Linear Interpretable Model-Agnostic Explanations, proposé par Marco Tulio Ribeiro, Sameer Singh et Carlos Guestrin (2016) et recréé pour R par Thomas Lin Pedersen (2018)), qui explique chaque observation à l'aide d'un modèle linéaire simple afin de donner un aperçu de la manière dont les variables influencent celles-ci, a été exploré pour fournir une solution à ce problème. Conceptuellement, cette approche fonctionne après avoir implanté la méthode permettant à LIME de lancer les prédictions en utilisant l'algorithme et est en mesure de donner une idée grossière de l'effet de chaque variable sur l'uplift (ces résultats sont toutefois eux aussi exclus en raison de l'entente de confidentialité avec l'entreprise qui a fourni les données). Cependant, il est nécessaire au préalable de réduire le jeu de données que l'on tente d'expliquer aux seules variables utilisées dans le modèle combiné, et le temps de calcul et la capacité de mémoire vive sont significatifs. Là encore, l'approche est imparfaite et pourrait être améliorée.

L'implantation de modèles Uplift en entreprise requiert encore un changement culturel au sein des organisations qui souhaitent utiliser la technique. En effet, puisque la variable optimisée n'est pas la réponse elle-même, les taux de réponse bruts sont généralement plus faibles qu'en utilisant des modèles réponse traditionnels. Il faut donc adopter un point de vue plus global et intégrer la notion de coût d'acquisition dans l'évaluation de la performance pour reconnaître les pleins avantages de la modélisation Uplift. Aucune solution à cette problématique n'a été suggérée dans la littérature existante.

La problématique centrale de la modélisation Uplift, soit le fait que le lift réel est inconnu, est étudiée de manière indirecte dans le cadre de ce mémoire. En effet, si la performance d'estimateurs du lift n'est pas directement évaluée, l'utilisation d'estimateurs différents sous-tend les deux approches évaluées et la difficulté d'obtenir un estimateur fiable a été un problème récurrent dans l'exploration des approches possibles pour le stacking, puisque aucune variable-cible n'était immédiatement disponible pour alimenter le modèle de

second niveau et la variable transformée de la méthode Class Transformation ne parvenait pas à estimer de manière fiable le lift pour cette utilisation.

Dans l'ensemble, donc, les deux méthodes présentées dans cet ouvrage parviennent à améliorer la performance des modèles Uplift, leur performance étant au mieux nettement supérieures aux modèles existant, au pire en deuxième position derrière le modèle individuel le plus performant au cours des tests. Il est donc clair qu'il y a une valeur ajoutée à cette méthode, mais il est crucial d'évaluer au préalable si le gain de performance apportera des bénéfices suffisants pour justifier la complexité d'implantation et de calcul supplémentaire.

8. Conclusion

La présente recherche a permis de démontrer qu'il est effectivement possible de combiner les trois méthodes existantes de modélisation Uplift en utilisant le stacking pour améliorer la performance. Deux approches différentes ont été étudiées pour parvenir à cette fin, l'une utilisant le matching pour pairer les observations de traitement et contrôle, l'autre utilisant une estimation préliminaire pour grouper les observations et utiliser des moyennes par groupe pour calibrer les poids des modèles sous-jacents.

La technique a d'abord été testée sur des données simulées en utilisant deux processus de génération de données différents en faisant varier plusieurs paramètres afin de valider comment se comporte la technique dans des contextes variés. Dans le cadre des simulations, la méthode a également été testée dans une situation de variable-cible binaire et continue.

Par la suite, la méthode a été utilisée sur une campagne marketing réelle fournie par une entreprise afin de vérifier si la méthode fonctionne également dans un contexte d'affaires réel où les données ne sont pas contrôlées.

Les résultats démontrent que l'une et l'autre des techniques de stacking se comparent avantageusement avec les modèles sous-jacents, leur coefficient Qini se situant systématiquement devant au moins deux des trois techniques existantes. Les deux méthodes étudiées ont chacune des avantages différents et sont préférables dans des contextes différents, la méthode par matching étant nettement plus fiable que tous les modèles existants alors que la méthode par strates conserve la volatilité des modèles existants, mais donne fréquemment des prédictions plus précises. Les résultats se sont non seulement confirmés sur le jeu de données réel, mais la performance relative du stacking est significativement meilleure sur celui-ci que sur les données simulées. Il est également possible, et souhaitable, d'entraîner les deux méthodes en même temps afin de déterminer laquelle est la meilleure pour un contexte donné; le temps de calcul incrémental est minime puisque l'étape la plus coûteuse est l'entraînement des modèles sous-jacents par validation croisée, réalisé de manière identique sur les deux méthodes et donc réutilisable pour les deux approches.

