Mémoire présenté le :

pour l'obtention du Diplôme Universitaire d'actuariat de l'ISFA
et l'admission à l'Institut des Actuaires

Par : Camille Bachelet

Titre : Création d'une variable tarifaire d'interactions en assurance automobile par machine learning

Confidentialité : ☐ NON ☒ (Durée : ☒ 1 an ☐ 2 ans)
Les signataires s'engagent à respecter la confidentialité indiquée ci-dessus

Membres présents du jury de l'Institut des Actuaires *Signature*

. . . . . .

. . . . . .

. . . . . .

Membres présents du jury de l'ISFA

. . . . . .

. . . . . .

. . . . . .

Entreprise :
Nom : Generali France

Signature :

Directeur de mémoire en entreprise :
Nom : MICHEL Rémi

Signature :

Invité :
Nom :

Signature :

Autorisation de publication et de mise en ligne sur un site de diffusion de documents actuariels *(après expiration de l'éventuel délai de confidentialité)*

Signature du responsable entreprise

Signature du candidat

Résumé

Dans un contexte de prix toujours plus compétitifs, les assureurs automobiles se doivent de construire des tarifs de plus en plus attractifs pour attirer de nouveaux clients. Pour cela, il est nécessaire de rechercher plus de performances dans les modèles de sinistres des garanties proposées. Cela se fait notamment par la sophistication de ces modèles en y intégrant des méthodes innovantes. Dans cette optique, ce mémoire utilise certains algorithmes de machine learning pour diminuer à plusieurs niveaux les erreurs des modèles tarifaires automobiles, tout en conservant le caractère interprétable des modèles linéaires généralisés, structure classique de tarification.

Pour cela, nous partons des modèles actuels en fréquence et coût moyen, dont nous complétons l'information externe pour perfectionner le zonier. Des machine learning sont ensuite appliqués pour stabiliser les résidus du modèle, avant de leur appliquer un arbre choisi par bootstrap. Les branches de cet arbre constituent les modalités de la variable d'interaction cible. Nous évaluons finalement dans quelle mesure cette nouvelle variable fait gagner en performance lorsqu'elle est ajoutée aux modèles.

Cette démarche est testée sur les garanties bris de glace et dommages tous accidents, qui ont chacune des spécificités de modélisation. Elle donne des résultats très mitigés sur ces deux garanties, en bris de glace où elle n'améliore que très peu la fréquence, tandis que le gain sur la dommage tous accidents est également faible et situé uniquement au niveau du coût moyen.

Mots clés : assurance automobile, variable d'interaction, machine learning, arbre CART, régression pénalisée, random forest, gradient boosting, régression elasticnet, modèles linéaires généralisés

Abstract

As prices in the car insurance industry are getting more and more competitive, car insurers must propose increasingly attractive rates to attract new customers. As such, it is necessary to seek higher performances in proposed guarantees for loss models. Those designs can be sophisticated by incorporating innovative methods. Keeping that in mind, this thesis uses machine learning as a way to reduce errors in the car insurance's pricing models at several levels, while retaining the interpretable character of generalized linear models, as a classic pricing structure.

For this, we start from the current models for frequency and average cost of claims, to which we supplement the external information to perfect the ratig by geographic areas. Machine learning is then applied to stabilize the residuals of the model, before applying to them a regression tree chosen by bootstrap. The branches of this tree constitute the modalities of the target interaction variable. Finally, we evaluate to what extent this new variable improves performance when added to the models.

This approach is tested on breakage and all accident damage guarantees, which have specific modeling features. The results shows very minor improvements in models : in windscreen cover it only slightly improves frequency performance, while the gain on damage from all accidents is bearly visible on the average cost of claims.

key words : car insurance, interaction variable, machine learning, Decision tree learning, penalized regression, random forest, gradient boosting, elasticnet regression, generalized linear model

Remerciements

Je souhaite tout d'abord adresser mes remerciements à l'ISFA et son équipe d'enseignement, pour m'avoir permis d'acquérir les bases de l'assurance non-vie et en Data Science. Ces connaissances m'ont permis d'aborder mon alternance et mon mémoire dans les meilleures conditions possibles. J'ai pu, en intégrant cette école, acquérir les fondements nécessaires pour m'intégrer au monde de l'actuariat et mener à bien mon mémoire.

Je tiens à remercier tout particulièrement mon maître d'apprentissage, Rémi Michel, qui m'a encadrée tout au long de mon alternance. Rémi m'a guidée et assistée pour la réalisation de mon mémoire. Il a pu m'aider à acquérir des compétences à la fois en tarifications IARD et en Data Science mais aussi des compétences actuarielles qui m'ont permis de mener à bien ce projet.

Je remercie également mon manager, Sébastien Sépart, qui m'a offert l'opportunité de travailler sur un sujet aussi intéressant.

J'adresse une attention particulière à mon équipe qui a su me soutenir et m'aider lorsque je me suis retrouvée confrontée à des difficultés.

Et finalement je souhaiterais remercier l'intégralité de la TA IARD de Generali pour son accueil chaleureux et sa disponibilité.

Table des matières

Introduction **5**

1 Contexte du Mémoire **6**
1.1 L'entreprise d'accueil : Generali . . . 6
1.2 Marché de l'assurance automobile . . . 6
1.3 Tarification de l'assurance automobile . . . 7
1.4 Présentation des garanties principales . . . 8
1.5 Description du processus de création de la variable d'interactions . . . 9

2 Théorie **12**
2.1 Les modèles linéaires généralisés . . . 12
2.2 Les algorithmes de machine learning utilisés . . . 14
2.2.1 Préliminaire : principe de validation croisée . . . 15
2.2.2 Régressions pénalisées . . . 15
2.2.3 Arbre CART . . . 17
2.2.4 Random Forest . . . 19
2.2.5 Boosting . . . 20

3 Révision tarifaire **22**
3.1 Méthodologie mise en place pour l'amélioration du zonier . . . 22
3.2 Récupération d'informations externes . . . 23
3.2.1 Sélection et normalisation des variables . . . 24
3.2.2 Formatage des variables . . . 25
3.3 Modélisation des fréquences et coûts moyens des garanties principales . . . 27
3.4 Création d'un zonier . . . 30

4 Ajout de la variable d'interaction **33**
4.1 Choix des résidus à modéliser . . . 33
4.2 Détermination du meilleur algorithme de machine learning par garantie . . . 36
4.2.1 Principe . . . 36
4.2.2 Garantie bris de glace . . . 38

4.2.3 Garantie dommages tous accidents 49
4.3 Création de la variable d'interaction 59
4.3.1 Choix de l'arbre appliqué aux résidus lissés 59
4.3.2 Garantie bris de glace 62
4.3.3 Garantie dommages tous accidents 73

Conclusion **83**

Bibliographie **84**

Table des figures

1.1 Une représentation simplifié de la construction du tarif commercial . . . 7
1.2 Processus de création de la variable d'interactions . . . 10

2.1 Un arbre CART avec son partitionnement de l'espace des variables explicatives . . . 18

3.1 Variables départementales . . . 25
3.2 Création de format sur le taux de mortalité infantile . . . 26
3.3 Significativité des variables de notre modèle . . . 28
3.4 Effet de la variable de l'âge du conducteur sur le modèle de fréquence du bris de glace . . . 28
3.5 Comparaison de deux modèles Emblem avant et après ajout de variables . . . 29

4.1 Distribution des résidus Bris de Glace : à gauche prime pure, à droite fréquence . . . 34
4.2 Distribution des résidus Dommages : à gauche prime pure, à droite fréquence . . . 35
4.3 Visualisation du fonctionnement d'un tuning et de l'importance des variables . . . 37
4.4 Résultat du tuning de régression BDG . . . 39
4.5 Tuning de reg_pram avec elasticnet_param=1 . . . 40
4.6 Tuning de elasticnet_ param avec reg_pram=0 . . . 40
4.7 Tuning Arbre . . . 42
4.8 Importance des variables sur l'arbre BDG . . . 42
4.9 Tuning Random Forest BDG . . . 44
4.10 Importance des variables du Random Forest BDG . . . 45
4.11 Résultats tuning Gradient boosting BDG . . . 46
4.12 Tuning du gradient boosting BDG . . . 47
4.13 Importance des variables du gradient boosting BDG . . . 48
4.14 Tuning du paramètre elasticnet DOM . . . 51
4.15 Tuning du paramètre de régularisation DOM . . . 51
4.16 Tuning du paramètre de profondeur maximale DOM . . . 52
4.17 Importance des variables arbre de régression DOM . . . 53
4.18 Tuning du random forest DOM . . . 54
4.19 Importance des variables du random forest sur la garantie DOM . . . 55
4.20 Tuning du gradient boosting sur la garantie DOM . . . 57
4.21 Importance des variables avec gradient boosting sur la garantie DOM . . . 58

4.22 Schématisation de la création des deux variables d'interactions . . . 61
4.23 Importance des variables sur des arbres de profondeur 5 pour le bris de glace . . . 63
4.24 Comparaison des importances des variables d'interaction avec et sans prédiction des résidus bris de glace . . . 65
4.25 Expositions des deux variables d'interaction des résidus bris de glace . . . 66
4.26 Heatmap des interactions observées pour le BDG entre le type de carrosserie et l'âge du véhicule . . . 67
4.27 Extrait de chemin de la variable d'interactions BDG . . . 67
4.28 Interaction entre le type de carrosserie et l'âge du véhicule sur les résidus BDG . . . 68
4.29 effet des variables d'interactions dans le modèle de fréquence du bris de glace partiel . . . 70
4.30 effet des variables d'interactions dans le modèle de fréquence du bris de glace total . . . 71
4.31 Validation de l'usage de la variable d'interaction issue des résidus bruts sur la fréquence BDG partiel . . . 72
4.32 Validation de l'usage de la variable d'interaction issue des résidus bruts sur la fréquence BDG total . . . 73
4.33 Comparaison des importances des variables sur les échantillons bootstrap . . . 74
4.34 Exposition des deux variables d'interaction des résidus DOM . . . 75
4.35 Heatmap des interactions observées pour le DOM entre le type de carrosserie et la boite de vitesse . . . 76
4.36 Extrait de chemin de la variable d'interactions DOM . . . 77
4.37 Interactions entre le type de carrosserie et la boite de vitesse sur les résidus remodélisés . 77
4.38 Interactions entre le type de carrosserie et la boite de vitesse sur les les résidus non remodélisés 78
4.39 Effet de la variable d'interaction construite sur un arbre seul DOM . . . 79
4.40 Effet de la variable d'interaction construite après machine learning DOM . . . 80
4.41 Validation de l'usage de la variable d'interaction issue des résidus modélisés sur les coûts DOM . . . 81

Introduction

Le domaine de l'assurance automobile est un domaine très concurrentiel, qui pousse à une adaptation fréquente des prix de la part des assureurs. Pour pouvoir proposer des tarifs attractifs, les compagnies d'assurance doivent d'abord s'appuyer sur des modèles de sinistralité solides, qui leur assurent une segmentation optimale.

Les erreurs de ces modèles techniques peuvent être réduites par la captation d'informations supplémentaires significatives. Dans cette optique, le Big Data et les algorithmes de machine learning offrent de nouvelles opportunités pour exploiter des données volumineuses, avec des techniques complémentaires aux classiques modèles linéaires généralisés.

Mon mémoire s'inscrit dans cet objectif de sophistication lors de la révision tarifaire, qui consiste en une mise à jour du tarif. Nous utilisons des machine learning pour synthétiser des effets jusqu'alors absents des modèles. Plutôt qu'une modélisation directe de la prime pure par des machine learning, nous préférons utiliser ces derniers pour créer une nouvelle variable résumant les interactions. La valorisation de cette variable se fait ensuite en la réinjectant dans les modèles linéaires généralisés existants, réduisant par construction ses erreurs. Entre autres avantages, cette approche permet de conserver l'interprétabilité des coefficients tarifaires, indispensable en tarification automobile.

Avant d'aborder le cœur de ce mémoire, nous nous intéressons dans une première partie au contexte de cette étude, en abordant le marché de l'assurance automobile, l'entreprise d'accueil Generali, ainsi que les garanties sur lesquelles porte cette étude.

Dans une deuxième partie, nous détaillons les aspects techniques essentiels à la compréhension de la suite du mémoire, en particulier la théorie des modèles linéaires généralisés et de certains algorithmes de machine learning.

La troisième partie est consacrée à la révision tarifaire, où la mise à jour standard des modèles de sinistre est complétée par des informations externes supplémentaires, qui précisent un peu plus le zonier et diminuent par la même l'erreur d'estimation.

Ce cheminement nous permet finalement de détailler la construction et l'apport de la variable d'interaction qui a pour but de capter les interactions entre les variables sur les résidus, dans la quatrième et dernière partie de ce mémoire. Les résidus sont d'abord stabilisés par l'utilisation de machine learning. Nous créons ensuite un arbre sur les résultats de ces machine learning pour créer cette variable d'interaction. Enfin nous observons son impact sur la performance des modèles des garanties bris de glace et dommages tous accidents.

Chapitre 1

Contexte du Mémoire

1.1 L'entreprise d'accueil : Generali

Generali est une compagnie d'assurance fondée en 1831 à Trieste. Comme son nom l'indique, c'est une société d'assurance généraliste. Elle propose donc à ses clients divers types de contrats d'assurance. Ce groupe est aujourd'hui très présent dans le monde et se place parmi les leaders du marché européen. Generali France a été fondée en 2006, suite aux nombreuses implantations de filiales sur le territoire français. L'entreprise propose des produits pour satisfaire tous les besoins d'assurance, que ce soit en termes d'assurance vie ou non-vie, pour les particuliers comme les professionnels.

En ce qui concerne l'assurance automobile des particuliers, Generali fait partie des 10 premiers assureurs en France, en considérant les cotisations perçues sur l'année 2018 d'après une enquête FFA (Fédération Française de l'Assurance). De plus, la branche automobile représente 8% de l'activité de l'entreprise en 2018 ce qui la place en tête des produits d'assurance IARD (Incendie, Accidents et Risques Divers). Ainsi l'assurance automobile occupe une place particulière au sein de Generali France.

1.2 Marché de l'assurance automobile

Mon étude s'inscrit dans le cadre de l'assurance Automobile des particuliers en France métropolitaine chez Generali. Le produit d'assurance automobile est sujet à une forte concurrence entre organismes d'assurance. Cela nécessite donc de créer des tarifs toujours plus attractifs et segmentant pour les prospects ce qui implique, du côté de la tarification, de rechercher la performance des modèles utilisés. Cette recherche de performance s'est traduite ces dernières années par l'ajout de zonier aux modèles de risque, et plus récemment par l'usage de machine learning. Ces nouvelles méthodes, apportées par le développement des Big Data, devraient permettre une meilleure compréhension du risque de chaque individu, et donc, une meilleure segmentation de notre portefeuille et ainsi de pouvoir concrétiser les contrats pour les prospects portant les meilleurs risques.

Le marché de l'assurance automobile des particuliers est constitué de plus de 40 millions de véhicules de première catégorie en 2018 d'après une enquête FFA (*le marché de l'assurance automobile en 2018*, Juin 2019). Ce marché représente 39% de l'ensemble des cotisations de l'assurance dommages aux biens et responsabilité civile, ce qui en fait son produit principal. On observe une hausse des cotisations entre 2017 et 2018. Le ratio combiné net de réassurance de la branche était à 100% en 2018, cependant il est la plupart du temps au-dessus du seuil des 100% ; ce qui illustre le problème auquel les assureurs automobiles sont confrontés : il faut trouver le compromis entre un tarif agressif et rentabilité du portefeuille.

Il est donc nécessaire de construire de manière intelligente ses tarifs pour, à la fois pouvoir faire face à la rude concurrence, tout en ayant un produit permettant à l'entreprise de dégager des bénéfices. Un aspect important aussi est la revalorisation des contrats automobiles. La prime est modifiée à chaque échéance anniversaire du contrat, le but étant d'augmenter ou diminuer la prime selon la rentabilité du contrat avec la contrainte qui est de conserver le contrat en portefeuille. Il peut arriver qu'un contrat subisse une forte majoration de sa prime lorsqu'il est très déficitaire. Dans ce cas il est probable que ce soit une volonté de l'assureur pour inciter le client à résilier. Ce mémoire entre uniquement dans le cadre de la tarification des affaires nouvelles. Nous ne nous intéresserons pas aux revalorisations des contrats en portefeuille.

1.3 Tarification de l'assurance automobile

La tarification consiste à construire un prix attractif et segmentant permettant d'attirer des clients et donc augmenter la taille du portefeuille en concrétisant plus de devis. Il faut donc essayer de modéliser au mieux le risque que porte chacun, ainsi que les coûts pour l'entreprise pour pouvoir aboutir à un tarif cohérent avec le profil des individus. L'équipe chargée de la tarification va donc mettre en place une décomposition de la prime pure en des modèles de fréquences et de coûts moyens, avec une hypothèse d'indépendance entre ces deux grandeurs, en se basant sur la sinistralité des contrats en portefeuille pour pouvoir attribuer à chaque caractéristique déterminée comme étant discriminante, un certain effet sur le tarif. C'est à partir de la prime pure, qu'est déterminé le tarif technique qui prend également en compte les chargements. Pour aboutir au tarif commercial il faut également y intégrer des ajustements commerciaux. Ce processus est schématiquement représenté dans la *Figure 1.1*. Notre étude s'effectue uniquement dans le cadre de la modélisation de la prime pure par sa décomposition en fréquence et coûts moyens.

FIGURE 1.1 – Une représentation simplifié de la construction du tarif commercial

Mon mémoire s'inscrit dans la révision tarifaire des affaires nouvelles des contrats d'assurance automobile des particuliers en France métropolitaine. La révision tarifaire dans cette branche se fait annuellement. Elle vise à prendre en compte les différents changements de sinistralité observés, en améliorant les modèles déjà en place. Le but est de proposer un tarif aux assurés qui soit, le plus adapté possible, à leurs risques réels et à l'évolution de leurs besoins. Un tarif mal calibré peut engendrer de l'anti-sélection (ou sélection adverse), c'est à dire, les tarifs attireraient des mauvais profils (profils risqués) et feraient fuir les bons ce qui détériorerait la rentabilité du portefeuille.

Pour répondre à ce besoin, l'équipe chargée de la tarification se constitue des bases de travail et met en place des méthodes de modélisation des risques. Son objectif est de choisir les différentes variables tarifaires à intégrer dans ses modèles, pour chacune des garanties qu'elle propose à ses clients. Pour cela, elle mesure l'impact de l'ajout de ces variables à l'aide d'indicateurs, comme la déviance ou le critère d'Akaike (AIC), permettant de mesurer la qualité des modèles. Elle met ensuite en place un zonier, afin de caractériser le risque et l'information géographique qu'elle ne pourrait pas prendre en compte autrement. En effet la sinistralité d'un assuré peut également dépendre de son lieu de résidence, cela peut jouer sur la fréquence et les coûts moyens des sinistres.

En temps normal la modélisation de la prime pure en assurance automobile se limite aux étapes précédemment évoquées. Il en résulte un modèle ainsi que des résidus qui lui sont associés. Nous avons décidé de travailler sur les résidus de cette modélisation. Nous allons les exploiter afin de créer une nouvelle variable à intégrer à nos modèles de fréquence et coûts moyens, si celle-ci s'avère pertinente. Pour cela nous allons implémenter différentes méthodes de machine learning, nous permettant d'obtenir cette nouvelle variable.

1.4 Présentation des garanties principales

Nous allons par la suite travailler sur les principales garanties du produit d'assurance automobile des particuliers de Generali. Ces garanties sont les suivantes : responsabilité civile (RC), vol (VOL), bris de glace (BDG), Dommage tous accidents (DOM). Generali propose des formules à ses clients incluant tout ou partie de ces garanties en plus de quelques autres qui ne seront pas abordées dans ce mémoire.

La garantie la plus représentée dans le portefeuille Generali est la garantie responsabilité civile. Elle est en effet une garantie légalement obligatoire. Elle est incluse dans toute formule d'assurance automobile des particuliers proposée par Generali. Cette garantie s'exerce lorsque l'assuré cause des dommages à autrui, que ce soient des dégâts matériels ou des dégâts corporels. Cela peut impliquer de très gros montants lorsqu'il s'agit d'un sinistre corporel, qui peuvent avoir des délais de clôture très importants. Cette garantie est donc un peu particulière de par sa décomposition matérielle et corporelle. Cela nécessite ainsi deux modélisations, d'un côté la modélisation des sinistres entrant dans la catégorie responsabilité civile matérielle et de l'autre les sinistres entrant dans la catégorie des sinistres corporels.

La garantie bris de glace et une garantie qui n'est pas une obligation légale d'assurance. Elle intervient lorsqu'une partie vitrée du véhicule est endommagée. L'assureur rembourse alors la réparation ou le coût de remplacement de la pièce à l'assuré si le sinistre entre dans le cadre défini de la garantie, dans les dispositions générales ou les conditions particulières, de la garantie.

La garantie vol intervient lors du vol du véhicule. Les assurés ayant choisi de souscrire à cette garantie sont indemnisés à la hauteur de la valeur de leur véhicule si le sinistre entre dans le cadre défini par les conditions générales ou les dispositions particulières de son contrat.

La garantie dommages tous accidents intervient lorsque le véhicule assuré subit des dommages. Si le sinistre entre dans ce cadre de cette garantie et que l'assuré y a souscrit, il est indemnisé à la valeur de remplacement.

Quelle que soit la formule choisie, donc à fortiori, les garanties souscrites, l'assuré va devoir verser, à une fréquence déterminée, une prime dépendant des méthodes de calcul sélectionnées par les équipes de tarification. On peut affecter à chaque garantie souscrite, une part de la prime totale versée.

1.5 Description du processus de création de la variable d'interactions

Rappelons que l'objectif de ce mémoire est d'améliorer la prédictivité des modèles de tarifications automobile par l'exploitation de leurs résidus avec une approche machine learning. Pour ce faire nous pouvons décomposer notre démarche en deux parties :

- la mise à jour des modèles usités pour la construction du tarif affaire nouvelle,
- exploitation des résidus de ces modèles.

Notre étude commence donc par la mise à jour des modèles avec une première phase de modélisation de nos fréquences et coûts par des variables internes à l'entreprise et des variables externes issues du site de l'INSEE comme indiqué dans l'étape 1 de la Figure 1.2 ci-dessous. Dans une deuxième étape, nous exploitons ces premiers résidus de modélisation pour mettre à jour nos zoniers (étape 2 Figure 1.2). Une fois ces zoniers mis à jour, nous les intégrons à nos modèles de l'étape 1 tout en enlevant les variables externes.

En effet ces variables contiennent de l'information sur le risque spatial et ce risque a été intégré dans les zoniers. Il n'est donc plus nécessaire de les conserver. A la fin de cette étape (étape 3 de la figure 1.2) nous en avons fini avec la mise à jour du tarif de manière classique. Les étapes 4 à 6 concernent l'exploitation des résidus de modélisation de l'étape 3 afin de créer une nouvelle variable tarifaire.

Nous commençons à l'étape 4 la construction de notre variable d'interactions par la modélisation des résidus de fréquence et de prime pure issues de l'étape 3. Nous obtenons des résidus modélisés sur lesquels nous construisons des arbres de régression dans l'étape 5. En effet construire des arbres sur ces résidus nous permet de créer des classes en fonction de nos variables explicatives. Les branches de cet arbre constitueront les modalités de notre variable d'interactions que nous obtenons à la fin de cette étape. Finalement nous intégrons la variable d'interactions dans nos modèles afin de mesurer son effet et de conclure quant à sa significativité dans l'étape finale 6.

FIGURE 1.2 – Processus de création de la variable d'interactions

Pour la réalisation de ces travaux nous disposons d'une base comportant des informations sur la sinistralité, les caractéristiques des contrats, les informations relatives aux véhicules et les personnes assurés du portefeuille automobile de Generali. Chaque ligne correspond à une période de risque, c'est-à-dire, une période pendant laquelle le risque est considéré comme identique. Sur cette période il n'y a pas de modifications des informations dont nous disposons. Cette base est enrichie de données issues de bases open data accessible sur le site de l'INSEE. C'est cette base qui nous sert dans un premier temps à mettre à jour nos tarifs (étape 1 à 6 de la Figure 1.2). Après la mise à jour des tarifs, nous intégrons également à chaque période de risque l'estimation réalisée par nos modèles à la fois de la prime pure et l'estimation de la fréquence. Nous calculons des résidus de prime pure et de fréquence que nous ajoutons également à notre base. Cela va nous permettre dans un second temps de travailler sur ces résidus pour chercher à les expliquer et créer notre variable d'interaction.