La présente recherche est la première à étudier le stacking pour des modèles Uplift. La démonstration que cette utilisation conjointe des deux techniques peut avoir un impact positif sur la performance constitue donc un premier apport à la littérature Uplift.

Les deux approches présentées sont elles aussi inédites et forment une autre contribution et ce, de plusieurs manières. Prises globalement, les approches par matching et par strates peuvent être utilisées telles quelles pour modéliser des problèmes Uplift et forment un apport substantiel pour faire avancer la recherche de cette problématique.

L'idée qui sous-tend chacune des deux approches constitue par ailleurs un apport distinct à la recherche Uplift, puisque les deux estimateurs du lift utilisés pour calibrer les poids des modèles n'avaient jamais été testés dans la littérature existante. La quête centrale du domaine de la modélisation Uplift étant de trouver un estimateur fiable du lift réel puisque celui-ci est impossible à observer, ce mémoire apporte donc deux réponses potentielles à cette question.

Le stacking n'est toutefois pas une panacée pour la modélisation Uplift. La technique requiert toujours d'élaborer quatre modèles prédictifs différents (cinq si l'on inclut le modèle potentiellement utilisé pour évaluer le score de propension) pour servir de prédicteurs pour le modèle de second niveau, ce qui représente un travail considérable. Le temps de calcul est également significatif étant donnée la nécessité d'employer la validation croisée pour obtenir les poids à accorder à chaque modèle.

En plus d'étudier pour la première fois le stacking dans le contexte Uplift, ce mémoire contribue également deux nouveaux estimateurs potentiels qui n'ont pas été étudiés par le passé dans la littérature, correspondant respectivement à la méthode par matching et la méthode par strates. La bonne performance du matching semble indiquer que lorsque le nombre d'observations est suffisamment grand, les lifts individuels obtenus par matching pourraient être un bon estimateur, et la même supposition est vraie du lift moyen observé par la méthode par strates. Il serait alors possible en théorie d'utiliser ces deux variables comme cibles pour de nouvelles approches individuelles de modélisation Uplift dans la même veine que la méthode Class Transformation. Bien que l'étude de cette possibilité se situe au-delà du sujet de ce mémoire, il ouvre néanmoins ces nouvelles avenues de recherche potentielles.

D’autre part, seules des forêts aléatoires ont été considérées dans le cadre de cette recherche. L’éventail de modèles Uplift existants étant beaucoup plus large, une avenue de recherche pourrait être d’explorer le stacking avec d’autres algorithmes pour les modèles sous-jacents. Par ailleurs, le modèle de second niveau pourrait également être différent; seules des méthodes de combinaison linéaires ont été considérées, et un nombre plus grand d’algorithmes de second niveau pourraient potentiellement améliorer encore la performance de la technique.

Bibliographie

Ardia, D., Mullen, K., Peterson, B., Ulrich, J., & Boudt, K. (2016, Décembre 19). *Package 'DEoptim' - Global Optimization by Differential Evolution.* Récupéré sur Comprehensive R Archive Network: https://cran.r-project.org/web/packages/DEoptim/DEoptim.pdf

Athey, S., & Imbens, G. W. (2015). Machine Learning Methods for Estimating Heterogeneous Causal Effects. *stat, 1050*(5).

Austin, P. C. (2011, Juin 8). An Introduction to Propensity Score Methods for Reducing the Effects of Confounding in Observational Studies. *Multivariate Behavioral Research, 46(3)*, 399-424.

Breiman, L. (1996). Stacked Regressions. *Machine Learning, 24*(1), pp. 49-64.

Breiman, L., Cutler, A., Liaw, A., & Wiener, M. (2018, Mars 22). *Package 'randomForest'.* Récupéré sur Comprehensive R Archive Network: https://cran.r-project.org/web/packages/randomForest/randomForest.pdf

Günes, F. (2017, May 18). *Why do stacked ensemble models win data science competitions?* Récupéré sur SAS Institute Blog: Subconscious Musings: https://blogs.sas.com/content/subconsciousmusings/2017/05/18/stacked-ensemble-models-win-data-science-competitions/

Guelman, L. (2014, March 17). *Package 'uplift'.* Récupéré sur Comprehensive R Archive Network: https://cran.r-project.org/web/packages/uplift/uplift.pdf

Gutierrez, P. (2016, Mars 3). *Introduction to Uplift Modelling.* Récupéré sur LinkedIn SlideShare: https://www.slideshare.net/PierreGutierrez2/introduction-to-uplift-modelling

Gutierrez, P., & Gerardy, J.-Y. (2016). *Causal Inference and Uplift Modelling - A review of the literature.* International Conference on Predictive Applications and APIs.

Himmetoglu, B. (2017, Février). *Stacking models for Improved Predictions*. Récupéré sur KDnuggets: https://www.kdnuggets.com/2017/02/stacking-models-imropved-predictions.html

Imai, K., Ho, D., King, G., Stuart, E., & Whitworth, A. (2018, Janvier 9). *Package 'MatchIt'.* Récupéré sur Comprehensive R Archive Network: https://cran.r-project.org/web/packages/MatchIt/MatchIt.pdf

Jaskowski, M., & Jaroszwicz, S. (2012). *Uplift Modeling for clinical trial data.* Edinburgh: ICML 2012 Workship on Clinical Data Analysis.

Kane, K., Lo, V. S., & Zheng, J. (2014). Mining for the truly responsive consumers and prospects using true-lift modeling: Comparison of new and existing methods. *Journal of Marketing Analytics, 2*(4), 218-238.

Larocque, D. (2018). Advanced Data Mining (80-619-11). Montréal, Québec, Canada.

Leblanc, M., & Tibshirani, R. (1994). Combining Estimates in Regression and Classification. *Journal of the American Statistical Association, 91*(436), 1641-1650.

Lo, V. S. (2002). The True Lift Model - A Novel Data Mining Approach to Response Modeling in Database Marketing. *SIGKDD Explorations, 4*(2), pp. 78-86.

Mesalles Naranjo, O. (2012). *Testing a New Metric for Uplift Models.* Edinburgh: University of Edinburgh School of Mathematics.

Parfait, G. (2018). *Subset a dataframe excluding rows in another dataframe.* Récupéré sur StackOverflow: https://stackoverflow.com/questions/50991856/subset-a-dataframe-excluding-rows-in-another-dataframe

Pedersen, T. L. (2018, Mars 8). *Package 'lime':Local interpretable model-agnostic explanations.* Récupéré sur Comprehensive R Archive Network: https://cran.r-project.org/web/packages/lime/lime.pdf

R Core Team. (2018). R: A language and environment for statistical computing. Vienna, Austria: R Foundation for Statistical Computing.

Radcliffe, N. J. (2007). Using Control Groups to Target on Predicted Lift: Building and Assessing Uplift Models. *1*, 14-21.

Radcliffe, N. J. (2012). *Moment of Uplift.* Edinburgh.

Ribeiro, M. T., Singh, S., & Guestrin, C. (2016, Août 12). *Introduction to Local Interpretable Model-Agnostic Explanations (LIME).* Récupéré sur O'Reilly: https://www.oreilly.com/learning/introduction-to-local-interpretable-model-agnostic-explanations-lime

Rzepakowski, P., & Jaroszewicz, S. (2010, Décembre). Decision Trees for Uplift Modeling. *Data Mining (ICDM)*, pp. 441-450.

Sekhon, J. S. (2018, Mai 4). *Package 'Matching'.* Récupéré sur Comprehensive R Archive Network: https://cran.r-project.org/web/packages/Matching/Matching.pdf

Siegel, E. (2013, Janvier 23). *The Real Story Behind Obama's Election Victory.* Récupéré sur The Fiscal Times: http://www.thefiscaltimes.com/Articles/2013/01/21/The-Real-Story-Behind-Obamas-Election-Victory#page1

Soltys, M., Jaroszewicz, S., & Rzepakowski, P. (2014). Ensemble Methods for Uplift Modeling. *Data Mining and Knowledge Discovery, 29*(6), 1531-1559.

Storn, R., & Price, K. (1997). Differential Evolution - A Simple and Efficient Heuristic for Global Optimization over Continuous Spaces. *Journal of Global Optimization, 11*(4), 341-359.

Surry, P. D., & Radcliffe, N. J. (2011). *Real-world uplift modelling with significance- based uplift trees.* Stochastic Solutions.