Finalement, notre base comportera plus de 21 millions de lignes et 35 colonnes. D'un point de vue pratique, cette base pèse plus de 10Go, cela est particulièrement lourd pour les logiciels que nous utilisons dans la seconde partie de ce mémoire.Pour la réalisation de cette étude nous avons envisagé de programmer nos travaux suivant différentes méthodes. Tout d'abord nous avons essayé de travailler en local avec R. Cela a été impossible au vu de la taille de notre base, qui était alors impossible de charger dans l'environnement de travail. Nous avons donc choisi de travailler ensuite sur JupyterLab en lien avec Hadoop pour disposer d'une puissance de calcul bien supérieure. Nous avons commencé par programmer l'ensemble de notre

étude en utilisant des packages R classiques sur un sous-échantillon de notre base afin de nous assurer que tout fonctionnait correctement avant de lancer de gros traitements. Ces tests s'étant révélés concluants nous avons décidé de lancer nos traitements sur l'ensemble de notre base. Cela a généré une saturation des ressources disponibles, la méthode était donc inexploitable. Finalement nous avons fait le choix de travailler avec R sur Hadoop en langage distribué via le package sparklyR pour essayer d'obtenir des programmes les plus rapides possibles et utilisables dans des conditions réelles. Cela nous a conduit à recoder intégralement notre programme. Néanmoins la plateforme sur laquelle nous travaillons partage ses ressources entre les différents utilisateurs et a été sous-dimensionnée par rapport au besoin réel des différentes équipes. Cela a engendré de nombreux plantages lorsque les capacités de ladite plateforme sont saturées et fait perdre toute progression non enregistrée des process en cours lors des plantages. De plus, cette plateforme subit de nombreuses instabilités qui peuvent influer sur la bonne exécution des traitements.

La taille de notre base couplée avec des outils inadaptés a engendré d'énormes temps de calculs qui peuvent aller jusqu'à 8h dans le cas des garanties les plus volumineuses, ce temps d'exécution s'est retrouvé augmenté par les instabilités de la plateforme précédemment évoquées. Cela nous a limité dans le tuning des algorithmes de machine learning que nous utilisons dans la quatrième partie du mémoire (*Chapitre 4*) car c'est un processus qui nécessite de faire varier des paramètres sur différents intervalles. Plus nous faisons varier nos paramètres, plus les algorithmes sont lents en testant les combinaisons possibles de ces paramètres. Réduire le nombre de paramètre, réduit les temps de calculs mais également détériore notre approximation des paramètres optimaux. Cela dégrade donc nos résultats du *Chapitre 4* de ce mémoire.

Chapitre 2

Théorie

Dans cette partie, nous détaillerons les bases théoriques nécessaires pour comprendre les mécanismes mis en œuvres dans la suite de ce mémoire. Nous considérons que nous disposons des observations (Y_i, X_i) pour chaque individu i de nos bases de modélisation de taille n.

Avec :

- Y_i la réalisation sur l'individu i de la variable que nous cherchons à expliquer
- X_i un vecteur de taille m composé des variables explicatives $x_{i,j}$ (caractéristiques) de l'individu i

Il est également nécessaire, de savoir distinguer régression et classification. Lorsque nous réalisons une régression, nous cherchons à prédire une variable quantitative, à l'inverse d'une classification, qui a pour but de prédire une variable qualitative.

Puisque nos applications s'effectuent dans le cadre de la régression, nous ne détaillerons que la théorie correspondante. Que ce soit lors de l'application de nos modèles linéaires généralisés, des algorithmes de machine learning ou de la création de notre variable d'interaction, nous ne travaillons qu'avec des variables à expliquer de type numériques.

2.1 Les modèles linéaires généralisés

Les modèles linéaires généralisés sont utilisés en assurance auto pour expliquer, par exemple, la fréquence ou les coûts moyens en tarification. Ce sont des modèles très largement utilisés dans le monde de l'assurance IARD. Nous nous en servirons dans la suite de ce mémoire pour la modélisation de nos garanties.

Les modèles linéaires généralisés sont une extension des modèles linéaires. Voici une formulation des modèles linéaires :

$$Y_i = \sum_{j=1}^{m} \beta_j x_{i,j} + \epsilon_i$$

Nous supposons dans ce cas que les $x_{i,j}$ sont indépendants, connus et les β sont les paramètres à estimer. Dans le cadre de ces modèles, la variable à expliquer possède une relation linéaire avec ses variables explicatives. L'usage des modèles linéaires généralisés permet de s'affranchir de cette relation et admet donc des dépendances non linéaires. Nous pouvons exprimer ces derniers de la manière suivante :

$$g(E[Y_i|X_i]) = \sum_{j=1}^{m} \beta_j x_{i,j}$$

qui peut se réécrire :

$$g(E[Y_i|X_i]) = \beta X_i^t$$

Un modèle linéaire généralisé dépend de trois composantes :

- Une variable à expliquer dont la distribution fait partie de la famille exponentielle,
- Un prédicteurβX_i^t,
- Une fonction lien g permettant de faire le lien entre la moyenne des Y_i et le prédicteur. La fonction g doit être inversible.

L'usage des modèles linéaires généralisés requiert les hypothèses suivantes :

- conditionnellement à X_i, les Y_i sont indépendants,
- Y suit une loi de la famille exponentielle.

La famille exponentielle

La famille exponentielle est composée des lois de paramètres θ et ϕ dont la fonction de densité s'écrit sous la forme suivante :

$$f(y|\theta,\phi) = exp\left(\frac{y\theta - b(\theta)}{a(\phi)} + c(y,\phi)\right)$$

Où b est une fonction définie sur les réels et deux fois dérivable, a est une fonction définie sur les réels et non nulle et c est une fonction définie sur $\mathbf{R}^2$.

Quelques lois appartenant à cette famille : loi de Bernoulli, loi binomiale, loi binomiale négative, loi de Poisson, loi Gaussienne, loi Gamma.

Nous pouvons facilement, à partir de l'écriture de la densité précédente, retrouver l'espérance et la variance d'un variable aléatoire Y suivant une loi de cette famille par les formules :

- $E[Y] = b'(\theta)$
- $V[Y] = b''(\theta)\phi$

De plus chaque loi de cette famille possède un fonction $g_*(.)$, dite fonction de lien canonique qui relie son espérance μ au paramètre θ comme suit :

$$g_*(\mu) = \theta$$

Nous exploitons par la suite les lois de Poisson pour modéliser les fréquences des sinistres des garanties étudiées et Gamma pour modéliser leurs coûts dans les modèles linéaires généralisés, avec la fonction de lien $log(.)$.

Fonctionnement

Lors de la mise en place d'un modèle linéaire généralisé il faut tout d'abord choisir une loi de distribution de la variable réponse. Cette distribution doit correspondre aux données que nous cherchons à prédire. Nos données sur la sinistralité de notre portefeuille nous poussent à choisir pour le modèle de fréquence des sinistres une loi de Poisson. Cette loi discrète permet de modéliser un comptage du nombre de sinistres. En ce qui concerne la fonction lien, cela dépend également de la grandeur à modéliser. Une variable de comptage est positive ou nulle, nous optons pour une fonction de lien log. Une telle fonction permet de prédire des grandeurs supérieures à zéro lors du passage à l'exponentielle. En effet si :

$$log(E[Y_i|X_i]) = X_i^t\beta$$

alors :

$$E[Y_i|X_i] = exp(X_i^t\beta)$$

La fonction de lien log permet de plus d'avoir un modèle multiplicatif :

$$E[Y_i|X_i] = \prod_{j=1}^{n} exp(x_{i,j} \times \beta)$$

Pour les modèles de coûts, nous choisissons d'utiliser une loi gamma avec une fonction de lien log. Une fois la fonction de lien et la distribution de la variable réponse déterminées, nous pouvons estimer les paramètres β par maximum de vraisemblance, comme c'est le cas lors d'une régression linéaire.

2.2 Les algorithmes de machine learning utilisés

Nous aborderons dans cette partie les bases théoriques pour comprendre les algorithmes utilisés pour la construction de la variable d'interaction. Ces algorithmes comprennent du gradient boosting, du random forest et des arbres CART et des régressions elasticnet. Nous nous y référerons, par abus de langage, par les termes "machine learning".

Il existe trois types de méthodes d'apprentissage : les méthodes supervisées, les méthodes non supervisées et les méthodes semi-supervisées. Les méthodes supervisées désignent les méthodes où l'algorithme dispose d'exemples de résultats attendus pour apprendre. Ce n'est pas le cas des méthodes non supervisées. Celles-ci demandent à l'algorithme d'apprendre de manière autonome. Elles s'utilisent essentiellement dans des buts de regroupement d'éléments de groupes hétérogènes suivant des critères communs. Les méthodes semi-supervisées ne sont qu'un mélange des deux types d'apprentissages précédents.

Les méthodes d'apprentissages supervisées sont les méthodes qui seront mises en oeuvre et décrites dans ce mémoire.

2.2.1 Préliminaire : principe de validation croisée

La validation croisée, ou cross validation, est une méthode que nous utilisons par la suite pour déterminer les hyper-paramètres de nos modèles.

Il en existe de plusieurs types. Celle dont nous nous servons est la *k-fold cross validation*. Son principe est le suivant :

1 - séparation de la base de taille m en k échantillons de tailles identiques n
2 - construction de modèles sur ces k échantillons et validation avec la part restante de la base (m-n)
3 - calcul de la moyenne des k erreurs ainsi déterminées

L'avantage de ce type de validation croisée est qu'elle nous permet d'utiliser toutes les données disponibles.

2.2.2 Régressions pénalisées

Nous allons parler dans cette section d'algorithmes de régularisation. Ils dérivent des régressions linéaires multiples, auxquelles s'ajoute un paramètre de pénalisation. Nous cherchons à estimer la variable Y par le modèle suivant :

$$Y = X^t\beta + \epsilon$$

Où Y est la variable à expliquer, X le vecteur des variables explicatives, β les paramètres associés et ϵ un terme d'erreur.

Le but de ces méthodes est de gérer la colinéarité des variables tout en faisant converger le modèle.

Nous nous intéresserons tout d'abord à la régression de Ridge, puis à la régression de Lasso et finalement à la régression elasticnet qui est un mélange des deux.

Régression de Ridge

La régression de Ridge introduit un critère de pénalisation qui influe sur les valeurs de β. Ainsi le calcul de β_{Ridge} est le suivant :

$$\hat{\beta}_{Ridge} = \sum_{i=1}^{n} \left(y_i - \beta_0 \sum_{j=1}^{m} x_{i,j} \times \beta_j \right)^2 + \lambda \sum_{j=1}^{m} \beta_j^2$$

Cette régression se base sur un critère des moindres carrés avec une norme de type l_2 qui est représentée par le terme $\lambda \sum_{j=1}^{m} \beta_j^2$.

Le paramètre de complexité est représenté par λ. Il prend des valeurs positives. Plus il est proche de zéro, et moins la régression est pénalisée, ce qui induit un risque de sur-apprentissage. Plus ce paramètre est élevé et plus nous risquons de sous-ajuster le modèle. Ainsi il est essentiel de bien choisir ce paramètre. Nous le ferons dans notre application par l'usage de la validation croisée.

Bien que contraignant les valeurs des paramètres β, la régression de Ridge ne permet pas de réduire le nombre de variables explicatives, ce qui réduit sa lisibilité.

Régression de Lasso

Cette régression permet une sélection de variables explicatives, elle a été développée par R.Tibshirani en 1996. Elle est basée sur la norme l_1. Avec cette méthode, le calcul des β_{Lasso} s'effectue de la manière suivante :

$$\hat{\beta}_{Lasso} = \sum_{i=1}^{n} \left(y_i - \beta_0 \sum_{j=1}^{m} x_{i,j} \times \beta_j \right)^2 + \lambda \sum_{j=1}^{m} |\beta_j|$$

Le paramètre λ doit être positif. Quand il prend la valeur zéro, l'estimateur de β est celui des moindres carrés ordinaires. Lorsque λ augmente, la pénalisation force certains paramètres à la valeur zéro. Cela a pour conséquence d'enlever la variable du modèle. Plus λ est grand, plus il y a de variables avec un coefficient nul.

Le paramètre de régularisation de la régression de Lasso est donc également à choisir avec soin. La cross-validation permet une bonne sélection de celui-ci. C'est de cette manière que nous procéderons par la suite.

Cette méthode a cependant le défaut de négliger les corrélations entre les variables explicatives, notamment entre celles retenues dans le modèles et celles dont le coefficient est nul. La regression elastic net peut être une alternative à ce problème.

Régression Elastic Net

Ce type de régression est un mélange des régressions de Ridge et de Lasso. Elle a été développée en 2005 par Zou et Hastie. Elle permet d'effectuer une régression pénalisée en limitant les défauts des deux types de régressions précédentes : elle tient compte des corrélations entre variables tout en les sélectionnant avec parcimonie.
Les coefficients β sont estimés de la manière suivante :

$$\hat{\beta}_{elasticnet} = \sum_{i=1}^{n} \left(y_i - \beta_0 \sum_{j=1}^{m} x_{i,j} \times \beta_j \right)^2 + \lambda \sum_{j=1}^{m} \left[\alpha |\beta_j| + (1-\alpha)\beta_j^2 \right]$$

Nous introduisons ici le paramètre α qui prend ses valeurs sur $[0; 1]$. Il permet une pondération entre les régressions de Ridge et Lasso. Le choix de ce paramètre va dépendre de notre base de données.

Elastic net correspond à une régression de Ridge lorsque α vaut zéro et une régression de Lasso quand il vaut un.

2.2.3 Arbre CART

Un arbre de décision CART (Classification And Regression Tree), est une méthode statistique permettant de construire des prédicteurs. Cette méthode a été introduite par Breiman et al. en 1984. Elle peut être utilisée aussi bien pour des problématiques de régression que de classification. Nous ne nous intéresserons qu'au cadre de la régression puisque que c'est celui de notre étude.

L'arbre CART partitionne notre base de départ et, en régression, effectue la moyenne locale de notre variable à expliquer sur chacune des partitions. Pour cela il effectue des divisions binaires successives de l'espace des variables explicatives. Ces divisions sont pensées de manière à former des partitions optimales pour la prédiction. Vous trouverez ci-dessous une représentation de la construction et du fonctionnement d'un arbre CART, avec X (*Figure 2.1*).

Un arbre de décision CART (Classification And Regression Tree) est un algorithme donnant des moyennes locales par partitions. Ces partitions sont obtenues en réalisant des divisions successives d'hyperplans. Un arbre est composé de nœuds, qui représentent les divisions d'hyperplans, et de feuilles ou nœuds terminaux, qui représentent les partitions finales. Le premier nœud est appelé nœud racine. Les nœuds suivants sont appelés nœuds fils.

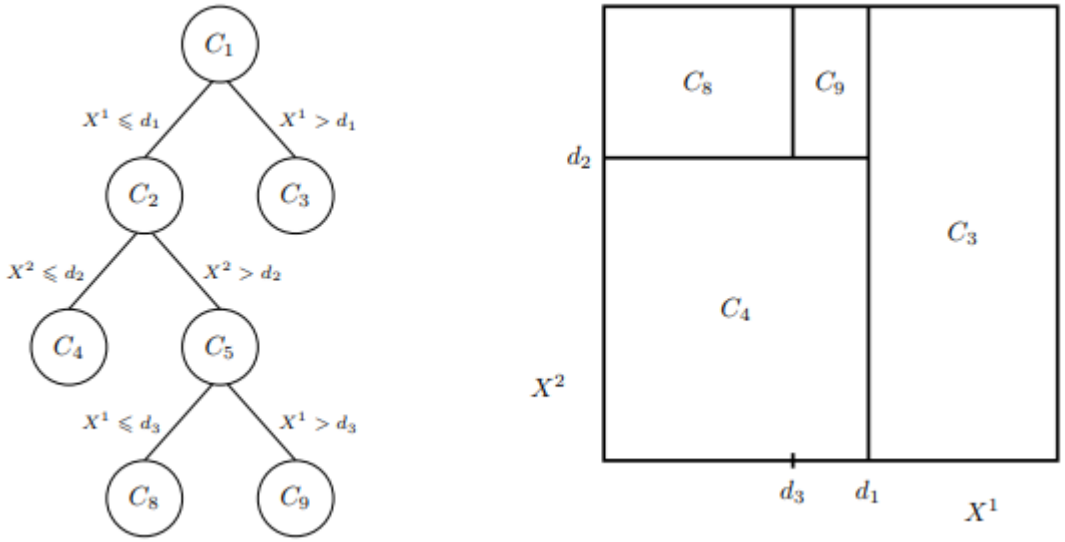

FIGURE 2.1 – Un arbre CART avec son partitionnement de l'espace des variables explicatives

Construction d'un nœud

Le choix de la division se fait par hétérogénéité des observations de la variable à expliquer sur nœud. En régression on mesure cette hétérogénéité par la variance. On cherche à minimiser la différence entre la variance du nœud parent et la somme des variances des deux nœuds fils. Autrement dit, nous cherchons à minimiser la quantité suivante :

$$Variance_{parent} - (Variance_{droite} + Variance_{gauche})$$

Une fois la division effectuée, le procédé précédent est réitéré sur les nœuds fils. A la fin de chaque branche de l'arbre se situe une feuille, et nous attribuons dans le cas de la régression la valeur moyenne de la variable à expliquer de la partition constituant la feuille.

Un arbre CART se construit en élaborant dans un premier temps un arbre dit maximal. Il s'agit d'un arbre dont les feuilles sont soit constituées d'un groupe d'observations homogènes, soit comportent le nombre d'individus minimal qui peut être déterminé par l'utilisateur au préalable. Dans les deux cas précédents, plus aucune division n'est possible, l'arbre est le plus grand possible d'où son nom arbre maximal.

Il s'agit ensuite d'élaguer l'arbre en extrayant une sous-suite de l'arbre maximal précédemment construit. Un arbre trop élagué a un grand biais mais une faible variance contrairement à l'arbre maximal qui a un faible biais mais une grande variance. Nous recherchons donc un modèle avec un faible biais (pouvoir prédictif) et une faible variance (généralisable à d'autres données).

Le but de l'élagage est de trouver le meilleur arbre en termes d'erreur de généralisation (erreur sur une base de test ou avec validation croisée) à l'aide d'un critère pénalisé. Le critère de pénalisation dépendra du nombre de feuilles de l'arbre. Il n'est pas nécessaire de reconstruire un arbre entier à chaque longueur d'arbre testée. En effet, cela serait beaucoup trop long, il nous suffit juste d'élaguer successivement les branches de l'arbre. La seule condition pour la construction successive d'arbres est que leurs racines doivent être celles de l'arbre maximal. Nous pénalisons ensuite l'erreur d'ajustement de l'arbre élagué, nous pouvons par exemple appliquer le critère des moindres carrés.

Si T est l'arbre élagué de T_{max}, il s'agit donc de pénaliser :

$$\overline{err}(T) = \frac{1}{n} \sum_{t\ feuille\ de\ T} \sum_{(x_i,y_i)\in t} (y_i - \overline{y_t})^2$$

Nous appliquons le critère des moindres carrés suivant :

$$crit_\alpha = \overline{err}(T) + \alpha|T|$$

avec $|T|$ étant une fonction linéaire du nombre de feuilles de l'arbre T qui quantifie sa complexité. Ainsi de manière évidente, augmenter α pénalise les arbres comportant le plus de feuilles.

Nous avons vu jusqu'à présent que les arbres CART subissent une forte influence de leurs bases d'apprentissage. Une des solutions pour faire face à cet aspect est l'usage de méthodes ensemblistes comme les forêts aléatoires, le bagging, ou le boosting. Ces méthodes permettent de contrer l'effet de sur-apprentissage qu'il peut y avoir avec les arbres CART en agrégeant plusieurs arbres. Ainsi, ils permettent d'obtenir des résultats plus stables et généralisables. Cependant, ces méthodes ont le défaut de réduire l'interprétabilité des résultats.

2.2.4 Random Forest

Pour bien comprendre les algorithmes de random forest, il faut tout d'abord avoir des notions de bases sur les algorithmes de bagging, dont les random forests sont apparentés. Le bootstrap aggragation (bagging) est un ensemble de méthodes ensemblistes qui construisent des arbres maximaux sur des échantillons bootstrap. Elle a été introduite par L. Breiman en 1996.

Le bagging repose sur le tirage aléatoire avec remise de B échantillons sur une base initiale. L'algorithme construit ensuite sur chacun de ces échantillons un arbre maximal. L'agrégation se fait sur les arbres construits. Dans le cadre de la régression, cette agrégation se fait en calculant la moyenne des B estimations des arbres maximaux. Augmenter le nombre d'arbre améliore l'erreur d'apprentissage, mais un trop grand nombre d'arbre induit un fort risque de sur-apprentissage. Ce paramètre est donc à calibrer avec attention.

En ce qui concerne les forêts aléatoires, elles ont été développées par L. Breiman en 2001. Cette méthode ensembliste s'apparente au bagging en y ajoutant de l'aléa lors de la construction des arbres.

Tout comme le bagging, les forêts aléatoires construisent des arbres indépendants sur des échantillons bootstrap. Cette méthode varie néanmoins lors de la création des nœuds. Le bagging choisit la meilleure division parmi toutes les variables explicatives alors que le random forest choisit la meilleure division parmi un échantillon aléatoire des variables explicatives. Une fois la forêt construite, les différents arbres sont agrégés, comme lors d'un bagging.

Avec cette méthode chaque arbre est moins performant qu'un arbre lors d'un simple bagging. Cependant, tirer les variables explicatives des arbres au hasard permet de réduire leurs dépendances et donc d'améliorer le modèle final.

Le Random Forest se construit de la manière suivante : l'algorithme agrège un ensemble d'arbres qu'il aura construit sur des vecteurs aléatoires d'une base de départ. Nous supposons que les vecteurs aléatoires sont indépendants les uns des autres et qu'ils sont issus de la même distribution. Une fois qu'un certain nombre d'arbres (calibré au préalable) est créé, l'algorithme choisit la solution la plus représentée. En construisant une forêt, tout comme le bagging, augmenter le nombre d'arbres permet de réduire l'erreur d'apprentissage mais à partir d'un certain seuil, le modèle entre en situation de sur-apprentissage, et donc n'est plus généralisable.

Lors de la construction d'une forêt aléatoire, nous pouvons jouer sur quatre paramètres :
- la taille de chaque échantillon bootstrap
- le nombre d'arbres de la forêt
- la profondeur de chaque arbre
- le nombre de variables pour la construction de chaque arbre

Nous avons vu que le nombre d'arbres est un paramètre essentiel à calibrer, cependant, par des contraintes utilisateur, nous fixerons ce paramètre à 500 dans la suite de notre étude.

En ce qui concerne le nombre de variables à tirer aléatoirement, il existe plusieurs méthodes. Breiman recommande d'en tirer $\sqrt{m}$ si m est le nombre de variables explicatives du modèles en cas de classification et $\frac{m}{3}$ en cas de régression.

2.2.5 Boosting

Le boosting repose sur l'agrégation d'estimateurs. Dans le cadre de la régression cela se fait en les moyennant. Ces techniques ont été développées par Freund & Schapire (1996).

Contrairement au bagging, il s'agit de méthodes séquentielles : chaque estimateur s'adapte aux résultats du précédent. L'estimateur k apprend de l'estimateur $k-1$ en améliorant les lacunes de son prédécesseur. Pour cela, les algorithmes ont une fonction de perte à minimiser avec une augmentation des poids des

mauvaises prédictions précédentes.

Les prédicteurs sont composés de règles faibles de décision. La fonction $G(.)$ est une règle faible si :

$$\exists\ \gamma \quad tel\ que \quad P(G(X) \neq Y) = \frac{1}{2} - \gamma$$

En ce qui concerne la fonction de perte, elle représente l'écart entre la variable à expliquer et les prédictions. Cette fonction, usuellement notée $l(.)$, doit être convexe pour pouvoir la minimiser plus facilement.

Ainsi, l'algorithme de boosting cherche à minimiser la quantité suivante, avec n représentant le nombre d'observations :

$$\frac{1}{n}\sum_{i=1}^{n} l(Y_i, g(X_i))$$

Ces algorithmes restent très populaires avec par exemple l'extrem gradient boosting réputé pour ses performances. Nous détaillons dans la suite de cette section l'algorithme de gradient boosting, utilisé dans le Chapitre 4.

Gradient boosting

Cette méthode combine le principe du boosting avec la descente de gradient. Elle utilise la méthode du gradient pour minimiser la fonction de perte. Nous cherchons à minimiser :

$$L = E[l(Y, G(X))] = L(G(X_1), ..., G(X_n))$$

Pour cela nous effectuons nos itérations de boosting en ayant comme condition initiale :

$$\begin{pmatrix} \hat{G}^{[0]}(X_1) \\ . \\ . \\ . \\ \hat{G}^{[0]}(X_n) \end{pmatrix}$$

Puis l'algorithme passe de l'itération k à l'itération $k+1$ de la manière suivante avec le taux d'apprentissage ν :

$$\begin{pmatrix} \hat{G}^{[k+1]}(X_1) \\ . \\ . \\ . \\ \hat{G}^{[k+1]}(X_n) \end{pmatrix} = \begin{pmatrix} \hat{G}^{[k]}(X_1) \\ . \\ . \\ . \\ \hat{G}^{[k]}(X_n) \end{pmatrix} + \nu \begin{pmatrix} -\frac{\partial L}{\partial G(X_1)}(\hat{G}^{[k]}(X_1)) \\ . \\ . \\ . \\ -\frac{\partial L}{\partial G(X_n)}(\hat{G}^{[k]}(X_n)) \end{pmatrix}$$

Les itérations sont répétées jusqu'à obtenir les $\hat{G}^{[optimal]}$. Nous pouvons en extraire ensuite une estimation de la règle de décision finale.