Tian, L., Alizadeh, A. A., Gentiles, A. J., & Tibshirani, R. (2014). A Simple Method for Estimating Interactions between a Treatment and a Large Number of Covariates. *Journal of the American Statistical Association, 109*(508), 1517-1532.

Tibshirani, J., Athey, S., Wager, S., Friedberg, R., Miner, L., & Wright, M. (2018, Mai 9). *Package 'grf'.* Récupéré sur Comprehensive R Archive Network: https://cran.r-project.org/web/packages/grf/grf.pdf

Ting, K. M., & Witten, I. H. (1997). *Stacked Generalization: when does it work?* University of Waikato.

van der Laan, M. J., Polley, E. C., & Hubbard, A. E. (2007). Super Learner. *Statistical Applications in Genetics and Molecular Biology, 6*(1), p. 21.

Wolpert, D. H. (1992). Stacked Generalization. *5*(2), pp. 241-259.

Yan, Y. (2016, Mai 13). *Package 'MLmetrics - Machine Learning Evaluation Metrics.* Récupéré sur Comprehensive R Archive Network: https://cran.r-project.org/web/packages/MLmetrics/MLmetrics.pdf

Zhao, Y., Fang, X., & Simchi-Levi, D. (2017). Uplift Modeling with Multiple Treatments and General Response. *Proceedings of the 2017 SIAM International Conference on Data Mining* (pp. 588-596). Society for Industrial and Applied Mathematics.

Annexe 1: Tableaux de résultats sur données simulées

Simulation du package Uplift – binaire

Les tableaux suivants présentent les mesures de performance obtenues à la suite des simulations effectuées à l'aide de la fonction sim_pte du package Uplift. Les trois algorithmes de stacking testés sont inclus pour les deux méthodes élaborées (matching et strates), de même que la performance des trois modèles sous-jacents. La mesure présentée par le tableau est indiquée dans la toute première cellule en haut à gauche de chaque tableau. Pour ce jeu de données, cela correspond à l'erreur quadratique moyenne (MSE), au Qini et au Lift moyen sur le premier décile.

MSE	Méthode deux modèles	Méthode Class Transformation	Modélisation directe	Matching + optim	Matching + DEoptim	Matching + régression	Strates + optim	Strates + DEoptim	Strates + régression
1	0,035718	0,048383	0,034419	0,029839	0,029838	0,029838	0,04374	0,036954	0,04388
2	0,029249	0,050726	0,025027	0,021512	0,021514	0,021514	0,027962	0,024883	0,028095
3	0,026938	0,05331	0,020156	0,018595	0,018602	0,018602	0,023165	0,020047	0,023295
4	0,026601	0,055761	0,018853	0,018217	0,018225	0,018225	0,022169	0,018016	0,02237
5	0,026533	0,05733	0,018522	0,018022	0,018025	0,018025	0,022081	0,017827	0,022273
6	0,02684	0,058581	0,018578	0,018223	0,018227	0,018227	0,023581	0,017738	0,023802
7	0,026914	0,059315	0,018658	0,018342	0,018345	0,018345	0,022436	0,017491	0,022657
8	0,027217	0,059955	0,018738	0,018402	0,018401	0,018401	0,020968	0,017341	0,021155
9	0,027301	0,060397	0,018872	0,018434	0,018431	0,018431	0,023831	0,017374	0,024029
10	0,027453	0,060728	0,018966	0,018595	0,018594	0,018594	0,02212	0,0175	0,022358

Qini	Méthode deux modèles	Méthode Class Transformation	Modélisation directe	Matching + optim	Matching + DEoptim	Matching + régression	Strates + optim	Strates + DEoptim	Strates + régression
1	0,053753	0,047757	0,057201	0,058055	0,058047	0,058047	0,051245	0,054208	0,051194
2	0,089346	0,081614	0,091114	0,091689	0,09169	0,09169	0,089389	0,90395	0,089333
3	0,103629	0,096127	0,105656	0,105548	0,105538	0,105538	0,104092	0,105003	0,104061
4	0,110502	0,103173	0,112782	0,112315	0,112311	0,112311	0,111106	0,112232	0,111046
5	0,113875	0,106856	0,116299	0,115772	0,115766	0,115766	0,114572	0,115625	0,114507
6	0,115939	0,109006	0,118253	0,117815	0,117815	0,117815	0,116193	0,117649	0,116137
7	0,117215	0,110327	0,119683	0,119119	0,119122	0,119122	0,117746	0,119045	0,117687
8	0,118071	0,111162	0,120483	0,119904	0,119903	0,119905	0,119021	0,119981	0,118985
9	0,118747	0,111972	0,12107	0,120589	0,120598	0,120598	0,118922	0,120588	0,118867
10	0,119093	0,112237	0,121469	0,120877	0,120873	0,120873	0,119817	0,120948	0,119742

Lift 1er décile	Méthode deux modèles	Méthode Class Transformation	Modélisation directe	Matching + optim	Matching + DEoptim	Matching + régression	Strates + optim	Strates + DEoptim	Strates + régression
1	0,385361	0,297347	0,359008	0,412816	0,412835	0,412835	0,352027	0,375675	0,351879
2	0,56627	0,496068	0,557445	0,582132	0,582132	0,582132	0,564596	0,573177	0,563971
3	0,625372	0,576456	0,643146	0,642605	0,642583	0,642583	0,632703	0,64162	0,632608
4	0,654408	0,613138	0,678655	0,673371	0,67329	0,67329	0,663187	0,674251	0,663048
5	0,668403	0,63377	0,698866	0,689206	0,689237	0,689237	0,681707	0,691343	0,681409
6	0,677457	0,643766	0,708753	0,699785	0,699751	0,699751	0,689731	0,700814	0,689292
7	0,685422	0,648993	0,718824	0,707767	0,707727	0,707727	0,698335	0,708728	0,697684
8	0,687926	0,655819	0,722317	0,712335	0,712451	0,712452	0,707676	0,714817	0,707201
9	0,691033	0,658429	0,726663	0,715583	0,715589	0,715589	0,706662	0,717453	0,705967
10	0,692433	0,65961	0,7282	0,71854	0,718507	0,718507	0,712774	0,72201	0,711881

Simulation du package Uplift – continue

Les tableaux suivants présentent les mesures de performance obtenues à la suite des simulations effectuées à l'aide de la fonction sim_pte du package Uplift, telle que modifiée par Denis Larocque pour obtenir une variable-cible continue. Les trois algorithmes de stacking testés sont inclus pour les deux méthodes élaborées (matching et strates), de même que la performance des trois modèles sous-jacents. La mesure présentée par le tableau est indiquée dans la toute première cellule en haut à gauche de chaque tableau. Pour ce jeu de données, cela correspond à l'erreur quadratique moyenne (MSE) et au Lift moyen sur le premier décile, les fonctions d'évaluation de la performance utilisées pour mesurer le Qini (soit celles du package Uplift) étant incompables avec le cas d'une variable continue.

MSE	Méthode deux modèles	Méthode Class Transformation	Modélisation directe	Matching + optim	Matching + DEoptim	Matching + régression	Strates + optim	Strates + DEoptim	Strates + régression
1	0,848044	1,147544	0,654772	0,4378	0,437801	0,437801	0,900806	0,569351	0,902704
2	0,605425	0,978109	0,608605	0,383652	0,383629	0,383641	0,537064	0,416237	0,537496
3	0,47207	0,955192	0,598841	0,362424	0,360617	0,362444	0,465881	0,363574	0,466555
4	0,424644	0,948548	0,593428	0,348582	0,343795	0,348613	0,409807	0,344689	0,41022
5	0,400809	0,946138	0,590812	0,340355	0,333761	0,340398	0,40426	0,333385	0,404822
6	0,387874	0,944917	0,589452	0,336601	0,328199	0,336676	0,375016	0,325385	0,375427
7	0,378775	0,945605	0,588278	0,333546	0,324363	0,333622	0,368423	0,322242	0,368888
8	0,37312	0,944623	0,587804	0,331621	0,32243	0,331712	0,385456	0,325575	0,386084
9	0,369433	0,944836	0,586757	0,33069	0,321341	0,330777	0,36842	0,318566	0,368741
10	0,36669	0,945233	0,586463	0,329793	0,319728	0,329878	0,357976	0,316014	0,357976