Chapitre 3

Révision tarifaire

La révision tarifaire constitue la première étape de mon mémoire, et servira de base aux études de type machine learning. Elle consiste à revoir les modèles de risques, qui sont la base technique d'adaptation des tarifs. Sa mise à jour annuelle, fonction de l'évolution du risque observée sur la sinistralité récente du portefeuille, permet de garder des modèles performants dans la prédiction des sinistres à venir.
Nous avons au départ le modèle GLM suivant que nous souhaitons mettre à jour :

$$g(E[Y]) = \alpha X_{internes} + \alpha zonier_{avant\ refonte} + \epsilon_{avant\ revision}$$

Avec :

- Y la variable à expliquer
- X les variables explicatives
- ϵ les résidus du modèle
- $g(.)$ fonction de lien

Nous commençons par récupérer de l'information externe à l'entreprise sur des bases Open Data. Nous modélisons une première fois nos garanties, sans information géographique. Puis nous construisons notre zonier sur les résidus de cette modélisation. Nous mettons ensuite à jour nos modèles en utilisant cette fois notre zonier comme variable explicative.

3.1 Méthodologie mise en place pour l'amélioration du zonier

Notre travail commence par la modélisation classique de notre prime pure par garantie, en la décomposant en fréquence et coût des sinistres (sous l'hypothèse d'indépendance de ces deux grandeurs). Cela consiste en la modélisation par des modèles linéaires généralisés, de la fréquence et du coût moyen de chacune de nos garanties avec une sinistralité fixée aux trois années de survenance. Etant donné le nombre de véhicules en portefeuille, cette profondeur d'historique est suffisante pour avoir des modèles robustes. Les coûts des sinistres sont supposés indépendants de leur occurence, ce qui nous permet de faire le découpage

fréquence-coûts moyens.

Pour nos modélisations, que nous effectuerons avec le logiciel Emblem, nous utilisons d'une part des variables internes à l'entreprise, et d'autre part des données issues de bases Open Data. Nous lions ces données à nos bases via le code INSEE des communes. Nous créons ensuite un zonier pour chaque garantie et les modélisons à nouveau sous Emblem en remplaçant nos données externes par le zonier créé.

Composition de la base de modélisation :

Nous travaillons avec des bases constituées de nos contrats sur les différentes périodes de risques auxquelles ils sont soumis. Une période de risques correspond à un intervalle de temps pendant lequel les données risques sont restées identiques. L'exposition au risque, proratisation annuelle de cet intervalle, nous permet de pondérer nos observations.

Nous travaillons avec les contrats quatre roues des deux produits automobile Generali les plus récents ("l'Auto Generali" et "Felicita"). La sinistralité prise en compte est celle survenue durant chaque période d'observation, toutes vues quatre mois après la dernière année de survenance.

L'objectif est de représenter les sinistres avec un coût le plus adéquat à la réalité. Nous souhaitons ainsi prendre en compte le coût du sinistre le plus proche du moment où celui-ci ne bougera plus, c'est-à-dire lorsqu'il sera clos. Les garanties corporelles sont beaucoup plus longues à se finaliser comparées aux matérielles. Le taux de clotûre des matériels est élevé au moment où l'on observe ces sinistres, c'est pourquoi au niveau des modélisations de coûts moyens, nous sélectionnons uniquement les sinistres clos. Pour les corporels, non suffisamment clos, on conservera par contre tous les sinistres. Concernant les modélisations de fréquence, cette problématique n'est pas aussi importante car les sinistres très tardifs sont rares : un sinistre se déclare rarement cinq mois ou plus après qu'il soit survenu. Ainsi, pour cette partie on utilisera tous les sinistres en présence, qu'ils soient clos ou non.

Nos bases contiennent les données risques, que nous exploitons lors de nos modélisations, qui peuvent être de plusieurs natures :

- données risques contrat : la formule de garanties souscrites, le fractionnement des paiements, ...
- données risques liées au conducteur : âge, ancienneté de permis, catégorie socio-professionnelle, ...
- données risques relatives au véhicule : ancienneté, prix, marque, puissance fiscale, ...

3.2 Récupération d'informations externes

Récupérer le plus d'informations possible nous permettra de mieux discriminer la population, et par extension, d'améliorer nos modèles. Nous avons donc tout d'abord récupéré des variables Open Data sur le site de l'INSEE, contenant de l'information liée à des emplacements géographiques. Ces informations n'étaient que partiellement présentes dans les bases de travail jusqu'à présent, leur apport contribuera à perfectionner le zonier.

Les données récupérées distinguent différents niveaux de détail :

— communal
— départemental
— régional

Et contient de l'information sur divers domaines tels que :

— l'économie
— la démographie
— le pouvoir d'achat
— les conditions de vie
— le marché du travail
— les entreprises
— les domaines d'activité
— l'aménagement du territoire

Ces informations socio-démogaphiques nous permettront d'expliquer par la suite la part spatiale dans notre sinistralité par le biais d'un zonier.

Avant de commencer dans la modélisation, il nous faut au préalable retraiter l'information pour la rendre exploitable.

3.2.1 Sélection et normalisation des variables

Nous commençons par sélectionner les variables que nous allons ajouter à nos bases de travail. Si une information est redondante entre deux variables, nous conservons celle qui a le niveau de détail le plus fin.

De plus nous ne gardons que des variables sous forme de taux, car une quantité liée à une commune peut être grande simplement parce que la population dans cette commune est élevée. Pour enlever cet « effet population », il faut étudier la donnée non pas en absolue mais en relatif. Sur les 611 variables présentes initialement, il ne nous en reste que 66 à la fin de cette étape. Ci-dessous un exemple de variables après traitement à la maille départementale (*Figure 3.1*).

	Départements
variable	Libellé
Code_departement	Code_departement
TxNatalite	Taux de natalité 2016
EsperanceVieFemmes_60ans	Espérance de vie des femmes à 60 ans 2016
EsperanceVieFemmes_naiss	Espérance de vie des femmes à la naissance 2016
EsperanceVieHommes_60ans	Espérance de vie des hommes à 60 ans 2016
EsperanceVieHommes_naiss	Espérance de vie des hommes à la naissance 2016
TxBrutMortalite	Taux brut de mortalité 2016
RenouvellementPopulation	taux de natalite/taux de mortalité 2016
TxMortaliteInfantile	Taux de mortalité infantile 2014-2016
TxMortaliteStd_0a64ans	Taux de mortalité standardisé des 0 à 64 ans 2016
TxMortaliteStd_P65ans	Taux de mortalité standardisé des 65 ans ou + 2016
TxNuptialite	Taux de nuptialité 2015
TxDivorce	nbdivorces/population 2015
CreaIndustries	Part industrie dans les créations d'ent. 2016
CreaConstructions	Part construction dans les créations d'ent. 2016
CreaCmrceTranspHebergResto	Part commerce, transports, hébergement, restauration dans les créations d'ent. 2016
CreaServMarch	Part services marchands dans les créations d'ent. 2016
CreaServMarchPourMenage	Part services marchands auprès des ménages dans les créations d'ent. 2016
CreaServMarchPourEnt	Part services marchands auprès des entreprises dans les créations d'ent. 2016
TxChomage	Taux de chômage annuel moyen 2017
SalaireNetHoraire	Salaire net horaire moyen 2015
TxPauvrete	Taux de pauvreté 2015

FIGURE 3.1 – Variables départementales

Ce travail effectué, nous pouvons commencer à traiter les variables retenues, en vue de les ajouter par la suite à nos bases pour la modélisation.

3.2.2 Formatage des variables

Cette partie a pour but de traiter chacune des variables sélectionnées dans nos bases de modélisation. Nous souhaitons obtenir des modalités fines et régulières. L'intérêt sera d'opérer, au moment de la modélisation, des regroupements ou lissages sur la base du modélisé et non de l'observé. Ainsi, même si l'effet modélisé est un peu erratique au début, le lissage permet de capter une tendance moyenne non contrainte par des a priori en amont, qui n'auraient pas tenu compte de la séparation des effets entre variables effectuée par le GLM. Cette notion de finesse n'a de sens que pour des variables quantitatives (ou au moins ordonnées), ce traitement ne s'effectue donc que sur ce type de variables.

Pour pouvoir utiliser ces variables sous le logiciel de modélisation Emblem il nous faut tout d'abord créer des tranches et en faire un format sous SAS. En contrepartie de la finesse désirée pour les variables, il est pour autant nécessaire de ne pas conserver trop de modalités, car chacune doit avoir un volume suffisant pour avoir une modélisation pertinente. Nous créons ainsi des classes d'arrondis pour pouvoir exploiter l'information.

Les tranches se forment en choisissant un pas, une valeur minimale et une valeur maximale pour chaque variable. La valeur minimale correspond à un seuil dont on donne la valeur aux modalités étant en dessous

de celui-ci, la valeur maximale correspond à un seuil dont on donne la valeur aux modalités le dépassant et le pas correspond à l'écart entre les différentes valeurs possibles entre la valeur minimale et la valeur maximale. Ces trois valeurs sont déterminées avec comme indicateur les centiles des différentes variables. Il nous faut choisir un pas suffisamment petit pour garder le plus d'informations possibles tout en faisant attention à ne pas former de tranches vides. Le choix des valeurs minimales et maximales sert à regrouper les extrêmes. Ces tranches sont créées, non seulement sur les variables communales, mais aussi sur les variables départementales et régionales. Ces tranches régulières permettent des estimations cohérentes sous Emblem, car les variables numériques sont souvent approximées par des lissages polynomiaux.

Par exemple, la *Figure 3.2* ci-dessous représente les tranches formées dans le cas de la variable externe correspondant au taux de mortalité infantile.

TxMortaliteInfantile	Frequency	Percent	Cumulative Frequency	Cumulative Percent
999999	287868.7	14.4	287868.7	14.4
2.2	22408.54	1.12	310277.3	15.52
2.4	62265.86	3.11	372543.1	18.63
2.6	184216.9	9.21	556760.1	27.84
2.8	240757.8	12.04	797517.9	39.88
3	242239.3	12.11	1039757	52
3.2	382234.1	19.11	1421991	71.11
3.4	166138	8.31	1588129	79.42
3.6	113346.5	5.67	1701476	85.09
3.8	125457.8	6.27	1826934	91.36
4	66621.2	3.33	1893555	94.69
4.4	51126.38	2.56	1944681	97.25
4.6	31326.31	1.57	1976007	98.82
5	8447.237	0.42	1984455	99.24
5.2	15220.69	0.76	1999675	100

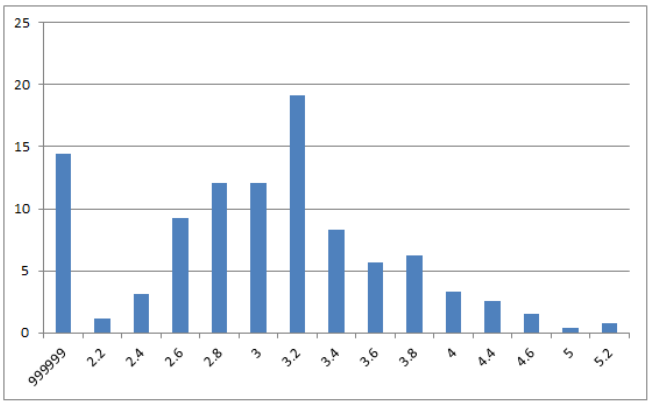

FIGURE 3.2 – Création de format sur le taux de mortalité infantile

Comme nous pouvons le constater, nous formons des tranches de valeur avec un espacement constant, cela dans le but d'y ajuster un polynôme interpolateur plus facilement. Mais nous gérons également les valeurs manquantes en attribuant une valeur inconnue (codifié 9999). Et enfin, nous ne formons pas forcément des classes égales en volume (méthode par quantiles), comme le représente l'histogramme construit sur la répartition de ces valeurs sur l'ensemble des communes françaises. Dans cet exemple nous avons choisi les paramètres suivants :

— Notre pas vaut 0.2
— Notre valeur seuil minimale vaut 2.2
— notre valeur seuil maximale vaut 5.2

Un traitement sur les inconnues est appliqué : nous attribuons la moyenne départementale aux lignes dont l'information des variables communales est manquante, ceci afin de perdre le moins d'informations possible. Nous appliquons ensuite un format aux variables externes pour pouvoir exploiter les tranches construites lors de la modélisation.

En ce qui concerne les variables contenant des lignes avec l'information qui reste manquante malgré le traitement à l'aide de la moyenne départementale, la valeur « 999999 » leur est attribuée pour pouvoir les différencier des autres modalités (cela peut arriver pour les données manquantes sur tout un département ou tout simplement pour des variables concernant l'échelle départementale ou régionale). En complément des variables internes et externes, une variable contenant aléatoirement des valeurs entre 0 et 9 est également créée. Elle permettra de séparer notre base en deux parties : l'une servant à l'apprentissage et l'autre à la validation de notre modèle.

La dernière étape du code permet de générer les bases de modélisation Emblem, en fréquence et en coûts moyens, pour les garanties responsabilité civile, bris de glace, vol et dommages.

3.3 Modélisation des fréquences et coûts moyens des garanties principales

A la suite de cette constitution de bases, nous pouvons entamer notre processus de modélisation. Nous effectuons des modèles linéaires généralisés sur les garanties principales via le logiciel Emblem.

Pour chaque garantie, nous modélisons la fréquence et le coût moyen à partir des bases générées précédemment. Nous faisons l'hypothèse que les fréquences suivent des lois de Poisson et les coûts moyens s'ajustent bien à des lois Gamma. Toutes ces hypothèses se vérifient par expérience en assurance auto, c'est pourquoi nous ne détaillerons pas les tests d'adéquations. En choisissant une fonction de lien log, pratique pour avoir une structure multiplicative des effets, nous mettons ainsi à jour des modèles "log-Poisson" et "log-Gamma".

Nous commençons par récupérer les modèles antérieurs, constitués uniquement des variables de risque. En se plaçant sur une base d'apprentissage qui représente 80% de notre base totale, nous utilisons une méthode forward pour actualiser les variables significatives présentes à notre modèle. La décision d'actualisation s'appuie sur le jugement de l'impact de la variable sur le modèle, des points de vue statistique et métier.

Ainsi nous commençons par observer la significativité des variables déjà présentes dans le modèle par des tests statistiques comme présentés dans la *Figure 3.3*. Ici nous pouvons conserver toutes les variables déjà présentes car celles-ci ont toutes une p-value inférieure à 5%, elles sont donc significatives.

Factor	Wald p Value (Chi-Sq)
RANNEE	0,0000%
RBOITEVIT	0,0189%
Marque	0,0000%
Carrosserie	0,0000%
Situation_matrimoniale	0,0000%
Usage	0,0000%
Classe_prix	0,0000%
Csp	0,0000%

FIGURE 3.3 – Significativité des variables de notre modèle

Pour avoir une meilleure connaissance de notre base, nous regardons également la matrice des corrélations, qui représente les corrélations entre les modalités des variables présentes dans le modèle. Cette étude nous évite d'y ajouter par la suite, deux variables corrélées qui pourraient rendre nos résultats instables.

C'est à ce moment que nous commençons à utiliser la méthode forward. Nous balayons les variables à tester afin de détecter des tendances significatives globalement, ou localement sur certains groupes de modalités. Si nous observons de telles tendances sur les variables non présentes dans notre modèle, alors nous les y ajoutons en lissant leur coefficient, et groupant les modalités qui nous semblent pertinentes. Le but des lissages est d'extraire la tendance de la variable en la séparant du bruit lié à la base de données. Cela permet d'améliorer la prédictivité du modèle sur d'autres jeux de données que la base d'apprentissage. Dans un souci d'interprétabilité, nous devons également nous assurer que la tendance observée est sensée, qu'elle respecte une certaine logique.

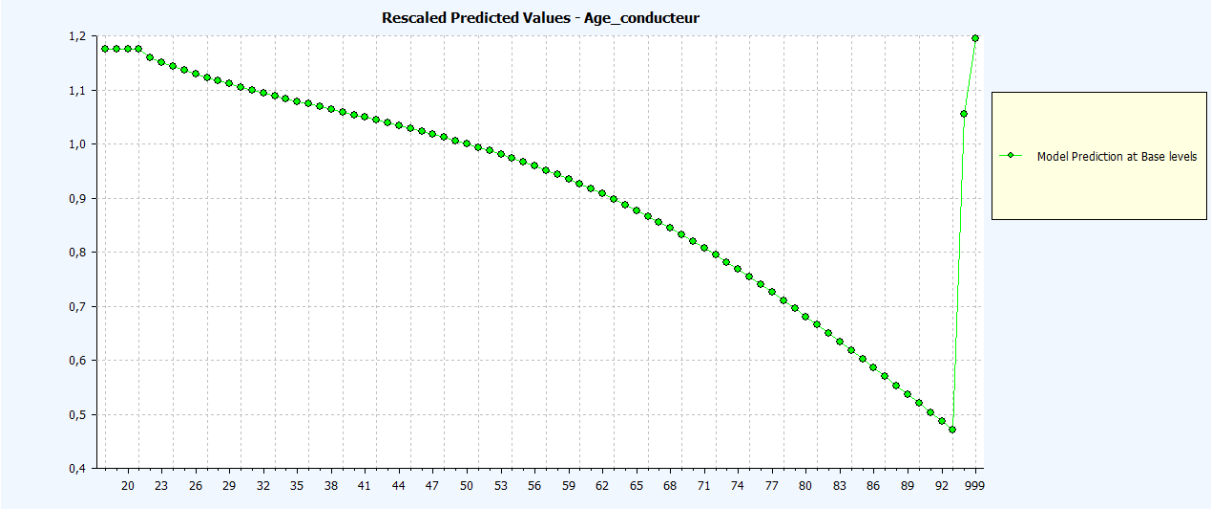

FIGURE 3.4 – Effet de la variable de l'âge du conducteur sur le modèle de fréquence du bris de glace

La *Figure 3.4* illustre cette démarche. Elle représente l'impact de l'âge du conducteur sur les fréquences de sinistres bris de glace. Nous y observons les coefficients calculés par le modèle. Nous pouvons y distinguer une tendance décroissante. Cette tendance est en accord avec une logique disant que plus le conducteur est jeune, plus il est susceptible d'avoir des accidents. Les personnes plus âgées et matures sont plus prudentes et sont donc moins impliquées dans des sinistres.

Sur cette figure, des lissages ont également été appliqués afin de ne modéliser autant que possible l'effet structurel de la variable séparé du bruit, comme expliqué précédemment. Des regroupements ont aussi été effectués sur les 18-21 ans.

Une fois le lissage et les groupements terminés, nous avons toujours la possibilité de nous assurer de la pertinence de l'ajout des variables en regardant le test de significativité du Chi2, ainsi que les indicateurs standards tels que la déviance, l'AIC, l'AIC corrigé ou le BIC. En ce qui concerne l'AIC et le BIC, il faut les consulter en les comparant avec ceux d'un autre modèle. Dans le cadre de la révision tarifaire, nous comparons ces grandeurs sur des modèles avec et sans ajout de la variable étudiée. Dans le cas où ces deux critères varient dans des sens contraires, nous privilégions le critère du BIC car il est plus pénalisant en cas de sur-paramétrisation du modèle que le critère AIC. Nous avons aussi la possibilité d'afficher les intervalles de confiance pour vérifier si la variable est significative.

Comme le montre la *Figure 3.5*, Emblem nous laisse la possibilité de comparer en détail différents modèles. Cela permet de prendre des décisions quant à l'ajout ou non de certaines variables. Ici nous avons ajouté la variable représentant la classe SRA, cela fait diminuer notre déviance, l'AIC, le BIC, l'AIC corrigé. Le modèle continue de converger et le Chi2 est à 0%. Nous pouvons donc ajouter la variable à notre modèle.

	Current Model	Reference Model	Difference
Model Description	Mean + RANNEE + Age_conducteur + Ancienneté_permis + RUSAGE + RCLSRA + Puissance_fiscale	Mean + RANNEE + Age_conducteur + Ancienneté_permis + RUSAGE	+ RCLSRA + Puissance_fiscale
Truncated Description	RANNEE + Age_conducteur + ...	RANNEE + Age_conducteur + ...	
Zero Weighted	0	0	0
Fixed or Simple Alias	1	1	0
Complex Alias	1	0	1
Fitted Parameters	204	158	46
Deviance	771.085,9	772.861,5	-1.775,605
Chi Squared Percentage		Sub-Model	0,0%
AIC	739.167,0	740.041,6	-874,5896
BIC	742.102,9	742.315,5	-212,5632
AICc	739.167,0	740.041,6	-874,587
Fitting Result	Converged OK	Converged OK	

FIGURE 3.5 – Comparaison de deux modèles Emblem avant et après ajout de variables

Nous contrôlons l'ajout des variables au regard des corrélations entre elles, et regardons attentivement dans quel mesure l'effet d'une variable a été modifiée lorsque nous avons ajouté une variable corrélée.

Ceci étant fait, nous poursuivons notre étude sur une base de validation composée des 20% de notre base

de départ. Cette base nous permet de nous assurer que les effets des variables explicatives restent stables sur une base différente de celle d'apprentissage. Lorsque nous constatons une discordance, nous cherchons à modifier les effets non généralisables. Plusieurs actions peuvent être envisagées, soit sur les lissages des variables en diminuant le degré du polynôme interpolé, soit en regroupant différemment, soit en enlevant totalement la variable du modèle en cas de tendance contraire notamment.

A la fin de cette étape nous obtenons le modèle suivant :

$$g(E[Y]) = \beta_{internes} X_{internes} + \beta_{externes} X_{externes} + \epsilon_{avant\ zone}$$

Avec :
- Y la variable à expliquer
- $X_{internes}$ les variables internes à l'entreprise
- $X_{externes}$ les variables récupérées sur les bases open data et retraitées
- $\epsilon_{avant\ zone}$ représente les résidus du modèle avant d'y intégrer un zonier
- $g(.)$ représente la fonction de lien

Ce modèle permet d'intégrer une partie de l'information géographique au moyen des variables externes. De cette manière, les effets des variables internes restent ainsi plus stables durant la modélisation, en particulier lorsqu'on remplacera les variables externes par le zonier.

3.4 Création d'un zonier

Toujours dans le but d'avoir un modèle le plus performant possible, nous exploitons les résidus des modélisations précédentes afin de construire un zonier par garantie. Ces zoniers ainsi créés seront réintégrés dans de nouveaux modèles. Pour cela nous nous aidons du logiciel Classifier.

Nous commençons par générer les bases en sélectionnant les contrats où le code commune est renseigné. Puisque nous ne disposons pas d'assurés dans la totalité des communes de France métropolitaine, nous devons générer de nouvelles lignes dans notre base de données. Cela nous permet d'inclure toutes les communes dans notre zonier. Pour ne pas biaiser notre analyse, nous accordons à ces lignes un poids très faible. En effet, le lissage s'effectuant par les communes adjacentes, les lignes factices hériteront, de cette manière, des caractéristiques de leurs voisines.

Pour contrôler la prédictivité du zonier, nous choisissons de minimiser ses erreurs sur la dernière année de survenance. C'est pourquoi le split apprentissage / validation est défini non pas de manière aléatoire, mais en fonction de l'année étudiée : l'année la plus récente sera notre base de validation et les autres constitueront notre base d'apprentissage.

La construction du zonier se fait à partir de résidus lissés. La génération de tels lissages utilise notamment la théorie bayésienne. L'ensemble de la théorie requise pour la construction de zonier sort du cadre de ce mémoire qui a pour objectif principal de greffer une méthode machine learning au processus de tarification. C'est pourquoi nous ne la détaillerons pas, et présentons simplement la logique de construction du zonier. Les résidus étudiés sont, par le biais du logiciel, issus de fréquence et coûts moyens normalisés pour écarter les différences de répartition de critères internes ou externes entre les communes : si certains assurés au risque interne ou externe plus élevé sont davantage présents dans une commune, on observe une variation de fréquence (ou coût moyen) non imputable à un critère géographique. Or on cherche justement cet effet géographique "pur" basé sur des communes comparables entre elles, d'où cette normalisation préalable. A partir de ces résidus, nous effectuons une série de lissages de différentes intensités, et sélectionnons le plus stable dans le temps : le critère d'optimalité est la minimisation de l'erreur quadratique sur la base de validation.