Lift 1er décile	Méthode deux modèles	Méthode Class Transformation	Modélisation directe	Matching + optim	Matching + DEoptim	Matching + régression	Strates + optim	Strates + DEoptim	Strates + régression
1	3,161255	3,045596	3,312553	3,313421	3,313421	3,313421	3,107162	3,246866	3,106705
2	3,278879	3,249795	3,319436	3,331222	3,331274	3,331226	3,260891	3,314184	3,260441
3	3,322796	3,284731	3,321164	3,340208	3,341289	3,34015	3,290596	3,328469	3,290042
4	3,341741	3,293773	3,321761	3,351547	3,353616	3,35154	3,315558	3,342955	3,315743
5	3,34854	3,300852	3,320193	3,355543	3,360228	3,355674	3,308696	3,3339698	3,308388
6	3,361453	3,299325	3,318957	3,359455	3,362766	3,359351	3,330754	3,352167	3,330332
7	3,35604	3,302406	3,318653	3,359657	3,363865	3,359739	3,329843	3,346304	3,329285
8	3,359455	3,303324	3,317766	3,355998	3,359584	3,355946	3,321817	3,343987	3,32157
9	3,359727	3,299633	3,317132	3,35922	3,361931	3,359252	3,330422	3,347443	3,32972
10	3,358875	3,302606	3,316543	3,362759	3,365807	3,362832	3,329827	3,345983	3,329046

Modèle avec variation de la proportion traitement/contrôle – binaire

Les tableaux suivants présentent les mesures de performance obtenues à la suite des simulations effectuées à l'aide de la fonction de génération de données fournie par mon directeur de mémoire, Denis Larocque. Les trois algorithmes de stacking testés sont inclus pour les deux méthodes élaborées (matching et strates), de même que la performance des trois modèles sous-jacents. Pour ce jeu de données, les mesures sont l'erreur quadratique moyenne (MSE), le Qini et le Lift moyen sur le premier décile.

Qini	Méthode deux modèles	Méthode Class Transformation	Modélisation directe	Matching + optim	Matching + DEoptim	Matching + régression	Strates + optim	Strates + DEoptim	Strates + régression
20%	0,021173	0,006463	0,02527	0,022902	0,022894	0,022876	0,019122	0,025235	0,018495
30%	0,021863	0,010764	0,02457	0,023262	0,02325	0,023251	0,020394	0,024572	0,0187
40%	0,023391	0,014448	0,025491	0,024649	0,024634	0,024632	0,022394	0,025482	0,02085
50%	0,022754	0,016824	0,024889	0,024047	0,024027	0,024032	0,019686	0,024892	0,019369
60%	0,023061	0,019312	0,024506	0,024464	0,024466	0,024466	0,018478	0,024499	0,018414
70%	0,023641	0,021708	0,025463	0,025369	0,025371	0,025366	0,02092	0,025553	0,020835
80%	0,023687	0,023164	0,024624	0,025258	0,025292	0,025256	0,020107	0,024414	0,018917

MSE	Méthode deux modèles	Méthode Class Transformation	Modélisation directe	Matching + optim	Matching + DEoptim	Matching + régression	Strates + optim	Strates + DEoptim	Strates + régression
20%	0,015034	0,185595	0,008592	0,011351	0,011421	0,011376	0,017854	0,007928	0,019686
30%	0,014579	0,110354	0,008801	0,011843	0,011884	0,011863	0,014343	0,008348	0,018677
40%	0,014009	0,060114	0,009104	0,011415	0,011432	0,011433	0,012077	0,008637	0,015622
50%	0,014012	0,030096	0,009444	0,011216	0,011236	0,011235	0,01737	0,008802	0,018411
60%	0,014063	0,017788	0,009706	0,010606	0,01062	0,01062	0,017786	0,008963	0,01795
70%	0,014293	0,020797	0,009539	0,009971	0,009982	0,009984	0,015388	0,008482	0,015642
80%	0,015171	0,041274	0,010607	0,010473	0,010421	0,010483	0,015232	0,008897	0,01893

Lift sur 1er décile	Méthode deux modèles	Méthode Class Transformation	Modélisation directe	Matching + optim	Matching + DEoptim	Matching + régression	Strates + optim	Strates + DEoptim	Strates + régression
20%	0,396747	0,339672	0,418495	0,404184	0,404287	0,404066	0,382586	0,418666	0,378678
30%	0,397164	0,355663	0,412639	0,406009	0,40587	0,405925	0,38856	0,412638	0,381746
40%	0,402077	0,376362	0,414645	0,406759	0,40676	0,406789	0,397557	0,414611	0,390042
50%	0,401692	0,390139	0,4165	0,409762	0,409588	0,409599	0,384703	0,416429	0,382844
60%	0,402252	0,395901	0,417798	0,412446	0,412288	0,412288	0,382697	0,41767	0,382219
70%	0,404832	0,403956	0,416559	0,414504	0,414446	0,414534	0,398189	0,417528	0,397095
80%	0,404273	0,410716	0,411989	0,412023	0,411664	0,411871	0,395713	0,415094	0,391789

Modèle avec variation de la proportion traitement/contrôle - continue

Les tableaux suivants présentent les mesures de performance obtenues à la suite des simulations effectuées à l'aide de la fonction de génération de données fournie par mon directeur de mémoire, Denis Larocque, dans le cas d'une variable-cible continue. Les trois algorithmes de stacking testés sont inclus pour les deux méthodes élaborées (matching et strates), de même que la performance des trois modèles sous-jacents. La mesure présentée par le tableau est indiquée dans la toute première cellule en haut à gauche de chaque tableau. Pour ce jeu de données, cela correspond à l'erreur quadratique moyenne (MSE) et au Lift moyen sur le premier décile, la fonction utilisée pour évaluer le Qini (celle du package Uplift) étant incompatible avec le cas d'une variable-cible continue.

MSE	Méthode deux modèles	Méthode Class Transformation	Modélisation directe	Matching + optim	Matching + DEoptim	Matching + régression	Strates + optim	Strates + DEoptim	Strates + régression
20%	0,02397	0,175988	0,003502	0,002312	0,002313	0,002315	0,00248	0,002375	0,003282
30%	0,002005	0,100687	0,002824	0,001936	0,001938	0,001939	0,02095	0,002014	0,003066
40%	0,001819	0,051684	0,002544	0,001769	0,001767	0,00177	0,002029	0,001908	0,003433
50%	0,001673	0,022388	0,002409	0,001635	0,001633	0,001637	0,00263	0,001772	0,003266
60%	0,001558	0,010418	0,002422	0,001539	0,001536	0,001546	0,003156	0,001558	0,003202
70%	0,001502	0,015088	0,002634	0,001495	0,001489	0,001507	0,002777	0,001493	0,002782
80%	0,001499	0,036856	0,003154	0,001511	0,001488	0,001515	0,00276	0,001491	0,00277

Lift sur 1er décile	Méthode deux modèles	Méthode Class Transformation	Modélisation directe	Matching + optim	Matching + DEoptim	Matching + régression	Strates + optim	Strates + DEoptim	Strates + régression
20%	0,452247	0,339329	0,458104	0,45296	0,452883	0,45279	0,452222	0,452219	0,44476
30%	0,453324	0,371648	0,46017	0,45414	0,454094	0,453998	0,452506	0,452056	0,444337
40%	0,454978	0,392586	0,460608	0,455478	0,455573	0,455432	0,452001	0,452786	0,441398
50%	0,454189	0,405654	0,459439	0,454401	0,454433	0,454346	0,445789	0,452146	0,441988
60%	0,454363	0,413414	0,458598	0,454631	0,454786	0,454428	0,45345	0,454034	0,44327
70%	0,454773	0,419952	0,459774	0,454841	0,454812	0,454588	0,445975	0,454773	0,445978
80%	0,453958	0,425965	0,458469	0,453289	0,45399	0,453279	0,445512	0,453958	0,445498

Annexe 2: Résultats complets du cas

Cette annexe contient les mesures de performance observées dans le contexte de l'étude de cas sur des données de production d'une entreprise. Dans un second temps, on retrouve également les poids qui ont été attribués par les six méthodes de stacking (3 algorithmes chacun pour les approches par matching et strates) aux trois modèles sous-jacents élaborés.

Modèle	Qini	Lift sur premier décile	Lift sur premier décile / aléatoire
Deux modèles	0.00355	6,49%	2,22x
Class Transformation	0.00348	7,10%	2,43x
Modélisation directe (upliftRF)	0.00263	5,12%	1,75x
Matching avec optimisation sans contrainte	0.00380	7,67%	2,63x
Matching avec optimisation et contrainte de non-négativité	0.00382	7,70%	2,64x
Matching avec régression linéaire	0.00382	7,70%	2,64x
Strates avec optimisation sans contrainte	0.00307	5,64%	1,93x
Strates avec optimisation et contrainte de non-négativité	0.00362	6,94%	2,38x
Strates avec régression linéaire	0.00307	5,64%	1,93x

Poids

Méthode	Poids deux modèles	Poids Class Transformation	Poids modélisation directe (upliftRF)
Matching+optimisation sans contrainte	0,2188	0,0977	0,695
Matching+optimisation avec contrainte de non-négativité	0,2164	0,0955	0,709
Matching+régression	0,2164	0,0955	0,709
Strates+optimisation sans contrainte	0,626	-0,326	0,647
Strates+optimisation avec contrainte de non-négativité	0,269	0,0000	0,689
Strates+régression	0,627	-0,326	0,646

Annexe 3: Tableaux de poids des modèles sous-jacents

Les tableaux suivants dans cet annexe présentent les poids moyens des trois modèles sous-jacents pour les 100 simulations roulées pour chaque valeur de chaque paramètre.