Une fois ce niveau de lissage optimal déterminé, nous l'appliquons à l'ensemble de notre base. Puis, pour obtenir l'effet géographique total, nous agrégeons ces résidus lissés avec les effets des variables externes. Le zonier est construit en prime pure pour presque la totalité des garanties. Il nous faut donc multiplier les effets géographiques construits sur les résidus de fréquence et ceux construits sur les résidus de coûts moyens, pour pouvoir obtenir l'effet géographique de prime pure.

La seule exception est la garantie bris de glace. Le zonier de cette garantie est construit à partir des résidus de fréquence seulement. En effet, la modélisation des coûts moyens de la garantie bris de glace est parasitée par la présence de forfaits réparations, ce qui la rend médiocre. Ainsi les résidus de mauvaise qualité que nous lisserions seraient eux aussi impactés par ce phénomène.

La dernière étape de création du zonier consiste à regrouper les effets géographiques totaux en zones homogènes. Ce clustering se fait en utilisant la Méthode de Ward, qui consiste à minimiser la variance intra-classes des différents groupes tout en maximisant leur variance inter-classes. Au final, nous nous retrouvons avec un zonier découpé en 15 à 20 zones selon les garanties.

Modélisation des garanties avec le zonier

Nous exploitons dans cette partie le zonier créé à l'étape précédente.

Nous reconstruisons nos différents modèles de fréquence et coûts moyens en y intégrant une nouvelle variable de zone à la place de nos variables externes.
Le modèle à la fin de cette étape peut être résumé de la manière suivante :

$$Y = \beta X_{internes} + \beta Zoniernouveau + \epsilon_{apres\ zone}$$

Nous ne rentrons pas dans tous les détails de performance dus à l'ajout du zonier, car même si l'information des variables externes a été augmentée, la logique d'explication géographique via une variable

zone était déjà intégrée dans le processus. Seuls quelques résultats illustratifs sont donc exposés ci-dessous.

La garantie Bris de glace n'intégrait jusqu'à présent pas de zonier, et on constate logiquement que son ajout permet une grande amélioration du modèle en fréquence :

- l'effet de la variable est net : croissance globale, et une échelle d'amplitude d'effet de 0.5 à 2.7 lorsqu'on passe de la zone la moins risquée (1) à la zone la plus risquée (20),
- la déviance, AIC et BIC diminuent respectivement de 6.05%, 6.04% et 0.43%.

La garantie Dommages tous accidents intégrait jusqu'à présent le zonier RC, dont le risque est proche puisque souvent co-sinistré avec la RC lorsque deux véhicules se percutent. En ajoutant un zonier dédié mis à jour, on constate là aussi une amélioration notable. Exemple du coût moyen Dommages tous accidents :

- l'effet de la variable est net : croissance globale, et une échelle d'amplitude d'effet de 0.6 à 1.81 lorsqu'on passe de la zone la moins risquée (1) à la zone la plus risquée (20),
- la déviance, AIC et BIC diminuent respectivement de 3.1%, 0.25% et 0.25%.

Nous avons maintenant fini la révision tarifaire classique. A ce stade, nous avons mis à jour nos modèles de fréquences et coûts moyens et avons actualisé nos différents zoniers. A l'issue de cette étape, nous obtenons à nouveau des résidus, et c'est sur ceux-ci que nous effectuerons dans le chapitre suivant un travail de Machine Learning pour créer une variable d'interaction, en vue d'augmenter encore la performance des modèles.

Chapitre 4

Ajout de la variable d'interaction

Les modèles précédents ne prennent en compte aucune interaction. Pour cause, leur détection est plus aisée en utilisant des méthodes d'arbres, dont chaque branche représente explicitement une interaction. L'organisation rationnelle de la modélisation présentée dans ce mémoire distingue plusieurs couches d'explications successives : d'abord les effets internes simples liés aux assurés en eux-mêmes (non détaillés car non travaillés spécifiquement dans ce mémoire) ; ensuite les effets géographiques liés à l'environnement des assurés ; enfin les effets composés (interactions entre variables, y compris le nouveau zonier), présentés dans ce chapitre.

4.1 Choix des résidus à modéliser

Dans un premier temps, il nous faut déterminer si nous effectuerons les algorithmes de machine learning sur les résidus de la fréquence ou de la prime pure pour chaque garantie. C'est pourquoi nous nous intéressons tout d'abord à la distribution de chaque type de résidu sur les différentes garanties.

Ci-dessous, les représentations graphiques des distributions de chaque résidu que nous avons centrées et réduites de manière à les observer dans des domaines comparables. En effet une fréquence est très faible comparée à la prime pure. Leurs résidus sont donc tout aussi différents.

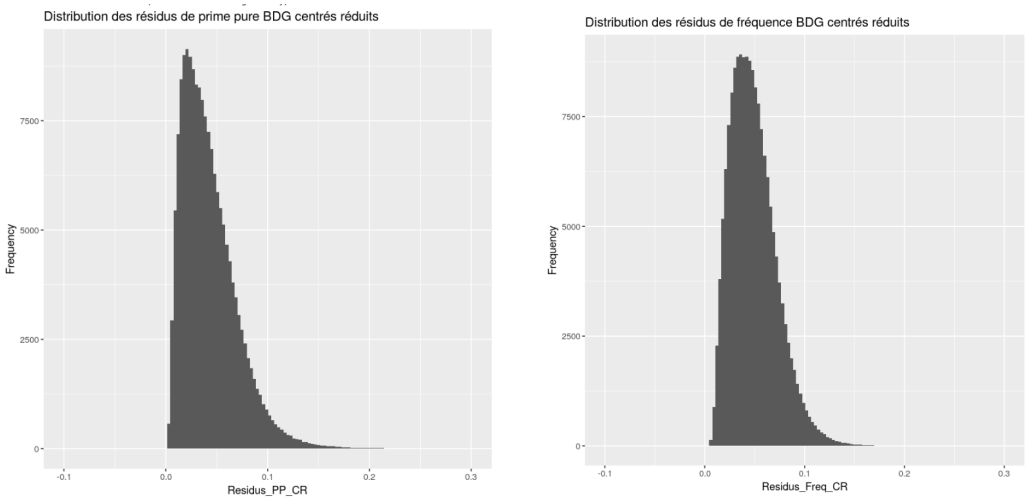

FIGURE 4.1 – Distribution des résidus Bris de Glace : à gauche prime pure, à droite fréquence

Nous commencerons notre travail sur la garantie bris de glace (BDG). Pour cette garantie nous disposons de plus de 18 millions de lignes avec une exposition totale de plus de 1 723 000, et une trentaine de variables sont ressorties comme discriminantes lors de la première étape de modélisation.Nous choisissons de travailler avec les résidus de fréquence. En effet, pour cette garantie notre zonier est basé uniquement sur la fréquence. En travaillant sur les résidus de prime pure directement, nous pourrions capter un effet lié à l'emplacement géographique qui n'a pas été pris en compte dans le zonier de cette garantie.

Dans un premier temps nous observons la distribution de nos résidus. Ceux-ci, une fois centrés et réduits, ont des formes similaires. De plus ils sont corrélés à 83% ce qui indique que nous pouvons fortement influencer les résidus de Prime Pure en expliquant les résidus de Fréquence, ce qui appuie notre première intuition quant au choix de la variable sur laquelle appuyer notre étude. Ces résidus de fréquence prennent en moyenne la valeur de -0,005, ils ont une médiane à 0,069. Leur valeur minimal est -365,98 et leur valeur maximale vaut 0,48.

$$Residus_{frquence} = frequence_{predite} - frequence_{observee}$$

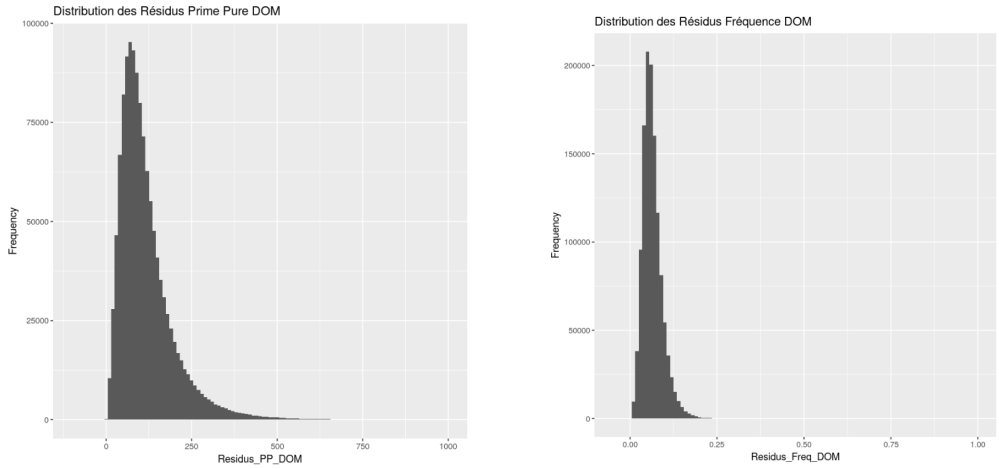

FIGURE 4.2 – Distribution des résidus Dommages : à gauche prime pure, à droite fréquence

Nous étudierons ensuite la garantie dommages tous accidents (DOM). Cette garantie rentre dans la formule la plus complète que Generali propose à ses assurés, c'est donc la base qui lui est associée qui comporte le moins de ligne et d'exposition. Nous disposons de 13 millions de lignes, une exposition totale de plus de 1 248 000, et d'une vingtaine de variables étant ressorties comme étant discriminantes lors de nos premières étapes de modélisation. Contrairement à la garantie « Bris de glace », nous choisissons cette fois-ci de travailler avec les résidus de prime pure. Le zonier de cette garantie est construit également en prime pure cela nous permet donc de rester cohérent dans notre démarche.

Nous commençons par étudier la distribution des résidus de cette garantie (*Figure 4.2*). Une fois centrés-réduits ceux –ci sont similaires. Cette étude ne réfute pas nos choix de travailler sur les résidus de prime pure. Ceux-ci ont pour moyenne -15, une médiane prenant la valeur 100, un minimum à -6 839 868, ce qui est un peu extrême et un maximum à 2 267.

Nous faisons donc apprendre nos algorithmes de machine learning sur un échantillon de notre base issue des modélisations de prime pure. La modélisation de la prime pure n'est autre qu'une combinaison par multiplication des modèles de fréquences et de coûts moyens.

$$Residus_{primepure} = PrimePure_{predite} - PrimePure_{observee}$$

4.2 Détermination du meilleur algorithme de machine learning par garantie

4.2.1 Principe

Nous voulons créer un critère synthétisant les interactions des variables de notre modèles. Les arbres simples sont très sensibles à la base de données, et sont donc difficilement généralisables. Nous préférons donc les appliquer sur des résidus "stabilisés" plutôt que sur les résidus bruts. Stabiliser -ou lisser- ces résidus va consister à les estimer par le biais de différents algorithmes, notamment de Machine Learning (par commodité on désignera l'ensemble de ces algorithmes de Machine Learning). La quantité résultante sera dès lors beaucoup moins dépendante de la base de données, puisqu'elle sera par construction un modèle prédictif. Nous cherchons donc à déterminer, dans un premier temps, le meilleur algorithme de Machine Learning pour chaque type de résidu. Pour cela nous devons tester différentes méthodes, en optimiser les paramètres (tuning), et enfin comparer, via différents indicateurs, les erreurs de ces algorithmes. Une fois cette étape accomplie, nous aurons, pour chaque garantie, un algorithme à appliquer à nos résidus permettant de les stabiliser, et donc par la même occasion, de créer dans la prochaine étape des arbres de régression plus stables.

Pour cela nous mettons en places les algorithmes suivants : Régression Elasticnet, arbre CART, RandomForest, Gradient Boosting. Nous utilisons le package "sparklyr" de R dans l'environnement Hadoop, afin d'uniformiser les syntaxes et les différentes méthodes de recherche de paramètres optimaux. Ce package a de plus l'avantage de faire du calcul distribué, ce qui nous permet de réduire notre temps de calcul. Ainsi, bien que les paramètres à optimiser diffèrent en fonction de l'algorithme de Machine Learning utilisé, cela facilite l'écriture et la lisibilité du code. Nous faisons tourner ensuite les algorithmes en parallèle dans un souci de performance temporelle et mettons en place une cross-validation.

Le package sparklyr n'a pas été notre premier choix. Nous avions initialement prévu de travailler avec le package "caret" et avions tout un script fonctionnant sur 10% de notre base. Cependant, en passant sur la base complète les traitements ont nécessité une grande quantité de mémoire et des temps de calculs démesurés. Nous avons donc été obligés de travailler avec "sparklyr" et Hadoop. Et bien que cette solution nous permette d'aboutir la plupart du temps à des résultats, des temps de calculs parfois très longs (bien que considérablement réduits) ne nous ont pas permis d'aller au bout de nos paramètrages.

L'exécution du code aboutit aux différents modèles spécifiés, tout en testant les différentes combinaisons de paramètres et compare les résultats des cross-validation en utilisant la mesure d'erreur choisie afin de sélectionner la combinaison de paramètres optimale. A la fin de cette étape nous avons la possibilité d'examiner les résultats de toutes les combinaisons de paramètres, de voir les paramètres sélectionnés, et également, de voir l'importance accordée aux différentes variables par l'algorithme choisi ainsi que des graphiques présentant la mesure de l'erreur au fil des tests de paramètres.

Il peut être judicieux, suivant le cadre de l'étude, de changer de mesure. Nous avons choisi d'optimiser nos modèles de la garantie bris de glace et dommages tous accidents en observant le RMSE (*Root Mean*

Square Error). Cette mesure est assez sensible aux écarts importants de prédictions, et peut donc accorder plus d'importance aux valeurs présentant de plus gros écarts (de par la présence du carré dans la moyenne). Nous aurions pu également choisir de travailler avec les MAE (*Mean Absolute Error*) qui considèrent tous les écarts avec le même poids. Ainsi un modèle globalement bon, peut avoir un RMSE plus mauvais dû à certains écarts considérés comme extrêmes qu'un modèle qui prédit moins bien mais sans écarts extrêmes, alors qu'un MAE a la logique inverse.

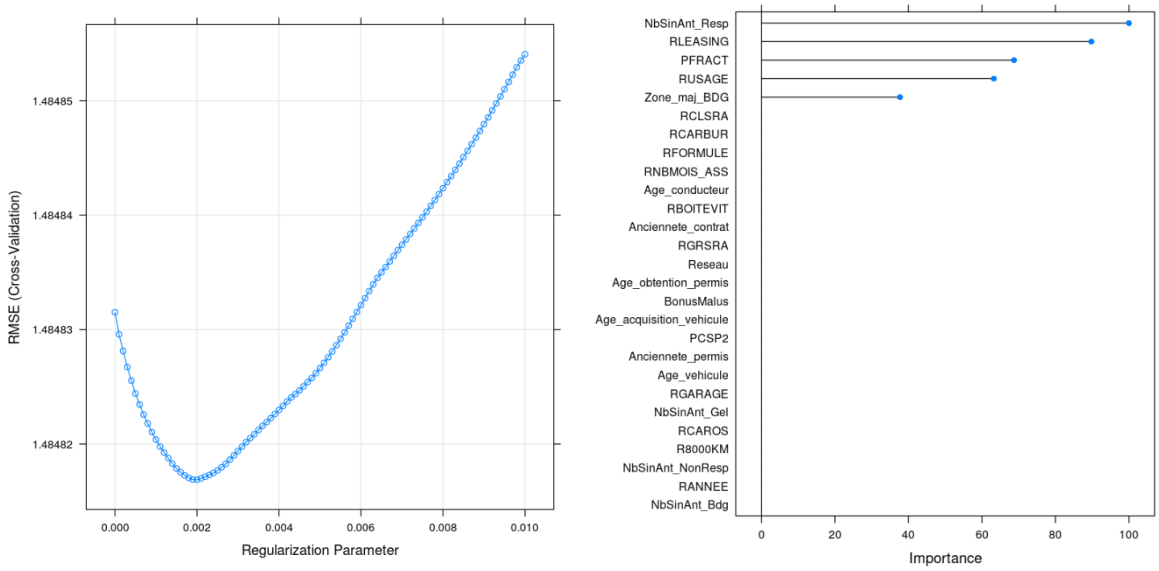

FIGURE 4.3 – Visualisation du fonctionnement d'un tuning et de l'importance des variables

La *Figure 4.3* est un exemple de visualisations de la variation de l'erreur qu'il est possible d'obtenir. Nous pouvons voir sur cet exemple, que l'algorithme choisit de n'accorder de l'importance qu'aux variables représentant le nombre de sinistres responsables antérieurs(NbSinAnt_Resp), le fait que les individus aient recours à du leasing ou non(RLEASING), la fréquence de paiement de la prime d'assurance(PFRACT), ainsi que le zonier Bris de glace(Zone_maj_BDG). Nous pouvons également constater l'importance du tuning des paramètres et de la validation croisée. En effet nous pouvons noter sur le graphique de gauche qu'augmenter le paramètre de régularisation permet dans un premier temps de capter de l'information significative (diminution de l'erreur), mais que passé un certain point l'erreur recommence à augmenter. Ceci est une bonne représentation du phénomène de sur-apprentissage : à partir d'un certain seuil l'algorithme sur-apprend sur les données de sa base d'apprentissage, ce qui mène à modéliser un bruit non reproductible sur une autre base de données. Il faut ainsi prendre garde à ne pas dépasser ce seuil pour rester prédictif et obtenir un modèle généralisable.

4.2.2 Garantie bris de glace

Rappelons que nous faisons apprendre nos algorithmes de machine learning sur un échantillon de notre base issue des modélisations de fréquence, contenant une variable avec les résidus de notre modèle. Nous travaillons avec une base d'apprentissage représentant 70% de la base totale et validons nos résultats avec les 30% restants.

Régression

Nous commençons par une régression. Nous choisissons de tuner : le paramètre de régularisation ainsi que le paramètre elasticnet. Ce dernier va nous permettre de réaliser notre régression elasticnet. Il peut prendre des valeurs comprises entre 0 et 1. Plus il sera proche de 1, plus notre régression sera proche d'une régression de Lasso et au contraire, plus il sera proche de 0, plus notre régression sera proche d'une régression de Ridge. Le paramètre de régularisation peut lui aussi prendre ses valeurs entre 0 et 1, il permet de contrôler l'impact de la pénalité. Ainsi pour le tuning de ces paramètres nous obtenons les résultats suivants :
De manière plus visuelle les résultats peuvent être présentés comme dans les Figures 4.5 et 4.6.

FIGURE 4.4 – Résultat du tuning de régression BDG

RMSE	elastic_net_param	reg_param
1.467730	0	0
1.467730	1	0
1.467730	0.5	0
1.467730	0.25	0
1.467730	0.75	0
1.467723	**0**	**1**
1.467731	1	1
1.467731	0.5	1
1.467731	0.25	1
1.467731	0.75	1
1.467725	0	0.5
1.467731	1	0.5
1.467731	0.5	0.5
1.467731	0.25	0.5
1.467731	0.75	0.5
1.467726	0	0.25
1.467731	1	0.25
1.467731	0.5	0.25
1.467731	0.25	0.25
1.467731	0.75	0.25
1.467723	0	0.75
1.467731	1	0.75
1.467731	0.5	0.75
1.467731	0.25	0.75
1.467731	0.75	0.75

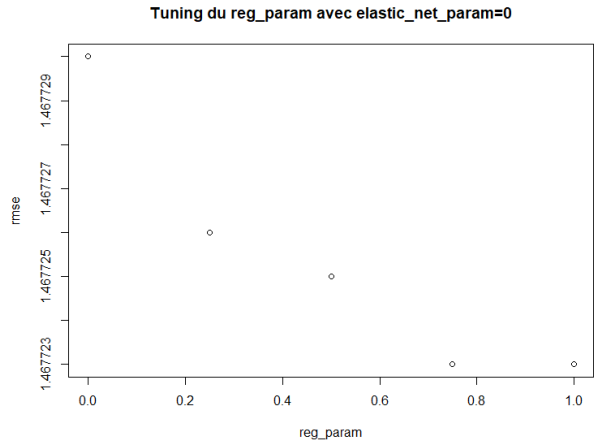

FIGURE 4.5 – Tuning de reg_pram avec elasticnet_param=1

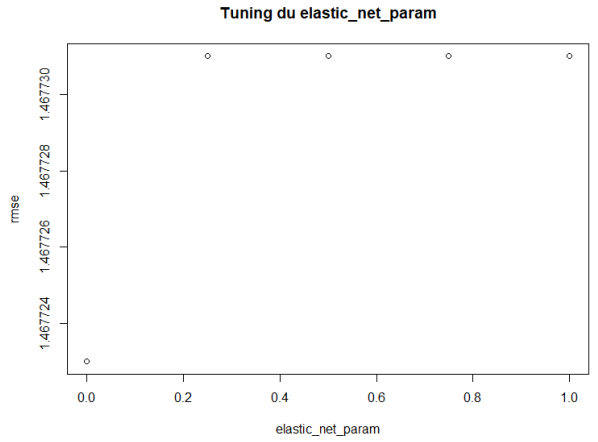

FIGURE 4.6 – Tuning de elasticnet_ param avec reg_pram=0

Comme nous pouvons le constater, l'algorithme sélectionne, suivant le critère des RMSE, un paramètre de régularisation à 1 et un paramètre elasticnet à 0. Il choisit donc une régression de Ridge. En ce qui concerne les deux Figures 4.5 et 4.6, le choix du paramètre de régularisation est assez évident. Il est facile d'observer jusqu'à quel moment, la variation de ce paramètre améliore notre algorithme. En effet nous pouvons observer une décroissance jusqu'à la valeur 0.75 puis une stabilité de ce RMSE en 1.

Avec un niveau de détail plus fin, il aurait peut-être été possible d'observer plus finement les hyper-paramètres optimaux, mais ajouter du détail dans le tuning allonge le temps de calcul et de travail. Le premier graphique a été obtenu en observant les RMSE avec un paramètre elastic net constant égal à 1 et le deuxième avec un paramètre de régularisation constant égal à 0.

Finalement pour la régression retenue nous obtenons les résultats suivants sur notre base de validation :
> RMSE = 1.58
> MAE = 0.15

Nous nous intéressons au RMSE, comme expliqué précédemment. Celui calculé sur notre base de validation est assez proche de celui de notre base d'entraînement (8% plus élevé, ce qui reste une écart acceptable). Un écart plus grand aurait été révélateur d'une situation de sur-apprentissage, que nous souhaitons éviter car alors le modèle ne serait pas généralisable à d'autres bases.

Arbre CART

Nous nous intéressons par la suite à la création d'un arbre de régression. Nous essayons toujours d'estimer les résidus de fréquence et mettons en entrée les mêmes variables que précédemment. Le but est de comparer la performance de différents algorithmes pour en sélectionner le meilleur. Pour la création de notre arbre nous décidons de tuner sa profondeur via le paramètre "max_depth". Nous testons les valeurs 5, 10, 20 et 30. Les résultats de ce tuning avec cross-validation sont les suivants :

rmse	max_depth
1.470896	**5**
1.531404	10
1.770683	20
1.949823	30

Le paramètre sélectionné est donc une profondeur à 5 cependant nous ne nous situons pas un minimum local comme le montre le graphique de la Figure 4.7. En effet nous observons toujours une tendance croissante et il n'y pas de changement de variations observé. Cela peut-être lié au manque de valeurs testées, et dans ce cas il y aurait un minimum entre deux valeurs testées (par exemple 5 et 10), cela peut également être lié à la fenêtre de notre tuning, le minimum pourrait donc peut-être se situer sur une profondeur plus petite que 5. Il peut donc être intéressant de refaire tourner un tuning en affinant l'intervalle sur lequel tuner notre paramètre de profondeur en testant par exemple les valeurs 2, 5 et 8.

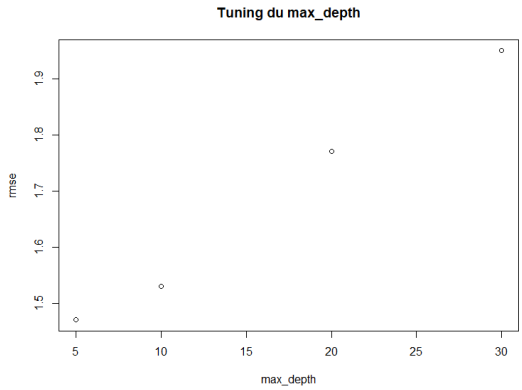

FIGURE 4.7 – Tuning Arbre

Implémenter un arbre de régression nous permet, au contraire de la régression, d'obtenir l'importance des variables dont voici une représentation :

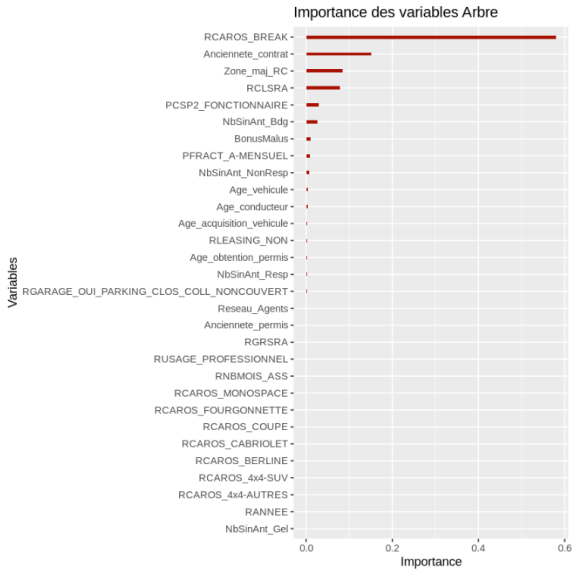

FIGURE 4.8 – Importance des variables sur l'arbre BDG

Nous pouvons constater que, ce qui joue le plus dans notre arbre est la carrosserie de type break, puis vient l'ancienneté du contrat et le zonier de la garantie responsabilité civile. Ainsi ce sont les trois plus importantes variables de notre arbre. Nous pourrons les comparer par la suite aux importances données aux variables par les algorithmes de Random Forest et de Gradient Boosting.

Sur une base de validation, l'arbre de régression nous donne les résultats suivants :

> RMSE = 1.58
> MAE = 0.15

En observant le RSME, celui calculé sur notre base de validation est assez proche de celui de notre base d'entraînement (8% plus élevé, ce qui reste une écart acceptable), pour considérer que nous ne sommes pas en situation de sur-apprentissage.

Random Forest

Passons maintenant à l'algorithme de Random Forest : avec des forêts composées de 500 arbres, nous tunons la profondeur de ces arbres en testant alors les valeurs 6, 8 et pour l'argument "max_depth". Dans un random forest, l'agrégation d'arbres indépendants permet d'en minimiser la variance. Pour privilégier des arbres plus indépendants les uns des autres, il nous faut prendre une profondeur assez faible mais suffisamment grande pour un pouvoir prédictif suffisamment bon, ce qui justifie le choix de cette fenêtre de tuning. Nous tunons également le nombre de variables testés dans les arbres de la forêt via le paramètre "feature_subset_strategy" qui fait varier ce nombre. Nous testons un tiers des variables avec la modalité "auto", et la racine carré du nombre de variables total par la modalité "sqrt". Nous avons les résultats suivants pour le tuning :

rmse	max_depth	feature_subset_strategy
1.467772	**6**	**sqrt**
1.468325	8	sqrt
1.470171	10	sqrt
1.468268	6	auto
1.470763	8	auto
1.475572	10	auto

La profondeur choisie est bien une profondeur de 6 avec une sélection de variables lors de la construction de la forêt suivant la méthode sqrt donc parmi la raciné carré du nombre total de variables. Graphiquement le tuning du Random Forest nous donne la chose suivante :

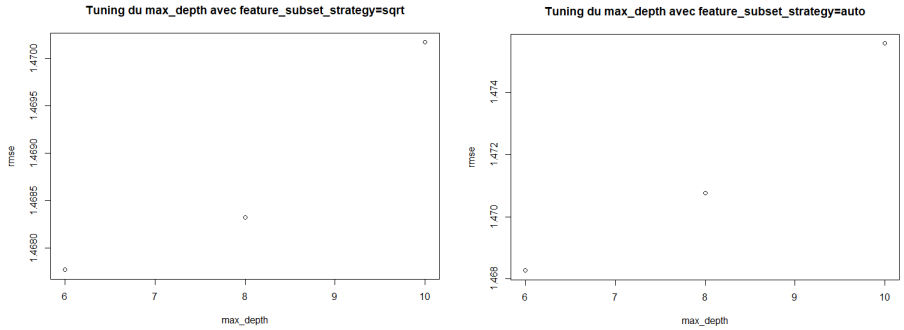

FIGURE 4.9 – Tuning Random Forest BDG

Comme pour l'arbre plus haut, le choix de l'hyper-paramètre sur la profondeur maximale ici n'est peut-être pas le meilleur. En effet nous disposons de peu de points pour visualiser où se situe le minimum. Néanmoins, nous pouvons supposer que la profondeur minimisant les RMSE n'est pas loin de la profondeur choisie car nous observons des variations plus faibles de RMSE dans ce voisinage-là.

En ce qui concerne l'importance des variables nous obtenons la *Figure 4.10*. Nous remarquons que nous avons l'âge du conducteur qui apparaît comme le facteur le plus important pour expliquer nos résidus ainsi que l'ancienneté du permis en seconde position. Les écarts d'importance entre les différentes variables sont moindres comparés aux importances des arbres. Nous avons également de fortes variations de ce classement par rapport à celui de l'arbre.

Sur une base de validation, le Random Forest nous donne les résultats suivants :

- RMSE = 1.58
- MAE = 0.15

En observant le RSME, celui calculé sur notre base de validation est assez proche de celui notre base d'entraînement (8% plus élevé, ce qui reste une écart acceptable), pour considérer que ne ne sommes pas en situation de sur-apprentissage.

FIGURE 4.10 – Importance des variables du Random Forest BDG

Gradient Boosting

Et pour finir avec cette étude des différents algorithmes, nous travaillons avec le Gradient Boosting. Pour cela nous décidons de tuner la profondeur des arbres avec le paramètre "max_depth", le taux d'apprentissage à chaque étape avec le paramètre "step_size" ainsi que le nombre d'itérations maximal via le paramètre "max_iter".

Nous testons les valeurs 20, 50 et 100 pour la profondeur pour tester un assez grand éventail de valeurs, les valeurs 2, 4 et 6 pour le taux d'apprentissage pour couvrir un maximum de possibilités dans le domaine du raisonnable pour ce paramètre qui ne doit pas être trop grand, et 20, 50 et 100 pour le nombre d'itérations toujours pour tester un large panel de valeurs. Cela nous donne les résultats du tableau de la Figure 4.11.

FIGURE 4.11 – Résultats tuning Gradient boosting BDG

rmse	max_iter	step_size	max_depth
1.467723	20	0.10	2
1.469292	20	0.10	4
1.479425	20	0.10	6
1.467720	**20**	**0.01**	**2**
1.469127	20	0.01	4
1.476096	20	0.01	6
1.467721	20	0.05	2
1.469189	20	0.05	4
1.476689	20	0.05	6
1.467731	50	0.10	2
1.470134	50	0.10	4
1.489274	50	0.10	6
1.467721	50	0.01	2
1.469144	50	0.01	4
1.476271	50	0.01	6
1.467723	50	0.05	2
1.469266	50	0.05	4
1.478319	50	0.05	6
1.467749	100	0.10	2
1.471266	100	0.10	4
1.497025	100	0.10	6
1.467720	100	0.01	2
1.469151	100	0.01	4
1.476597	100	0.01	6
1.467730	100	0.05	2
1.469869	100	0.05	4
1.482562	100	0.05	6

Nous retenons au final les hyper-paramètres 20 pour le nombre d'itérations maximal, 0.01 pour le taux d'apprentissage et 2 pour la profondeur maximale. Graphiquement le tuning du gradient boosting que nous avons implémenté nous donne la *Figure 4.12*.

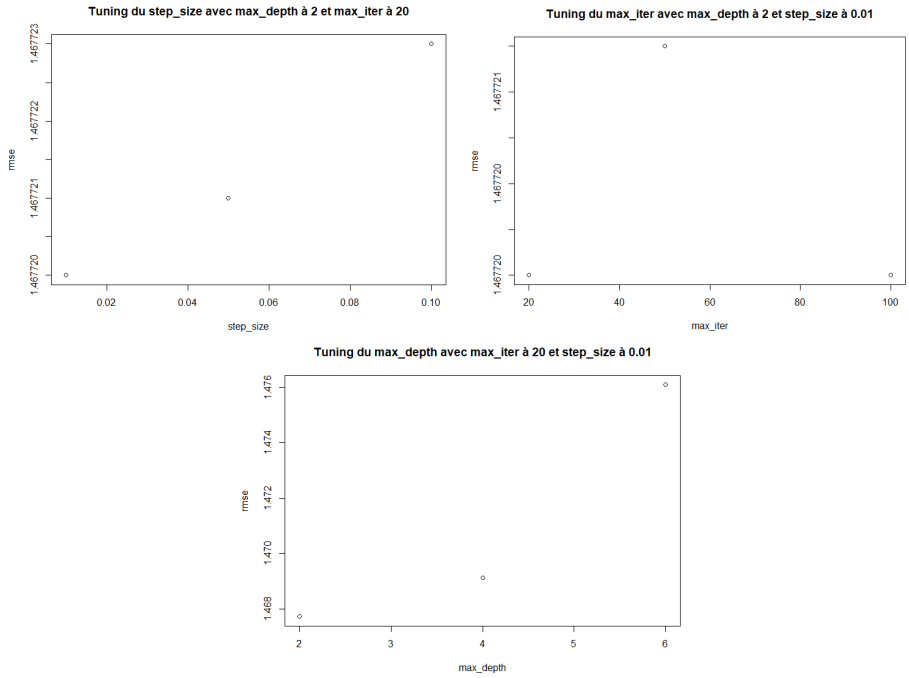

FIGURE 4.12 – Tuning du gradient boosting BDG

Comme pour tous les algorithmes testés jusqu'à présent, la contrainte temporelle et logicielle ne nous a pas permis de pousser notre étude jusqu'à un tuning totalement optimal. Néanmoins, sur le peu de valeurs testées, par exemple sur la profondeur maximale, les variations de RMSE sont amoindries au voisinage du paramètre que nous avons sélectionné. On peut donc supposer que nous avons fait un choix proche du meilleur pour cet hyper-paramètre.
Sur notre base de validation nous avons les mesures d'erreur suivantes :

- RMSE=1.57
- MAE = 0.15

En observant le RSME, celui calculé sur notre base de validation est assez proche de celui notre base d'entraînement (8% plus élevé, ce qui reste une écart acceptable), pour considérer que ne ne sommes pas en situation de sur-apprentissage.

Pour les importances des variables nous obtenons les résultats de la *Figure 4.13*.

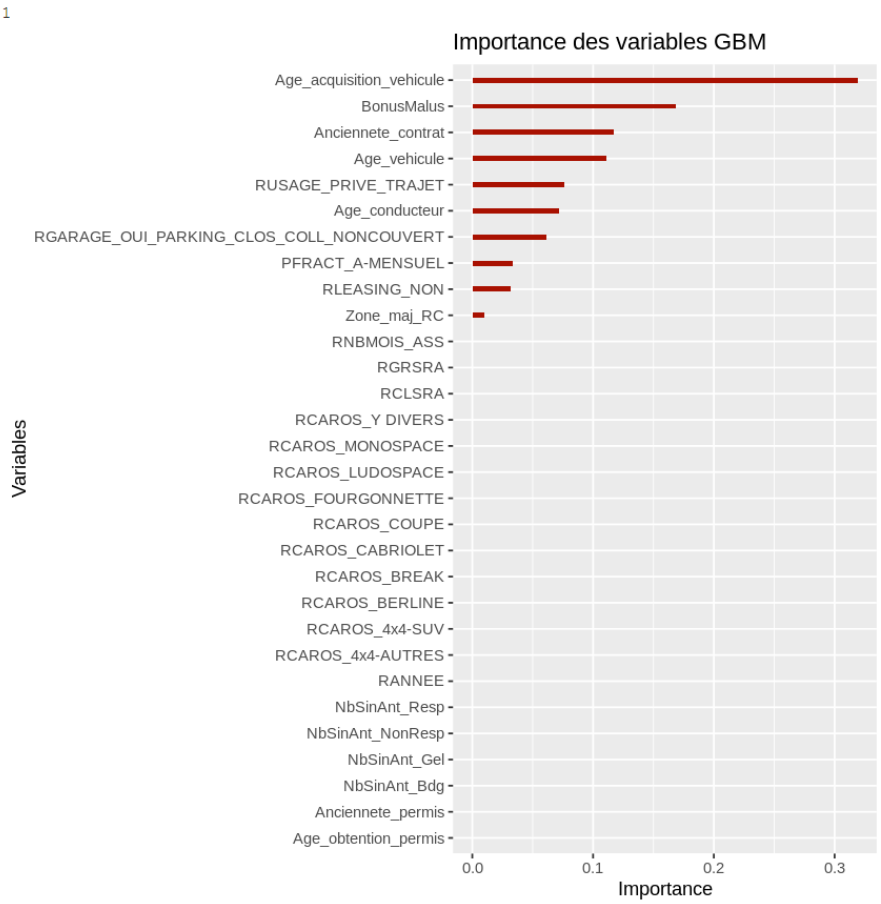

FIGURE 4.13 – Importance des variables du gradient boosting BDG

Nous constatons que les variables qui ressortent sont l'âge d'acquisition, le coefficient de réduction majoration et l'ancienneté du contrat. Nous pouvons conclure que d'un type de machine learning à un autre sur la garantie bris de glace, les variables considérées comme les plus importantes diffèrent grandement. Le choix de la méthode retenue va donc être déterminant pour la suite de notre étude car chaque algorithme va nous permettre de segmenter notre base d'une manière différente.

Après avoir fait tourner les différents Machine Learning sur les résidus de cette garantie nous obtenons les résultats suivants :

BDG	Régression	arbre	Forêt Aléatoire	Gradient boosting
RMSE	1.575830	1.5767592	1.5758395	1.5758273
MAE	0.152613	0.1525758	0.1526297	0.1526028

Ainsi, par mesure de performance, nous avons la chose suivante :

— Le meilleur MAE est celui de l'arbre CART avec un MAE de 0.1525758
— Le meilleur RMSE est celui du gradient boosting avec un RMSE de 1.5758273

En s'intéressant aux valeurs en tant que telles que prennent les erreurs, nous pouvons constater que les résultats ne sont pas très bons. Les MAE par exemple nous indiquent que nous faisons en moyenne une erreur (en valeur absolue) de la même grandeur que la quantité que nous modélisons. Cependant nous conservons les modèles car la finalité de notre étude est d'observer l'impact d'une variable d'interaction, ce qui ne sera mesurable qu'après une modélisation finale de la fréquence et des coûts moyens des garanties par des modèles linéaires généralisés.

Le RMSE est une mesure de l'erreur qui permet de prêter attention aux valeurs très élevées. Plus cet indicateur est grand, moins le modèle est bon. Le MAE au contraire des RMSE, accorde le même poids à chacune des observations. Cet indicateur est donc moins sévère en termes de pénalisation des erreurs importantes. Ainsi, dans le cadre de notre problème, nous décidons de pénaliser les grosses erreurs et choisissons de nous baser sur les RMSE afin de déterminer le modèle sur lequel nous appuyer pour la suite de nos études. L'algorithme retenu pour cette garantie est donc celui du gradient boosting.

4.2.3 Garantie dommages tous accidents

Rappelons que nous faisons apprendre nos algorithmes de machine learning sur un échantillon de notre base contenant une variable avec les résidus de prime pure de notre modèle. Nous travaillons avec une base d'apprentissage représentant 70% de la base totale et validons nos résultats avec les 30% restants.

Régression

Nous commençons par effectuer une régression elasticnet. Nous devons tuner les paramètres de régularisation ainsi que le paramètre elasticnet. Rappelons que ce dernier paramètre peut prendre des valeurs entre 0 et 1, ce qui permet de faire varier notre régression entre une régression de Ridge et de Lasso. Le paramètre de régularisation va moduler la pénalisation de la régression et peut également prendre des valeurs entre 0 et 1.

Nous obtenons le tuning suivant :

RMSE	elastic_net_param	reg_param
5016.373	0.00	0.00
5016.373	1.00	0.00
5016.373	0.50	0.00
5016.373	0.25	0.00
5016.373	0.75	0.00
5016.373	0.00	1.00
5016.311	**1.00**	**1.00**
5016.335	0.50	1.00
5016.335	0.25	1.00
5016.321	0.75	1.00
5016.373	0.00	0.50
5016.335	1.00	0.50
5016.335	0.50	0.50
5016.363	0.25	0.50
5016.342	0.75	0.50
5016.373	0.00	0.25
5016.355	1.00	0.25
5016.363	0.50	0.25
5016.368	0.25	0.25
5016.358	0.75	0.25
5016.373	0.00	0.75
5016.321	1.00	0.75
5016.342	0.50	0.75
5016.355	0.25	0.75
5016.330	0.75	0.75

De manière plus visuelle, nous avons les variations de RMSE des *Figures 4.14 et 4.15* en fixant dans la première figure le paramètre de régularisation à 1 et dans la seconde le paramètre elasticnet à 1.

L'algorithme finit par sélectionner, avec le RMSE comme critère, un paramètre elasticnet à 1 et un paramètre de régularisation à 1 aussi. Nous effectuons donc une régression de Lasso. Avec ces paramètres-là nous obtenons les résultats suivants sur notre base de validation :

> RMSE = 3708.8
> MAE = 254.39

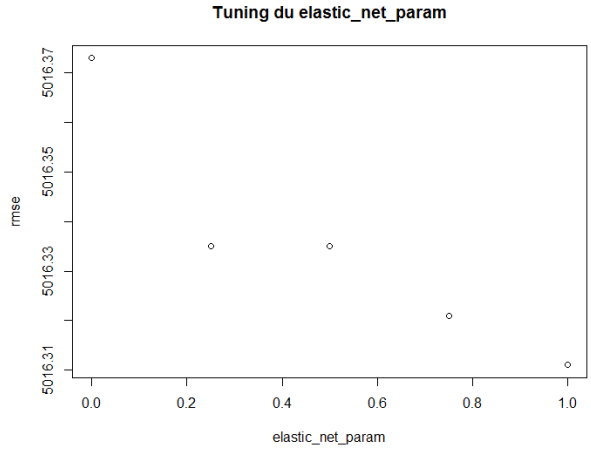

FIGURE 4.14 – Tuning du paramètre elasticnet DOM

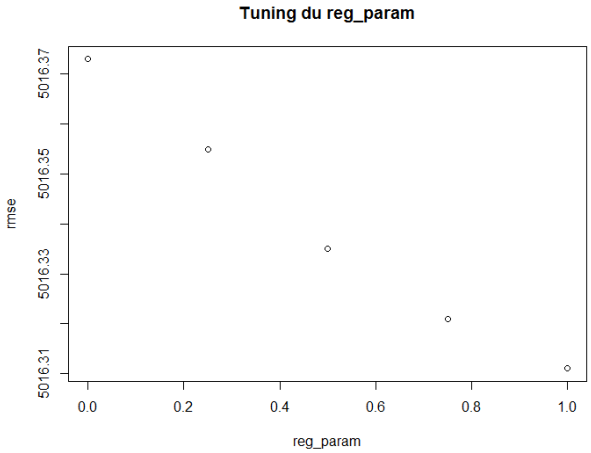

FIGURE 4.15 – Tuning du paramètre de régularisation DOM

Arbre CART

Nous testons par la suite un algorithme d'arbre CART de régression. Nous utilisons toujours les mêmes variables pour estimer notre résidus de prime pure et avoir des résultats comparables d'une méthode à une autre. Nous tunons la profondeur maximale de nos arbres via l'hyper-paramètre « max_depth » et testons les valeurs 5, 10, 20, 30 et nous obtenons les résultats suivants :

RMSE	max_depth
6582.848	**5**
7237.697	10
8507.707	20
8671.140	30

Nous sélectionnons donc une profondeur de 5. De par la tendance toujours croissante du RMSE sur les valeurs testées nous ne pouvons pas conclure sur le minimum. En effet il pourrait très bien se situer n'importe où entre les profondeurs 1 et 10. Nous conservons néanmoins la valeur 5 car comme indiqué précédemment, les traitements sont trop lents pour pouvoir effectuer une nouveau tuning entre 1 et 10. Une visualisation du tuning se trouve sur la *Figure 4.16*.

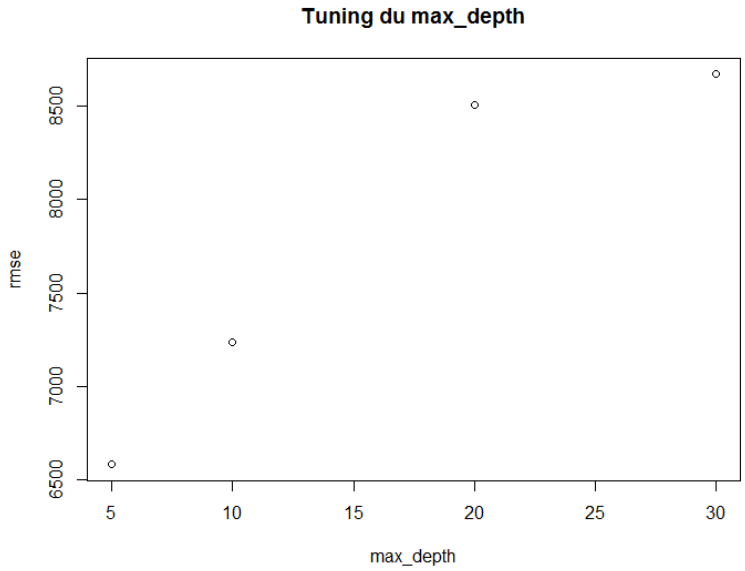

FIGURE 4.16 – Tuning du paramètre de profondeur maximale DOM

Cet arbre nous permet également d'avoir une première visualisation de l'importance des variables pour la modélisation par cet algorithme spécifiquement sur la *Figure 4.17*.

Donc pour un arbre avec une profondeur maximale de 5, les variables les plus significatives pour notre arbre sont :

- L'âge du conducteur
- Le coefficient de réduction majoration
- La catégorie socio-professionnelle « Artisan »
- L'ancienneté de permis

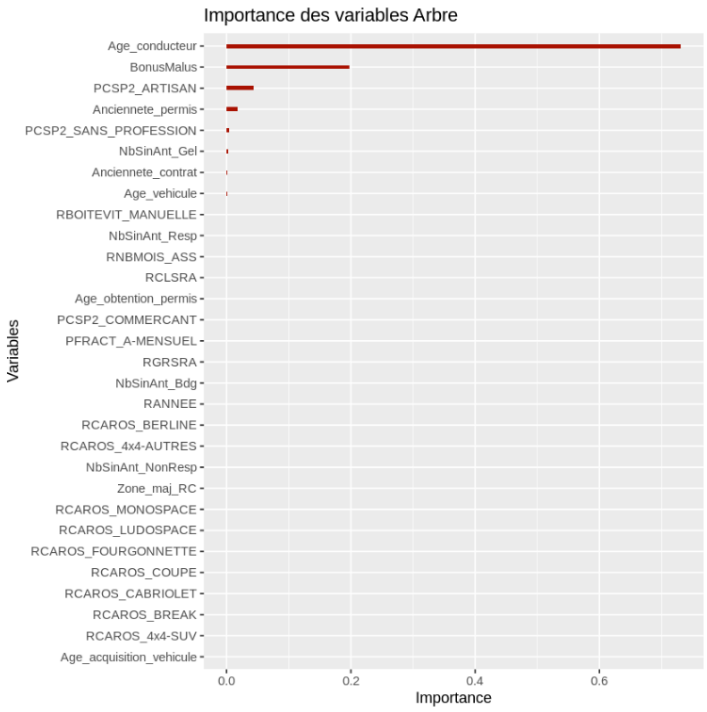

FIGURE 4.17 – Importance des variables arbre de régression DOM

L'importance de l'âge du conducteur est bien supérieure aux autres variables (on remarque un gros saut, très visible sur notre graphique), et capte le plus gros effet de notre arbre de régression. Nous pourrons comparer ces résultats par la suite à ceux issus de gradient boosting et de random forest.

Sur notre base de validation nous obtenons les mesures d'erreur suivantes pour une profondeur d'arbre de 5 :

- RMSE = 3707.66
- MAE = 251.31

Random Forest

Nous nous intéressons maintenant à un algorithme de random forest. Nous choisissons de tuner la profondeur des arbres en fixant la taille de la forêt à 500 arbres. Nous faisons varier le paramètre de profondeur (max_depth) ainsi que feature_subset_strategy qui correspond au nombre de variables considérées à chaque embranchement des arbres de la forêt. La modalité « auto » correspond, dans le cas d'une forêt comportant plus d'un arbre de régression, à un tiers des variables en input et la modalité « sqrt » correspond à la racine carré du nombre de variables totales.