À l'instar des résultats finaux présentés dans le texte, seuls les poids du meilleur algoritme, soit l'optimisation avec contrainte de non négativité, a été présenté pour chaque méthode (appariement et strates)

1. Variation de la propension de traitement - continue

Matching

	Deux modèles	Class Transformation	Modélisation directe
20%	0,961019	0,032188	0,061901
30%	0,938535	0,02661	0,072819
40%	0,940385	0,019471	0,06713
50%	0,960319	0,014767	0,045463
60%	0,982482	0,008491	0,024556
70%	1,002876	0,002767	0,009974
80%	1,013516	0,000242	0,002063

Strates

	Deux modèles	Class Transformation	Modélisation directe
20%	0,952754	0,077254	0,079671
30%	0,981996	0,121703	0,023272
40%	0,951916	0,152654	0,003161
50%	0,922092	0,135087	0,000000
60%	0,995869	0,019042	0,000000

70%	1,013834	0,000000	0,000000
80%	1,0133	0,000000	0,000000

2. Variation de la propension de traitement - Binaire

Matching

	Deux modèles	Class Transformation	Modélisation directe
20%	0,649697	0,036102	0,362423
30%	0,681687	0,0614	0,317647
40%	0,655561	0,091388	0,317826
50%	0,625872	0,129367	0,299814
60%	0,576756	0,145362	0,297115
70%	0,569322	0,117902	0,29831
80%	0,591758	0,073541	0,308726

Strates

	Deux modèles	Class Transformation	Modélisation directe
20%	0,001053	0,002282	0,906014
30%	0,000000	0,00038	0,924045
40%	0,000843	0,000000	0,925067
50%	0,00038	0,000000	0,925666
60%	0,000498	0,005991	0,913515

70%	0,007042	0,086032	0,811016
80%	0,027323	0,2484	0,526085

3. Variation de la taille d'effet Uplift - Binaire

Matching

	Deux modèles	Class Transformation	Modélisation directe
1	0,591366	0,205238	0,395566
2	0,657794	0,185755	0,364672
3	0,617683	0,180544	0,383367
4	0,585254	0,179743	0,401744
5	0,524618	0,189382	0,445253
6	0,502201	0,190165	0,461798
7	0,478352	0,195232	0,478352
8	0,453922	0,198702	0,498203
9	0,439113	0,20184	0,510225
10	0,436155	0,204357	0,510395

Strates

	Deux modèles	Class Transformation	Modélisation directe
1	0,614082	0,480474	0,117894
2	0,553959	0,328378	0,384713
3	0,364326	0,260053	0,620578
4	0,296789	0,195304	0,733503
5	0,296036	0,157377	0,760807
6	0,24712	0,150693	0,813989
7	0,266389	0,146858	0,796971
8	0,224926	0,135275	0,845516
9	0,197385	0,154809	0,859534
10	0,196953	0,138424	0,871846

4. Variation de la taille d'effet Uplift - Continue

Appariement

	Deux modèles	Class Transformation	Modélisation directe
1	0,446341	0,349094	0,513442
2	0,528179	0,57229	0,207281
3	0,587659	0,628413	0,055839
4	0,631772	0,602249	0,027046
5	0,652551	0,579329	0,016837
6	0,668988	0,562368	0,009219
7	0,675363	0,554588	0,006623
8	0,678737	0,551021	0,003579
9	0,679378	0,546147	0,00556
10	0,687509	0,535095	0,0043

Strates

	Deux modèles	Class Transformation	Modélisation directe
1	0,268653	0,575816	0,594446
2	0,346131	0,703953	0,418601
3	0,346568	0,684181	0,420722
4	0,333488	0,784172	0,350791
5	0,379806	0,696957	0,366517
6	0,403269	0,632927	0,390818
7	0,389092	0,700664	0,351095
8	0,369216	0,687407	0,384055
9	0,398739	0,676562	0,357679
10	0,43902	0,638481	0,34137