RMSE	max_depth	feature_subset_strategy
5025.683	**6**	**sqrt**
5050.019	8	sqrt
5082.594	10	sqrt
5121.207	6	auto
5178.498	8	auto
5293.034	10	auto

Le paramètrage retenu sélectionne la racine carrée des variables pour les embranchements d'arbres d'une profondeur de 6. Plus visuellement nous avons le tuning suivant :

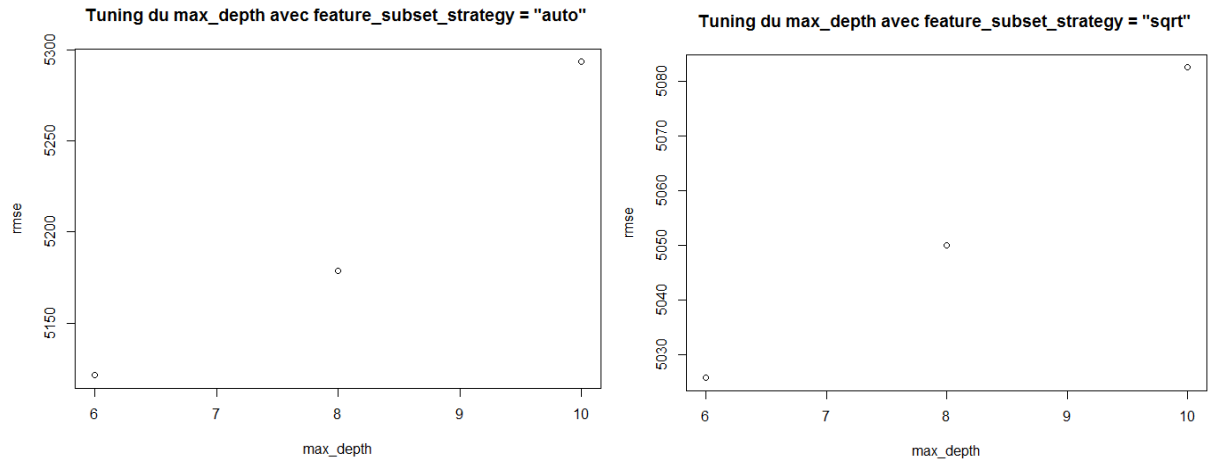

FIGURE 4.18 – Tuning du random forest DOM

Nous pouvons avoir la même réflexion que lorsque nous tunions notre arbre de régression. Le minimum de RMSE que nous obtenons en tunant la profondeur des arbres peut être remis en question quelle que soit la méthode de sélection des variables. Cependant l'écart entre une profondeur de 6 et une profondeur de 8 est plus faible que celle entre la profondeur de 8 et la profondeur de 10 donc on peut supposer que notre minimum se situe dans le voisinage d'une profondeur de 6. En ce qui concerne l'importance des variables nous observons la *Figure 4.19*.

Ainsi, l'algorithme de random forest considère les variables suivantes comme les plus discriminantes :

- Coefficient de réduction majoration
- L'âge du conducteur
- La classe SRA
- L'âge du véhicule
- L'ancienneté du contrat

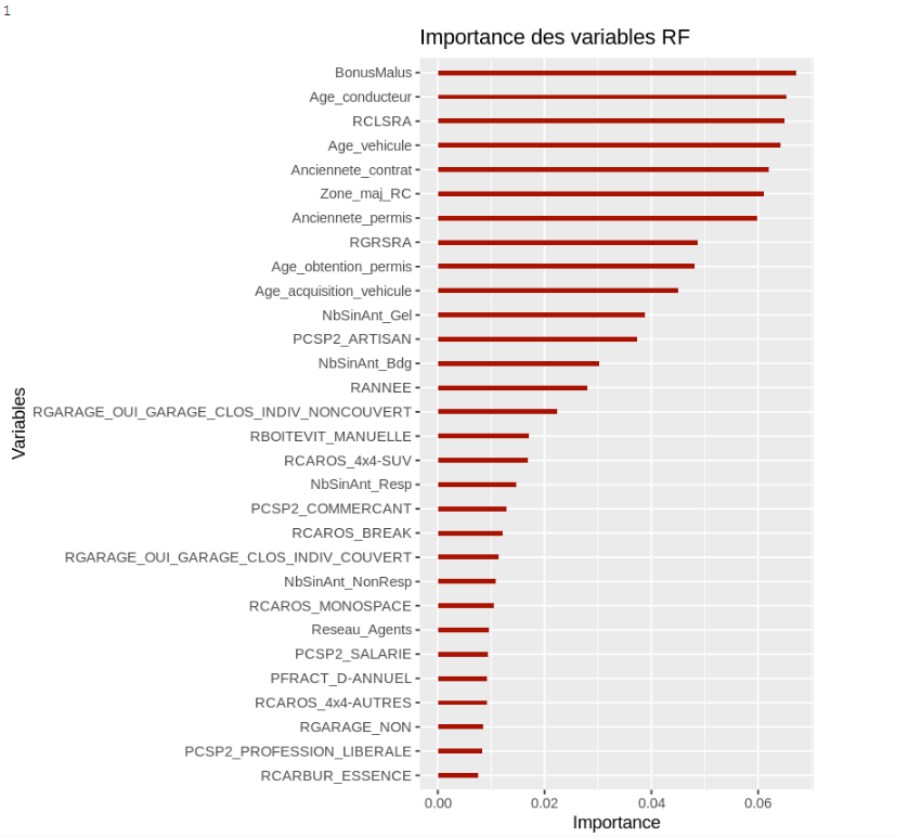

FIGURE 4.19 – Importance des variables du random forest sur la garantie DOM

Au-delà de ces variables nous observons des sauts d'importance entre les variables. Ainsi le random forest accorde de l'importance à plus de variables de notre base de modélisation avec des écarts d'importance plus modérés entre les variables par rapport à un arbre simple. Nous pouvons néanmoins noter que les variables les plus importantes de notre arbre de régression précédent se retrouvent également dans les variables les plus importantes de notre forêt avec quelques variations quant à leur position dans ce classement (échange entre le coefficient de réduction majoration et l'âge du conducteur par exemple).

Finalement, sur une base de validation nous obtenons les mesures d'erreur suivantes :

- RMSE = 3704.63
- MAE = 252.72

Gradient boosting

Le dernier algorithme que nous testons est l'algorithme de gradient boosting. Nous tunons la profondeur des arbres (max_depth), le taux d'apprentissage (step_size) et le nombre d'itérations maximal (max_iter) ce qui nous donne les résultats suivants :

RMSE	max_iter	step_size	max_depth
5017.954	20	0.10	2
5458.899	20	0.10	4
6817.855	20	0.10	6
5018.411	20	0.01	2
5512.707	20	0.01	4
6703.033	20	0.01	6
5017.996	20	0.05	2
5487.072	20	0.05	4
6760.662	20	0.05	6
5018.184	50	0.10	2
5490.068	50	0.10	4
6907.658	50	0.10	6
5018.088	50	0.01	2
5490.486	50	0.01	4
6714.153	50	0.01	6
5017.988	50	0.05	2
5474.986	50	0.05	4
6799.655	50	0.05	6
5018.695	100	0.10	2
5556.016	100	0.10	4
6997.289	100	0.10	6
5017.930	**100**	**0.01**	**2**
5482.085	100	0.01	4
6731.090	100	0.01	6
5018.070	100	0.05	2
5489.371	100	0.05	4
6878.490	100	0.05	6

Nous sélectionnons les paramètres suivants :

- Profondeur maximale 2
- Taux d'apprentissage : 0.01
- Nombre d'itérations max : 100

Visuellement, le tuning de ces hyper-paramètres nous est représenté de la manière suivante :

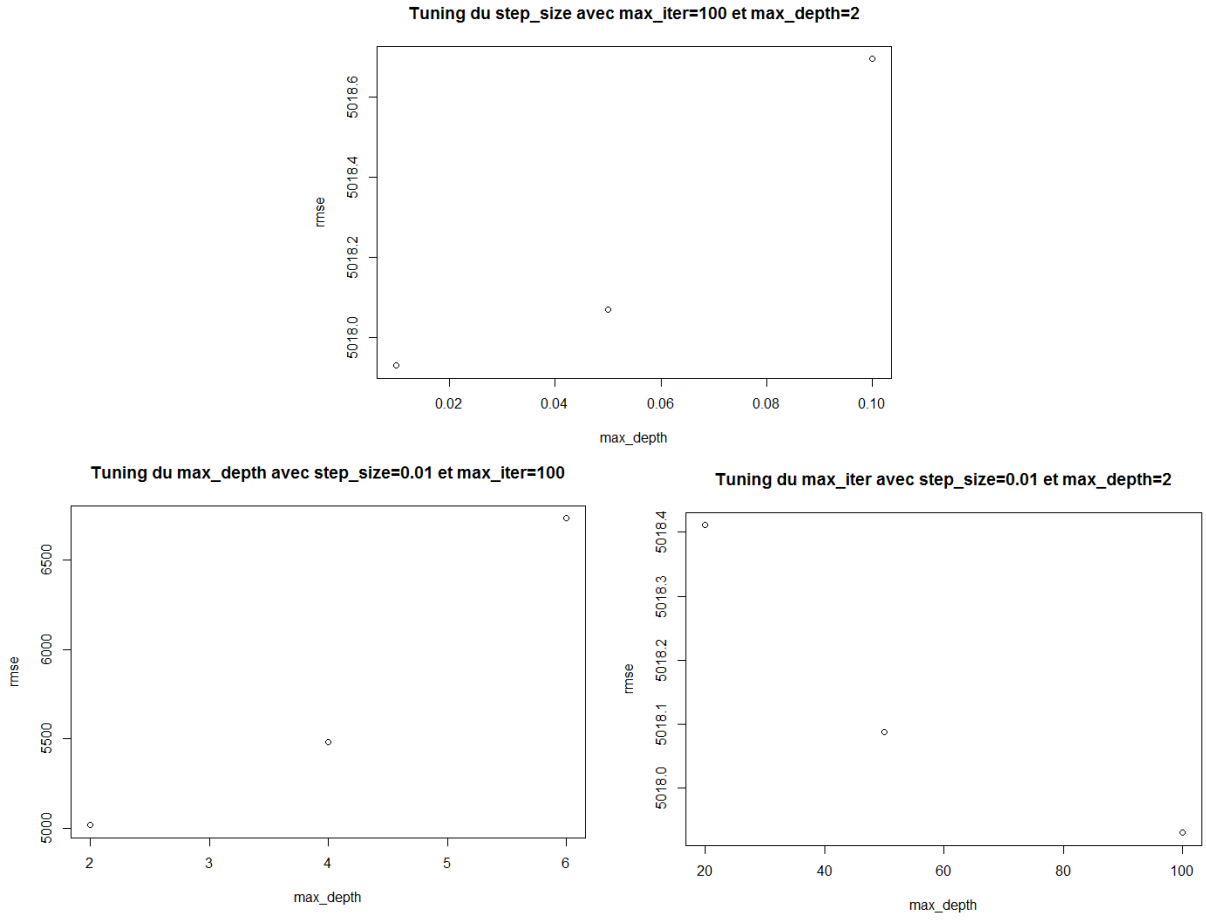

FIGURE 4.20 – Tuning du gradient boosting sur la garantie DOM

Comme précédemment, nous aurions pu affiner notre tuning après ces premiers résultats. Néanmoins le temps de calcul très long ne nous a pas permis d'aller plus loin dans notre étude. Avec ces paramètres-là, nous observons les importances de variables de la *Figure 4.21*

Nous observons que l'algorithme conserve moins de variables que notre random forest (décroissance de l'importance plus marquée) et plus qu'un arbre simple.

Nous notons comme variables étant les plus importantes :

- Le nombre de sinistres antérieurs gel
- La classe SRA
- Le groupe SRA

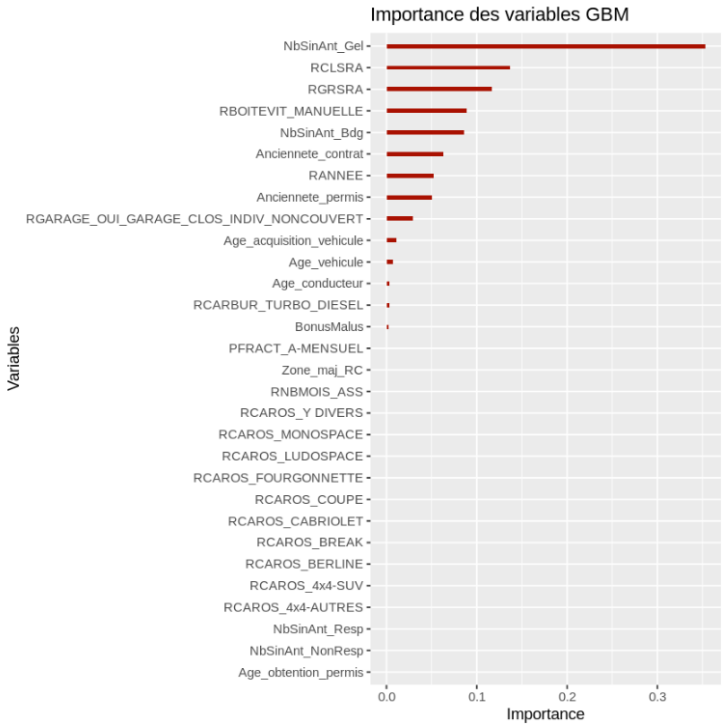

FIGURE 4.21 – Importance des variables avec gradient boosting sur la garantie DOM

Nous constatons que les variables retenues diffèrent grandement de celles retenues précédemment. L'âge du conducteur principal par exemple se situe ici avec une importance très négligeable comparée au nombre de sinistres antérieurs GEL qui, elle, a une importance prépondérante. Bien que présente dans les variables importantes des algorithmes précédents, le nombre de sinistres antérieurs GEL ne faisait pas partie des variables remarquables.

Nous obtenons donc avec cet algorithme des résultats bien différents des algorithmes précédents, avec une segmentation de notre base s'appuyant sur d'autres variables.

Finalement sur notre échantillon de validation nous obtenons les mesures d'erreurs suivantes :

> RMSE = 3704.69
> MAE = 253.77

Pour résumer, nous obtenons au total les résultats suivants :

DOM	Régression elasticnet	Arbre CART	Random Forest	Gradient Boosting
RMSE	3703.80	3707.66	3704.63	3704.69
MAE	254.39	251.31	252.72	253.77

Ainsi l'algorithme le plus performant en termes de RMSE est la régression elasticnet et en termes de MAE est l'arbre CART. Nous choisissons de nous baser sur l'indicateur RMSE car il est plus pénalisant sur les écarts importants que l'indicateur du MAE. Nous retenons alors la régression elasticnet qui a pour paramètres :

- Paramètre elasticnet = 1
- Paramètre de régularisation = 1

Comme pour la garantie précédente, en interprétant les valeurs des mesures d'erreur, nous remarquons que les modèles de machine learning ne sont pas très bons. Mais rappelons que la finalité de l'étude est d'observer l'impact d'une variable d'interaction, ce qui ne sera mesurable qu'après une modélisation finale de la fréquence et des coûts moyens des garanties par des modèles linéaires généralisés.

Une autre remarque importante est que nos algorithmes commettent des erreurs, en termes de RSME, 30% plus faibles sur notre base de test que sur notre base d'apprentissage. Cette remarque concerne l'ensemble des algorithmes testés pour prédire les résidus de prime pure de la garantie dommages tous accidents. Nous ne pouvons donc pas savoir si nous sommes en situation de sur-apprentissage ou de sous-apprentissage. Plusieurs raisons à ce phénomène sont envisageables. Une première piste potentielle est la séparation de notre base totale en base d'apprentissage et base de validation, dans notre cas la base de validation serait plus simple à prédire car elle contiendrait moins de valeurs aberrantes. Un découpage différent de notre base corrigerait alors ce souci. Nous pouvons également envisager de changer le paramétrage de nos algorithmes pour obtenir des meilleurs résultats.

Comme indiqué précédemment, certaines analyses ont malheureusement dues être raccourcies ou tronquées, conséquemment aux nombreux soucis de stabilté de la plateforme de travail. Un temps conséquent a en effet dû être sacrifié pour ajuster des scripts continuellement, du fait de problèmes multiples, successifs et indépendants de notre volonté. Avoir des résultats fiables demandait un minimum de volume pris en compte dans la modélisation, et nous étions donc tributaires du bon fonctionnement des outils. Dans le cas des bases volumineuses automobile, cela a posé problème.

4.3 Création de la variable d'interaction

4.3.1 Choix de l'arbre appliqué aux résidus lissés

Une fois l'algorithme de machine learning choisi, il faut déterminer la profondeur de l'arbre que nous souhaitons implémenter sur les prédictions de cet algorithme. Pour cela quelques contraintes sont à prendre en compte. Premièrement, nous devons chercher la profondeur telle que notre algorithme ne s'adapte pas

trop à sa base d'apprentissage (sur-apprentissage). Deuxièmement, nous devons déterminer au préalable une limite à la profondeur pour que celle-ci n'implique pas la création d'un trop grand nombre de modalités à implémenter lors de la mise en production. Pour déterminer avec certitude le nombre de variables dans chaque branche de nos arbres nous commençons par implémenter une fonction sur R à cette vocation. Celle-ci va permettre de tronquer nos arbres optimaux à des longueurs spécifiées en paramètre.

Pour la suite, nous nous inspirons de la méthode du bootstrap pour déterminer un nombre optimal de variables par branche. Pour cela, nous tirons de manière aléatoire 8 échantillons de notre base, ce nombre qui peut sembler faible pour du boostrap, a été choisi avec les contraintes de temps de calcul imposées par les outils utilisés. Chaque tirage se fait avec remise et permet d'obtenir une base de la même dimension que notre base d'origine.

Nous créons sur chacune de ces bases des arbres auxquels nous appliquons la fonction précédemment implémentée. Cela nous permet d'avoir, par base, des arbres de différentes profondeurs que nous pourrons comparer par la suite. Nous déterminons ensuite sur chaque arbre quelle est la profondeur qui minimise le RMSE. Puis nous comparons les différentes profondeurs considérées comme optimales. Cela nous permet dans une certaine mesure de conclure quant à la stabilité de notre choix sur la profondeur. En effet, si les résultats des différents arbres sont similaires ou proches, cela nous conforte dans notre choix. Si au contraire les profondeurs sont très éloignées les unes des autres, cela nous indique que nos arbres ne donnent pas des résultats stables malgré les algorithmes de machines leaning que nous avons appliqués à notre base au préalable. Les résultats ne sont donc pas stables et la méthode n'est pas réplicable à l'avenir. Cela peut poser problème que ce soit au niveau de l'interprétation des résultats obtenus, de l'explication et la justification de la méthode vis à vis des autorités de l'entreprise ou même lors de la mise en production de la méthode.

Puis, nous créons un arbre de cette profondeur déterminée comme étant optimale sur chaque échantillon bootstrap à notre disposition. Nous comparons à ce moment-là l'importance accordée à chaque variable par chaque arbre. Ainsi cela nous permettra de conclure quant à la stabilité de l'arbre issu des Machine learning. Nous pouvons également comparer les différents chemins et vérifier si les séparations s'effectuent avec les mêmes variables et avec des seuils similaires. En ce qui concerne l'importance des variables, nous les regardons dans leur globalité avec la moyenne et la variance de l'importance de chacune d'elles. La moyenne nous permet de déterminer si la variable est discriminante ou pas, et la variance nous permet de déterminer si l'importance est stable sur les arbres. Nous souhaitons que ces arbres soient stables. En effet cela permettrait de créer une variable robuste, de plus, nous effectuons du machines learning dans ce but-là.

Une fois que nous avons conclu quant à la stabilité des arbres ainsi qu'à leur profondeur optimale, nous pouvons en créer un final sur notre base complète. Nous nous basons à la fois sur les prédictions et sur la variable à expliquer directement. Cela nous permettra par la suite de conclure vis à vis de l'efficacité des machine learning. Nous nous retrouvons finalement avec deux arbres dont nous modifions les prédictions afin que celles-ci renvoient à un chemin que nous aurons référencé au préalable. Nous regardons encore une fois l'importance des variables ainsi que les différents chemins, puis nous établissons un parallèle entre l'arbre avec machine learning et l'arbre sans machine Learning. Cela nous permet d'avoir un premier aperçu de l'effet de nos machine learnings. Nous conservons également les importances des variables afin d'essayer

de croiser directement les variables étant ressorties via des croisements sur le logiciel de modélisation. La Figure 4.22 synthétise cette démarche avec $g(.)$ la fonction de lien des glm utilisés dans le Chapitre 3 lors de la révision tarifaire, les ϵ représentant les résidus, Y notre variable à expliquer (prime pure dans le cas du dommage tous accidents et fréquence dans le cas du bris de glace) et X les variables explicatives

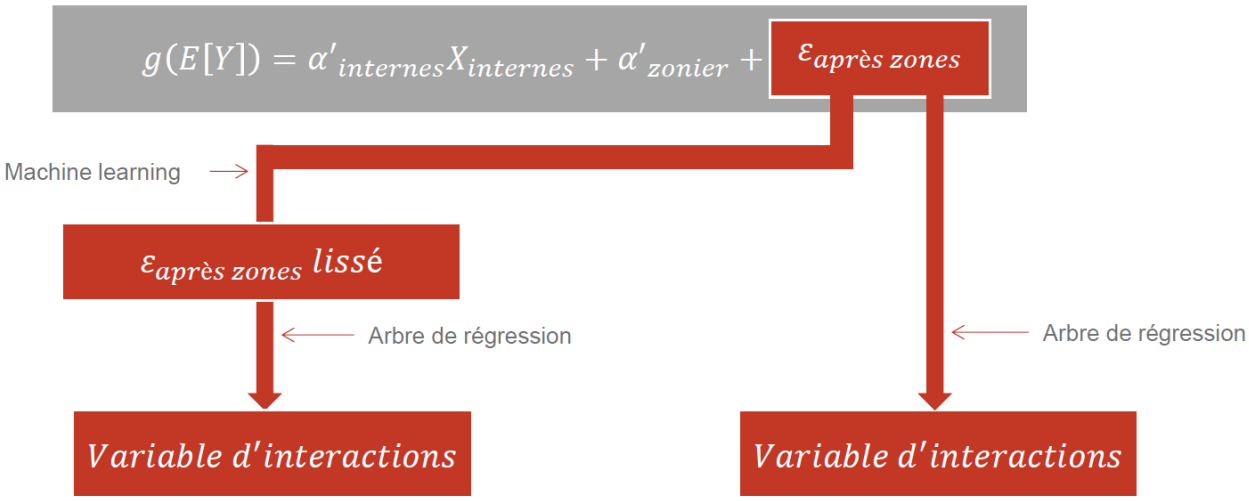

FIGURE 4.22 – Schématisation de la création des deux variables d'interactions

Finalement, une fois ces deux arbres créés, nous les appliquons sur notre base totale de modélisation afin de créer deux variables d'interactions. Les ajouter à notre base de modélisation nous permet ensuite de les tester dans nos modèles de risques. Nous étudions les variables une à une, sur chaque garantie sur chacun des modèles de risques (fréquence ou coûts moyens) pour conclure quant à leur pouvoir prédictif et segmentant. Pour cela nous cherchons en premier lieu si elles sont significatives, puis si elles améliorent les modèles par les indicateurs usuels tels que l'AIC ou la déviance. Enfin nous cherchons à déterminer le juste milieu entre complexité du modèle et apport de la variable sur sa prédictivité globale.

4.3.2 Garantie bris de glace

Création de la variable d'interaction

Pour l'application de cette méthode dans le cadre de la garantie bris de glace nous nous voyons obligés de réduire le nombre d'échantillons de notre base bootstrappée. En effet cela nous permet de réduire le temps de calcul ainsi que les ressources mémoires requises pour faire tourner notre script. Nous travaillons donc avec une base d'apprentissage représentant 50% de notre base totale et avec 3 échantillons bootstrappés (et non 8 comme prévu initialement). Cela permet de mettre en lumière une des préoccupations dont nous avons précédemment parlé : la méthode doit pouvoir être mise en production.
Nous testons donc des arbres avec des profondeurs variant entre 4 et 8. La profondeur retenue est 5. En appliquant cette profondeur à notre vecteur de base boostrappée nous obtenons les importances présentes sur la *Figure 4.23*.

Nous constatons que, contrairement aux résultats des algorithmes de machine learning de la section précédente, la variable la plus importante est le coefficient de réduction majoration suivi de l'âge d'acquisition du véhicule et l'ancienneté du contrat. Certaines variables ne sont jamais importantes. C'est le cas de la variable sur l'offre 8 000 km (R8000KM) ou la boite de vitesse (RBOITEVIT). De plus la variance de ces importances est assez faible dans le cas des variables les plus importantes. Nous pouvons donc supposer que les arbres sont assez stables et donnent des résultats similaires. Cependant notre variance est impactée par le nombre d'échantillon faible que nous avons du choisir et aurait apporté plus d'informations si nous avions pu réaliser un bootstrap avec un nombre d'échantillon bien plus élevé.

	moyenne	variance
Age_acquisition_vehicule	4.47615171	5.268100e-05
Age_conducteur	2.35272005	7.397912e-06
Age_obtention_permis	0.01328336	2.133361e-06
Age_vehicule	1.55121700	2.018837e-05
Anciennete_contrat	3.69100929	1.038509e-04
Anciennete_permis	2.01257089	2.246047e-06
BonusMalus	4.88738410	2.021197e-04
NbSinAnt_Resp	0.03299769	1.740801e-06
PCSP2	0.90602981	8.775890e-04
PFRACT	0.63409197	3.999132e-07
RCAROS	0.09262585	4.803898e-07
RFORMULE	0.35314113	9.314250e-02
RGARAGE	2.13482911	1.913275e+00
RLEASING	1.66756032	1.265110e+00
RNBMOIS_ASS	0.77595240	4.568951e-02
RUSAGE	1.09418450	1.487695e-01
Zone_maj_RC	0.46056659	5.439549e-01
RANNEE	0.01178774	4.168527e-04
RCLSRA	0.00000000	0.000000e+00
RGRSRA	0.00000000	0.000000e+00
RCARBUR	0.00000000	0.000000e+00
RBOITEVIT	0.00000000	0.000000e+00
R8000KM	0.00000000	0.000000e+00
NbSinAnt_NonResp	0.00000000	0.000000e+00
NbSinAnt_Gel	0.00000000	0.000000e+00
NbSinAnt_Bdg	0.00000000	0.000000e+00
Reseau	0.00000000	0.000000e+00

FIGURE 4.23 – Importance des variables sur des arbres de profondeur 5 pour le bris de glace

Nous passons à la construction sur notre base complète de notre variable d'interaction en attribuant un label à chaque feuille de l'arbre qui sert à la construire. Nous construisons cette variable, à la fois sur les prédictions du machine learning, à la fois sur nos résidus sans transformation. La construction de ces deux variables permettra de valider la démarche que nous avons eu à l'étape précédente. Nous pourrons à la fois comparer l'importance accordée aux variables explicatives ainsi que la performance des modèles avec l'ajout de l'une ou de l'autre. Ainsi en termes d'importance de variables nous avons les résultats présents dans la *Figure 4.24*.

Ainsi à première vue, les deux arbres semblent assez différents. Certaines variables n'ont pas la moindre importance pour l'arbre construit sur les prédictions, alors que toutes les variables de l'arbre construit sur les résidus ont une certaine importance. On constate également que l'ordre des importances diffère suivant l'arbre. C'est ainsi que la variable du zonier de la garantie responsabilité civile est la plus importante sur l'arbre des résidus non modélisés alors qu'elle est parmi les moins importantes de l'arbre des résidus modélisés par gradient boosting. Notre modélisation au préalable des résidus a donc un effet qui distingue les deux variables d'interactions construites sur des arbres de régression.

Avant d'intégrer ces deux variables à notre base de modélisation, nous visualisons une première fois la segmentation de notre base de modélisation par chacune des deux variables créées en ordonnant au préalable les modalités par la moyenne des résidus de chaque population. Nous regroupons si nécessaire, les modalités sous représentées, dans un but de modélisation plus juste. Conserver un groupe sous représenté, impliquerai une mauvaise estimation de l'effet de ce groupe en particulier par manque d'information et fausserai le coefficient attribué par le modèle linéaire généralisé final.

variable_ML	ImportanceML	variable_sansML	Importance_sansML
<fct>	<dbl>	<fct>	<dbl>
RANNEE	0.00000000	RNBMOIS_ASS	0.02987864
RGRSRA	0.00000000	RLEASING	0.04055810
RBOITEVIT	0.00000000	NbSinAnt_Gel	0.05006631
R8000KM	0.00000000	R8000KM	0.33203946
NbSinAnt_NonResp	0.00000000	PFRACT	0.40707953
NbSinAnt_Gel	0.00000000	RBOITEVIT	0.46536148
NbSinAnt_Bdg	0.00000000	NbSinAnt_Resp	0.46997563
Reseau	0.00000000	RCARBUR	0.54922586
Age_obtention_permis	0.03855673	Reseau	1.21718595
NbSinAnt_Resp	0.09982460	RFORMULE	1.25978721
Zone_maj_RC	0.36721819	NbSinAnt_NonResp	1.39805444
RCAROS	0.68882869	NbSinAnt_Bdg	1.45872046
RCLSRA	0.81906364	RUSAGE	1.82068998
RCARBUR	0.84271461	RGARAGE	3.87497640
RFORMULE	1.58505772	RCAROS	4.31742561
PFRACT	1.89105216	RANNEE	4.33365637
RNBMOIS_ASS	1.93588495	PCSP2	4.57155996
RLEASING	2.97450511	RGRSRA	5.58433421
PCSP2	3.74583726	Age_acquisition_vehicule	5.94671792
Age_vehicule	4.62118934	Age_obtention_permis	6.17774093
RGARAGE	6.76698679	BonusMalus	6.23150588
RUSAGE	7.90787254	Age_vehicule	7.14224713
Anciennete_permis	8.44654509	RCLSRA	7.95793791
Age_conducteur	10.44559500	Anciennete_contrat	8.09142854
Age_acquisition_vehicule	13.42222110	Age_conducteur	8.37979125
BonusMalus	16.59116058	Anciennete_permis	8.85988402
Anciennete_contrat	16.80988590	Zone_maj_RC	9.03217083

FIGURE 4.24 – Comparaison des importances des variables d'interaction avec et sans prédiction des résidus bris de glace

Nous obtenons les expositions par modalités suivantes de la *Figure 4.25* avec à droite les expositions de la variable construite sur les résidus modélisés et à gauche celle construite sur les résidus bruts. Nous pourrions encore regrouper les dernières catégories de la variables d'interaction construite sur les résidus modélisés, nous le ferons par la suite lors de la modélisation finale.

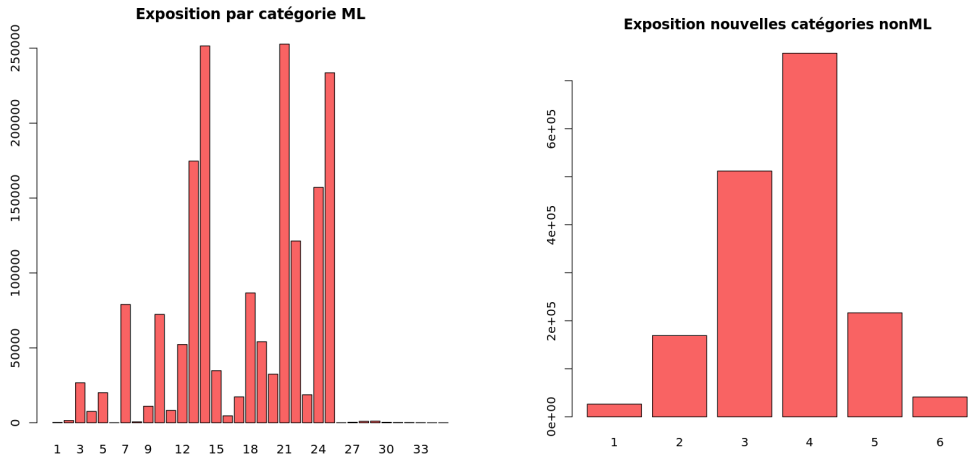

FIGURE 4.25 – Expositions des deux variables d'interaction des résidus bris de glace

Étude qualitative de la variable d'interactions

En s'intéressant au détail du contenu de notre variable d'interactions nous pouvons remarquer qu'elle permet de représenter certains phénomènes observés dans la pratique : en particulier, l'interaction entre le type de carrosserie (RCAROS) et l'âge du véhicule qui est significative pour nombre de garanties automobile, notamment la bris de glace. On peut observer ce phénomène dans le heatmap ci-dessous (Figure 4.26) représentant l'impact de l'interaction entre ces deux variables sur la fréquence bris de glace total. Cette représentation est issue d'un travail réalisé dans un autre cadre que celui du mémoire, et constitue une référence métier. Même si des différences qualitatives peuvent apparaître car le stade d'analyse n'est pas exactement le même, la logique de significativité de cette interaction est en tout cas vérifiée dans le chemin juste en dessous (Figure 4.27), extrait de notre arbre. Les branches observées lient bien le type de carrosserie et notamment les berlines avec un âge de véhicule faible.

Ce besoin d'interaction s'observe également sur l'univarié des résidus (Figure 4.28), qui montre des sur ou sous-estimations sur les segments décrits ci-dessus. Ainsi nous savons que cette interaction est bien absente des modèles de la garantie Bris de glace et que notre variable d'interaction peut nous permettre d'intégrer cet effet réel.

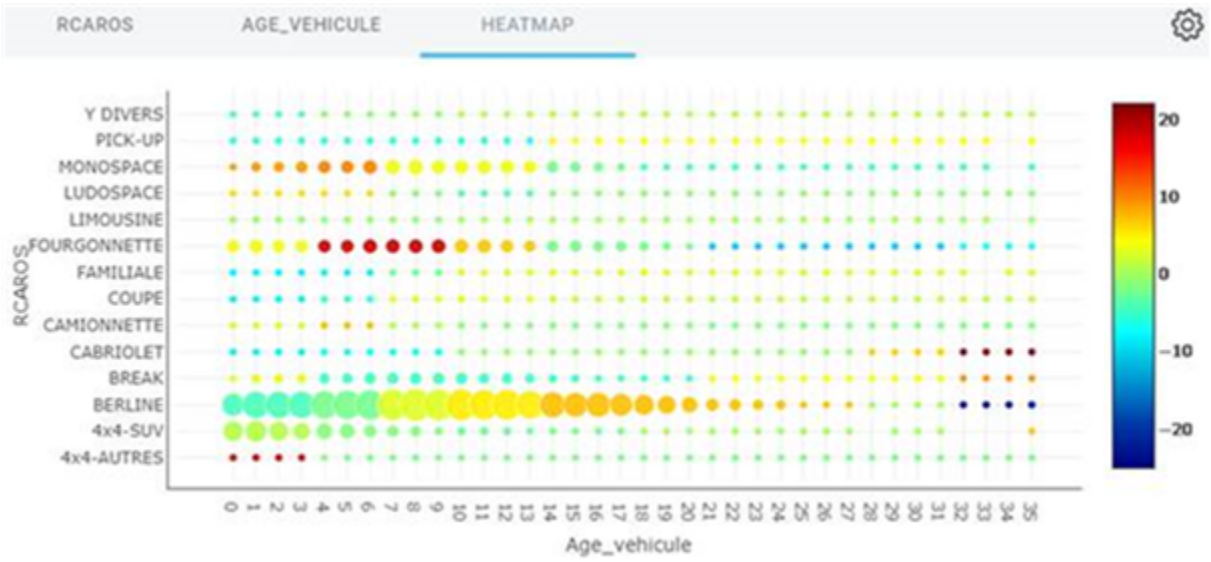

FIGURE 4.26 – Heatmap des interactions observées pour le BDG entre le type de carrosserie et l'âge du véhicule

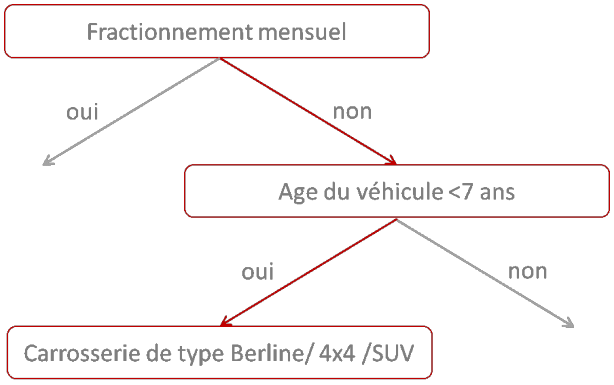

FIGURE 4.27 – Extrait de chemin de la variable d'interactions BDG

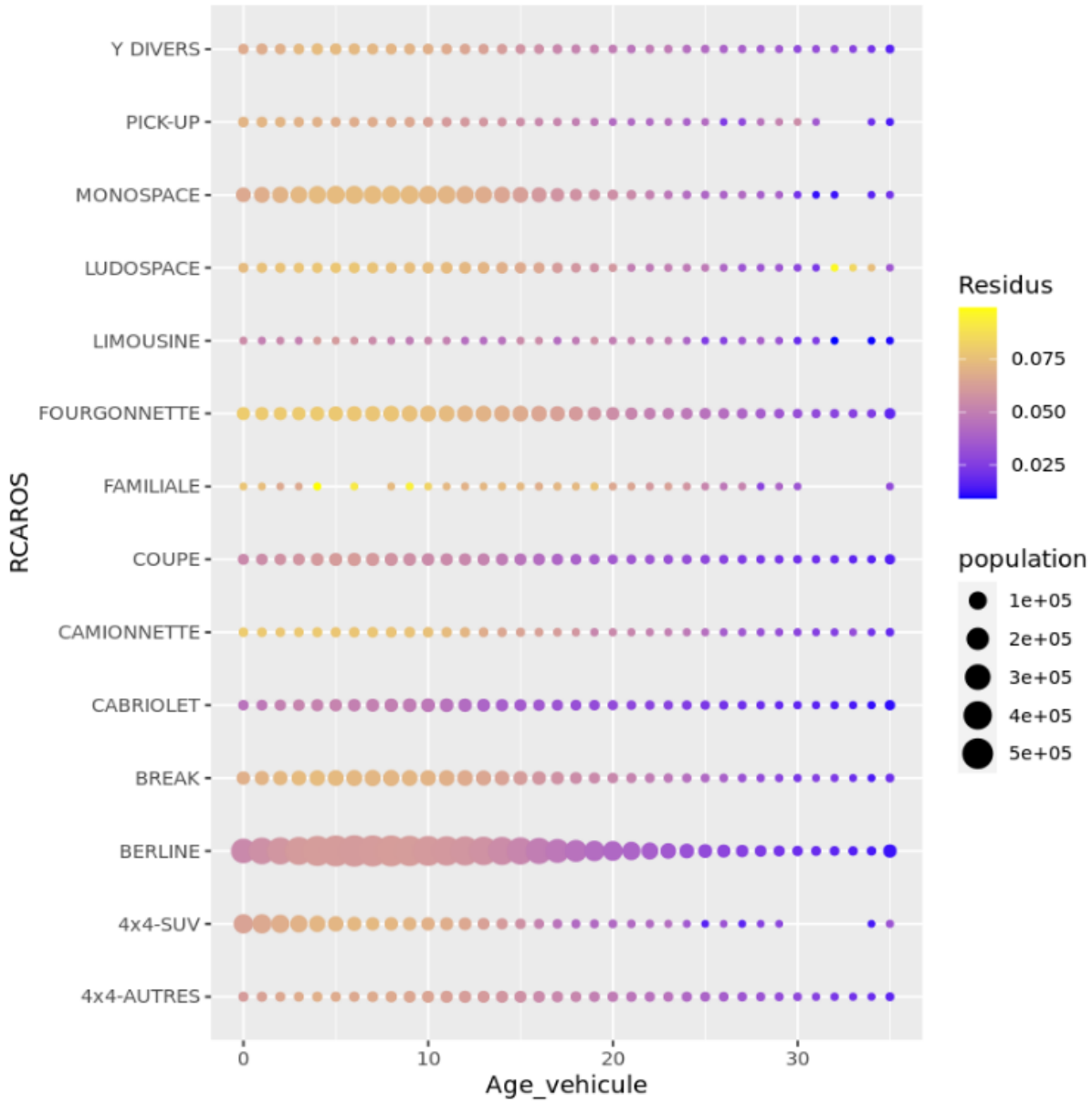

FIGURE 4.28 – Interaction entre le type de carrosserie et l'âge du véhicule sur les résidus BDG

Modélisation finale de la garantie

Nous avons construit deux variables d'interactions sur résidus de fréquence de nos précédentes modélisations. Nous allons maintenant en mesurer l'impact lorsque nous les ajoutons aux modèles et décider si nous les conservons ou pas. Une des particularités du bris de glace est que nous séparons sa modélisation en deux parties, d'un coté nous modélisons le bris de glace partiel et de l'autre le bris de glace total. Nous allons donc tester nos variables sur ces deux parties.

Que ce soit pour le bris de glace partiel ou total, aucune des variables d'interactions n'est ressortie comme étant significative dans les modèles de coûts moyens. Cela ne nous surprend pas car elles ont été construites sur les résidus de fréquence. Or les coûts moyens et la fréquence sont considérés comme indépendants dans le cadre de notre étude, c'est d'ailleurs ce qui nous permet de décomposer la prime pure par la fréquence et le coût moyen.

En ce qui concerne le modèle de fréquence du bris de glace partiel, les deux variables sont significatives, nous comparons dans le tableau suivants les variations de certains indicateurs de significativité des modèles avant et après ajout des variables :

	gain en AIC	gain en déviance	p-value de Wald
variable d'interaction des résidus modélisés	-16.92	-24.92	0%
variable d'interaction des résidus bruts	-242.16	-252.16	0%

Nous pouvons constater que les deux variables améliorent le modèle de base (diminution de la déviance et de l'AIC) et les deux variables sont significatives (p-value inférieures à 5%). Nous pouvons également remarquer que la variable construite sur les résidus bruts améliore grandement le modèle par rapport à la variable construite sur les résidus modélisés. Ce qui nous conduit à ne garder que celle-là.

Plus visuellement, les deux effets de ces variables dans le modèles peuvent être représentés comme dans la *Figure 4.29*. Nous pouvons y observer les lignes oranges qui représentent l'effet de variable avant regroupements et les vertes qui représentent l'effet des variables après un travail de celles-ci, l'histogramme en jaune représente l'exposition de chaque modalité (comme représenté précédemment sur la Figure 4.25).

Pour la variables issues des résidus modélisés (en haut), nous avons regroupé les modalités en 5 groupes pour gommer les bruits et simplifier le modèle en ne conservant pas trop de modalités. Pour la variable issue des résidus bruts (en bas), nous la conservons telle quelle. Dans les deux cas nous remarquons une tendance décroissante avec les modalités ce qui est cohérent avec la construction des variables. Les modalités sont également significativement différentes les unes des autres avec des amplitudes de 25% pour la variable issue des résidus modélisés et 70% pour l'autre.

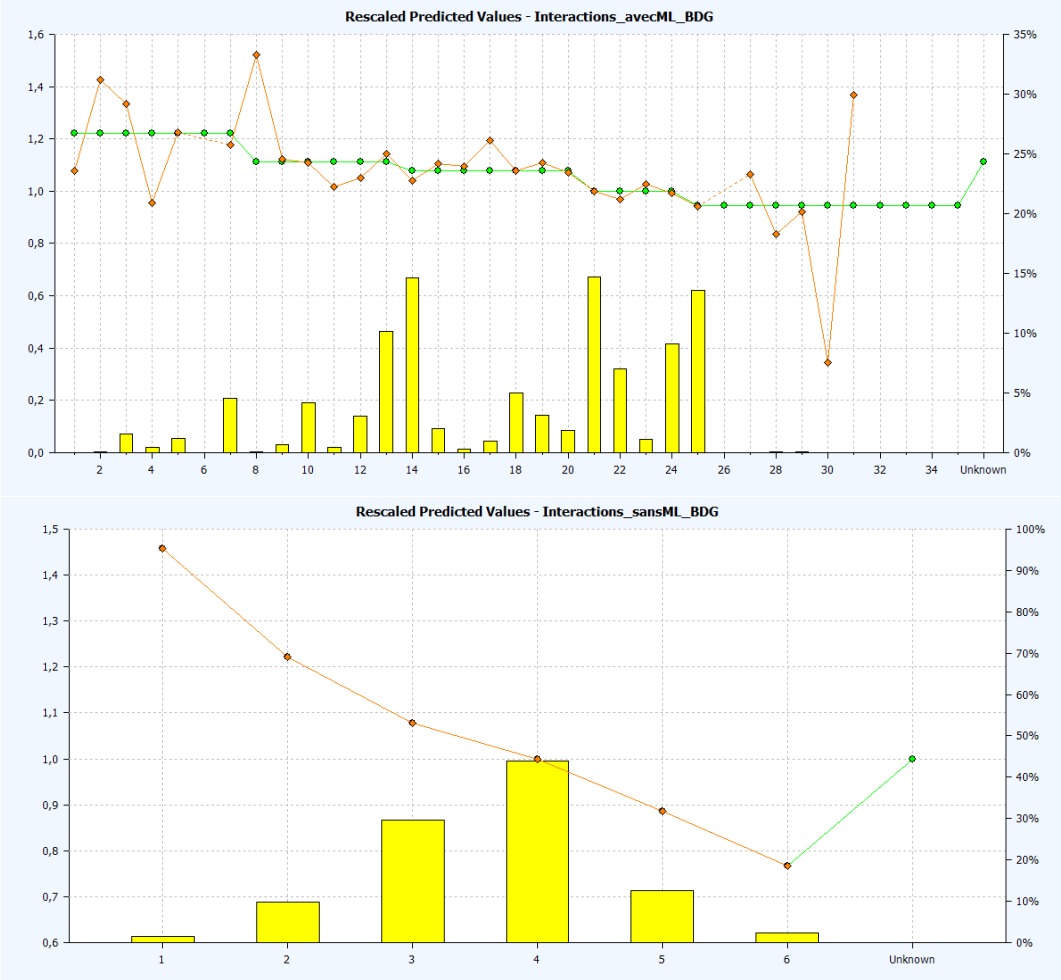

FIGURE 4.29 – effet des variables d'interactions dans le modèle de fréquence du bris de glace partiel

Pour conclure sur le modèle de fréquence du bris de glace partiel, bien que les deux variables soient significatives et discriminantes, nous choisissons de conserver celle issue des résidus bruts plutôt que celle issue des résidus modélisés car elle apporte des diminutions de l'AIC et de la déviance bien supérieures.

En ce qui concerne la modélisation de la fréquence du bris de glace total, nous obtenons le tableau comparatif suivant :

	gain en AIC	gain en déviance	p-value de Wald
variable d'interaction des résidus modélisés	-168.47	-176.47	0%
variable d'interaction des résidus bruts	-930.69	-920.69	0%

Comme lors de la modélisation du bris de glace partiel, les deux variables améliorent l'AIC et la déviance du modèle, les deux sont significatives avec des p-values inférieures à 5%. Et nous observons également que la variable construite sur les résidus bruts apporte des améliorations du modèle plus conséquentes. Nous représentons dans la *Figure 4.30*.

FIGURE 4.30 – effet des variables d'interactions dans le modèle de fréquence du bris de glace total

Nous avons, là aussi fait le choix de ne retravailler que la variable issue des résidus modélisés, nous obtenons au final 5 modalités. Nous retrouvons également une tendance décroissante avec les modalités, cohérente avec la construction des variables et des amplitudes de 50% dans le cas de la variable issue des résidus modélisés et de 80% dans le cas de la variable issue des résidus bruts. Au final nous pouvons en tirer des conclusions similaires. Bien que les deux variables améliorent la prédictivité du modèle, qu'elles soient discriminantes et significatives, l'apport de la variable issue des résidus bruts et supérieur à la variables issue des résidus de machine learning. Nous conservons pour la modélisation de la fréquence du bris de glace total, tout comme lors de la modélisation de la fréquence du bris de glace partiel, la variable issue des résidus bruts.

Nous étudions finalement les effets de ces variables que nous avons ajouté sur une base de validation. Nous obtenons les résultats des *Figures 4.31* et *4.32*. Nous pouvons y observer que les effets sur notre base de modélisation (courbe verte) peuvent se généraliser à notre base de validation car nous observons des tendances similaires (courbe orange). Ainsi bien que nous ayons construit cette variable sur des résidus non re-modélisés par un algorithme de machine learning, nous obtenons une stabilité dans l'impact de l'ajout de cette variable dans nos modèles linéaires généralisés. Cette dernière analyse nous permet de valider nos choix de modélisations quant à l'ajout de cette variable.

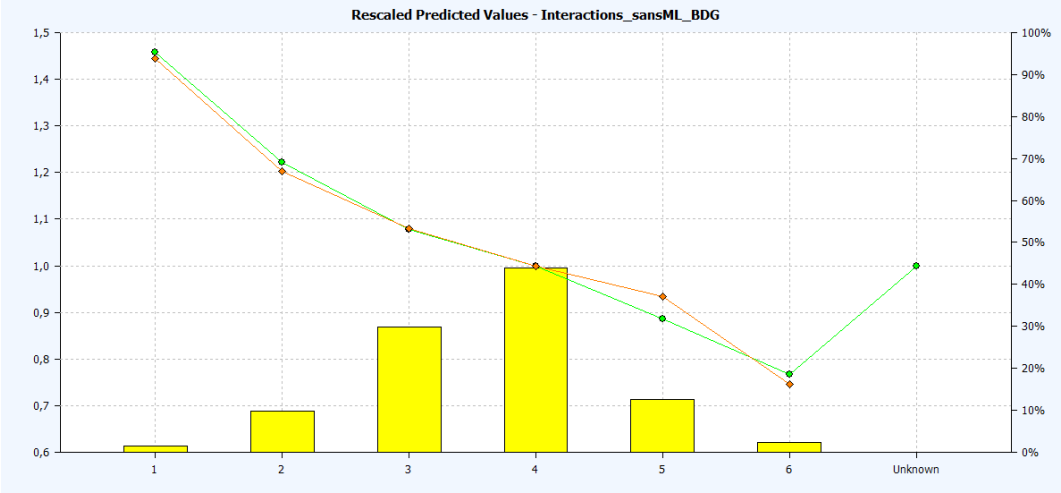

FIGURE 4.31 – Validation de l'usage de la variable d'interaction issue des résidus bruts sur la fréquence BDG partiel

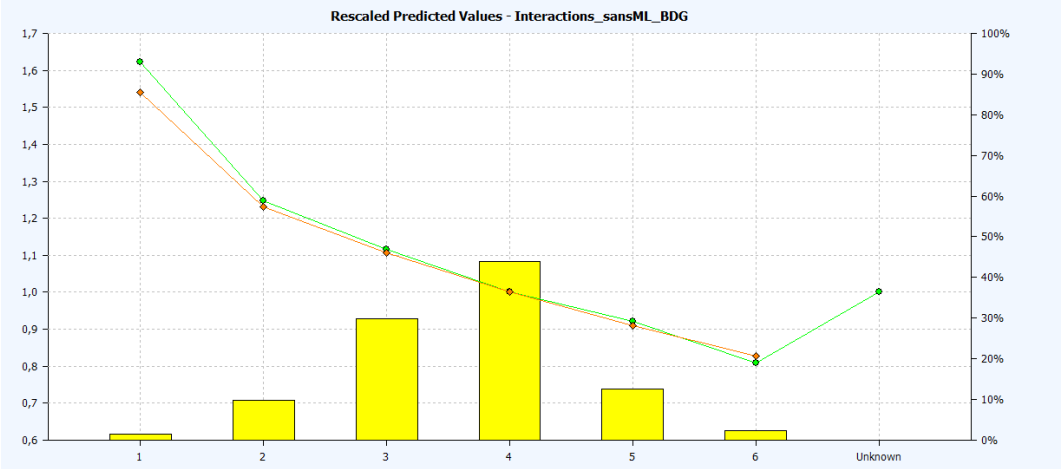

FIGURE 4.32 – Validation de l'usage de la variable d'interaction issue des résidus bruts sur la fréquence BDG total

Pour conclure sur la garantie bris de glace, nous avons considéré pour sa modélisation, deux variables construites à partir d'arbres. Le premier arbre a été construit sur la modélisation des résidus de fréquence par un algorithme de gradient boosting, le second sur les résidus sans retraitements. Alors que l'ajout d'aucune de ces variables n'a été concluant sur la modélisation du coût, les deux variables ont un effet bénéfique sur les modélisations des fréquences des sinistres bris de glace partiels et totaux. Puisque le pouvoir prédictif des modèles comprenant la variable d'interaction issue des résidus bruts est supérieur à celui comprenant la variable issue des résidus modélisés, nous choisissons de le conserver.

Ces résultats peuvent être amenés à changer d'une garantie à une autre. En effet le comportement et la population assurée sont différents d'une garantie à une autre ce qui modifie le risque et par extension sa modélisation. Nous nous intéresserons donc par la suite à l'impact de cette méthode sur la garantie dommages tous accidents, pour laquelle nous travaillons avec les résidus de prime pure plutôt que les résidus de fréquence.

4.3.3 Garantie dommages tous accidents

Création de la variable d'interaction

Pour l'application de cette même démarche sur la garantie dommages tous accidents, nous travaillons avec 3 échantillons bootstrap. Nous testons des arbres de profondeur comprise entre 4 et 8. Nous retenons une profondeur de 6. En la testant sur notre base bootstrappée, nous obtenons les importances de la *Figure 4.33*.

	moyenne	variance
Age_acquisition_vehicule	0.5027471	4.349640e-03
Age_conducteur	1.8407126	8.158419e-03
Age_obtention_permis	1.3345715	4.847901e-01
Age_vehicule	0.1871050	1.244762e-03
Anciennete_contrat	17.0145662	1.607794e-01
Anciennete_permis	4.3394145	1.321391e-02
BonusMalus	9.3418764	2.590488e-02
NbSinAnt_Bdg	10.3554468	2.667719e-01
NbSinAnt_Gel	11.5979374	2.146347e-01
NbSinAnt_NonResp	13.9498101	8.825345e-01
NbSinAnt_Resp	20.4210312	2.573888e-01
PCSP2	28.5776266	7.870961e-01
PFRACT	0.3283394	3.586978e-03
RANNEE	14.2203528	3.113968e-01
RBOITEVIT	0.5078517	6.347431e-07
RCARBUR	21.4724356	1.272168e+00
RCAROS	27.4281960	7.758342e-01
RCLSRA	12.1820521	2.390998e-02
Reseau	22.6488886	4.496234e-01
RGARAGE	5.9246341	3.377164e-01
RGRSRA	37.7040469	5.773024e-01
RLEASING	7.4286862	1.616138e+02
RUSAGE	15.8020730	1.875500e+02
RNBMOIS_ASS	0.0000000	0.000000e+00
RFORMULE	0.0000000	0.000000e+00
R8000KM	0.0000000	0.000000e+00
Zone_maj_RC	0.0000000	0.000000e+00

FIGURE 4.33 – Comparaison des importances des variables sur les échantillons bootstrap

Nous pouvons remarquer que les variables ayant la plus grand importance sont le groupe SRA (RCLSRRA), la carrosserie (RCAROS) et le réseau de distribution (Reseau). Ces trois variables n'étaient pas ressorties comme telles dans les machines learning précédents. Nous remarquons également que certaines variables n'entrent dans aucun arbres issu d'un échantillon bootstrap. C'est le cas pour le zonier de responsabilité civile (Zone_maj_RC), l'offres 8000km (R8000) ou la formule par exemple(RFORMULE). En s'intéressant à la variance de cette importance par modalité, le choix d'un bootstrap sur 3 échantillons la rend moins fiable. Cependant on peut remarquer la faiblesse de celle-ci sur les variables les plus importantes. On peut donc supposer que les arbres sont assez stables.

En construisant sur la totalité de notre base ces arbres nous permettant de créer notre variable d'interaction à la fois sur les résidus initiaux et les résidus lissés par machine learning, nous pouvons comparer une première fois l'apport de la partie 4.1.2. Les deux arbres ainsi construits sont à première vue bien différents. On remarque que l'arbre issu des résidus prédits par machine learning est bien plus segmentant car il crée plus de modalités pour notre variable d'interaction que l'arbre issue des résidus sans traitement.

Ci-dessous une représentation de l'exposition de notre garantie par modalité de la variable d'interaction :

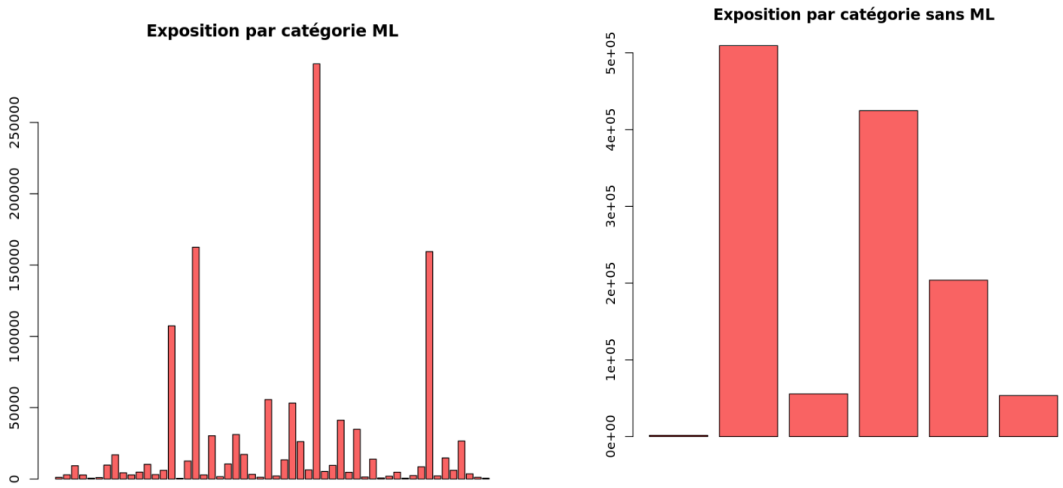

FIGURE 4.34 – Exposition des deux variables d'interaction des résidus DOM

Une fois ces deux variables construites et ajoutées à notre base de modélisation, nous pouvons mesurer leur apport dans la modélisation de la fréquence et du coût moyen des sinistres.

Étude qualitative de la variable d'interactions

Une fois ces variables construites nous cherchons à déterminer si elles permettent de représenter des effets que nous pouvons observer dans notre portefeuille. Nous notons que le croisement entre le type de carrosserie (RCAROS) et l'âge du véhicule ne se retrouve pas dans l'arbre. Une interaction de moindre importance peut être exhibée : type de carrosserie et boîte de vitesse. Son effet métier s'observe avec le heatmap de la Figure 4.35.

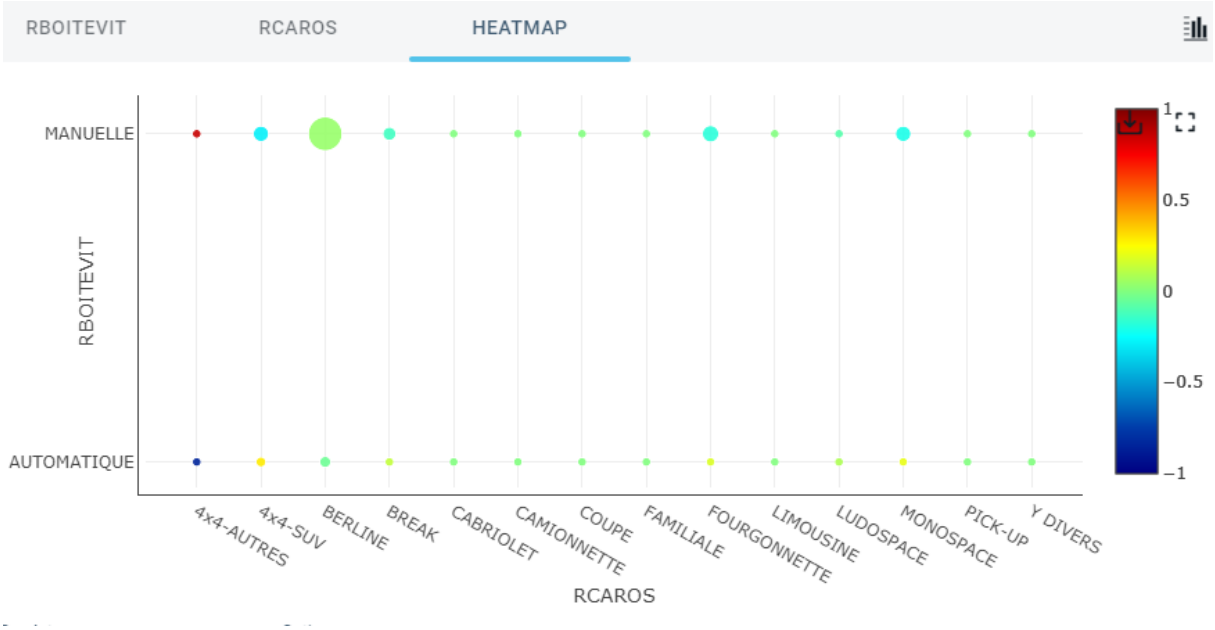

FIGURE 4.35 – Heatmap des interactions observées pour le DOM entre le type de carrosserie et la boite de vitesse

Dans la Figure 4.36 nous pouvons extraire ce croisement dans une des branches.Cet effet est observable également sur nos résidus re-modélisés par machine learning (figure 4.37 et 4.38) ainsi que nos résidus non re-modélisés. L'arbre a capté une information cohérente, mais moins significative que dans le cas du BDG, les amplitudes (différences de couleurs) de résidus étant bien moins marquées.

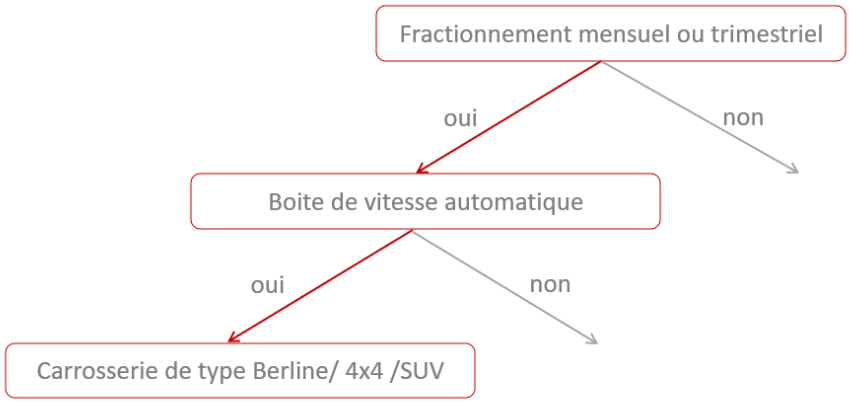

FIGURE 4.36 – Extrait de chemin de la variable d'interactions DOM

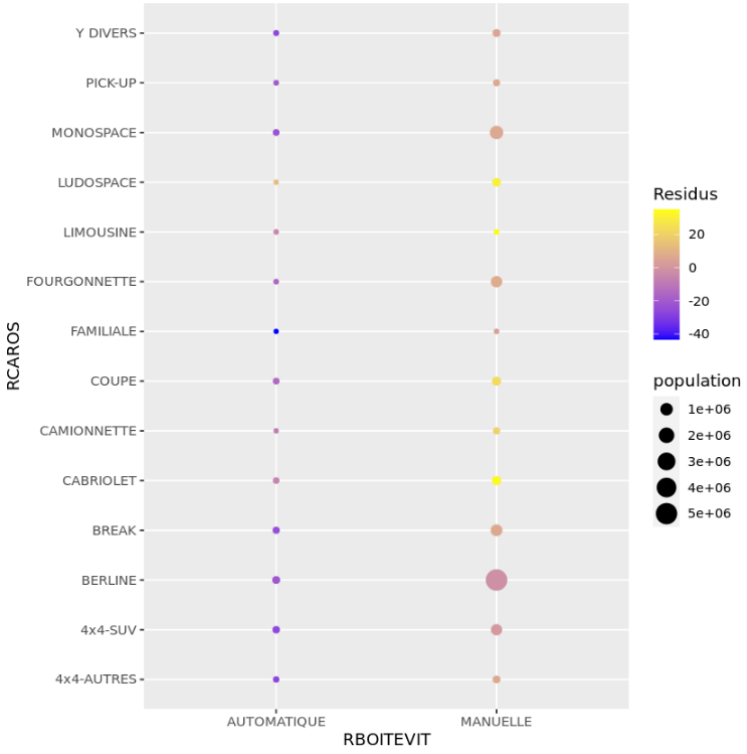

FIGURE 4.37 – Interactions entre le type de carrosserie et la boite de vitesse sur les résidus remodélisés

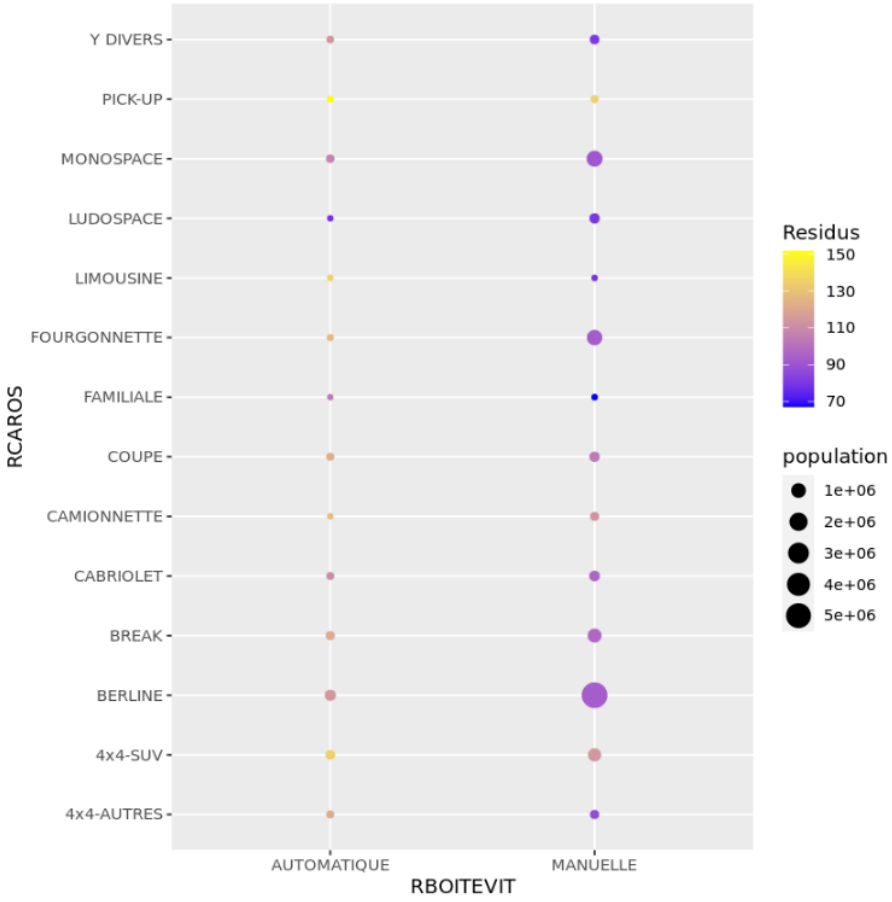

FIGURE 4.38 – Interactions entre le type de carrosserie et la boite de vitesse sur les les résidus non remodélisés

Modélisation finale de la garantie

La variable construite sur les résidus de prime pure sans traitement, n'est significative sur aucun des deux modèles. Aucune tendance ne se dégage de la variable, bien que nous ayons ordonnés ses modalités de manière à avoir une prédiction décroissante avec les modalités. Par exemple, pour le modèle de fréquence nous obtenons la *Figure 4.39*

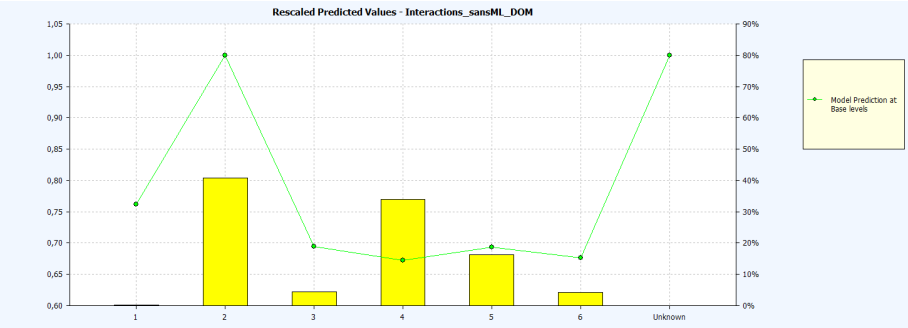

FIGURE 4.39 – Effet de la variable d'interaction construite sur un arbre seul DOM

Cette variable dégrade l'AIC de 2 et améliore la déviance de 5. La variable n'est pas significative et ne passe pas le test du $\chi 2$ avec une p-value de 0,2. Tout cela nous conduit à ne pas ajouter cette variable à notre modèle de fréquence, car elle détériorerait nos prédictions. Nous obtenons des résultats similaires sur le modèle des coûts moyens. La construction d'une variable d'interaction sans retraitement des résidus n'est donc pas concluante sur la garantie dommages tous accidents.

DOM : résidus bruts	gain en AIC	gain en déviance	p-value de Wald
Modèle de fréquence	+2.4	-5.6	23.1%
Modèle de coûts	-3.24	-3.48	2.8%

En ce qui concerne la variable construite sur les résultats de l'application des machine learning sur les résidus de prime pure, nous choisissons de la conserver sur le modèle de coûts moyens avec des regroupements. Elle est significative dans le modèle et on observe une tendance décroissante avec les modalités. La variable ainsi regroupée améliore légèrement la déviance ainsi que l'AIC du modèle. Elle passe également le test du $\chi 2$ (p-value inférieure à 5%) ce qui nous indique que cette variable est significative.

Comme l'illustre la *Figure 4.40* nous avons choisi de créer trois groupes afin d'avoir un effectif suffisant dans chaque classe pour avoir des modalités significatives, et éviter de modéliser du brut. Au final, la variable est significative et comporte des modalités significativement différentes les unes des autres sans trop complexifier le modèle initial. En effet un des dangers de la modélisation est d'intégrer trop de modalités différentes ce qui induit un risque de sur-apprentissage. Cette variable capte une tendance décroissante avec une amplitude de 6%. Il est donc nécessaire de la conserver dans notre modèle de coûts moyens.

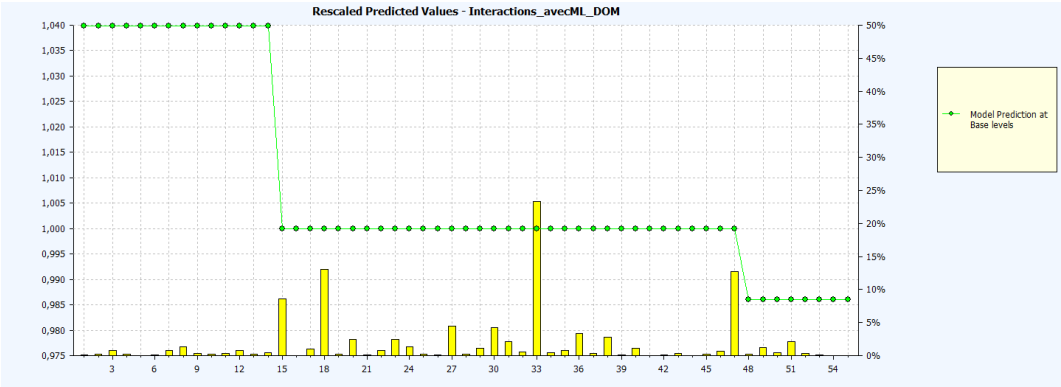

FIGURE 4.40 – Effet de la variable d'interaction construite après machine learning DOM

Pour le modèle de fréquence sa significativité est discutable. Nous créons également trois groupes, représentant une tendance décroissante ce qui est cohérent avec l'ordre octroyé aux modalités de notre nouvelle variable. La décroissance à la fois sur la fréquence et les coûts moyens implique la décroissance sur la modélisation prime pure qui est la multiplication de la fréquence par le coût moyen. Ces résultats sont cohérents avec la construction de notre variable.

Cependant la significativité sur le modèle de fréquence est discutable, le chi2 a une valeur de 14,8% ce qui est supérieur à 5% et donc réfute l'hypothèse selon laquelle notre variable est significative dans notre modèle. De plus les modalités ne sont pas significativement différentes les unes des autres. La variable améliore la déviance du modèle mais détériore légèrement son AIC. Nous choisissons donc ne pas la conserver dans le modèle de fréquence.

DOM : résidus modélisés	gain en AIC	gain en déviance	p-value de Wald
Modèle de fréquence	+0.17	-3.82	14.8%
Modèle de coûts	-8.40	-8.21	0.3%

Nous étudions finalement les effets de cette variable sur une base de validation. Nous obtenons les résultats de la *Figure 4.41* . Nous pouvons y observer que les effets sur notre base de modélisation (courbe verte) se généralisent difficilement à notre base de validation. Nous notons des tendances inversées entre la première et la deuxième modalité ainsi qu'entre les deux dernières. Cela peut être du à un manque d'exposition sur ces modalités.

FIGURE 4.41 – Validation de l'usage de la variable d'interaction issue des résidus modélisés sur les coûts DOM

Pour conclure sur cette garantie, nous avons construit deux variables pour capter les interactions entre celles déjà présentes dans le modèle. Pour cela nous avons choisi de travailler sur nos résidus de prime pure, comme nous l'avons fait pour la construction du zonier de la garantie DOM, dans un but de cohérence des méthodes. Nous avons construit une première variable sur les résidus de prime pure sans traitement préalable et la seconde sur les résidus auxquels nous avons appliqué une régression elasticnet. Nous avons pu observer en premier lieu que construire une variable sur nos résidus retraités, nous permettait de segmenter bien plus facilement notre population. Ensuite, la première variable ne s'est montrée significative sur aucun des modèles de fréquence ou de coûts moyens, au contraire de la seconde variable construite qui s'est avérée significative sur le modèle de coûts moyens.

Finalement, en conclusion de cette étape de modélisation, nous obtenons les tableaux récapitulatifs des modifications des modèles linéaires généralisés suivants :

BDG	Fréquence TOT	Coûts TOT	Fréquence PAR	Coûts PAR
variable issue des résidus modélisés	non	non	non	non
variable issue des résidus bruts	oui	non	oui	non

Avec les notations :
— PAR = bris de glace partiel
— TOT= bris de galce total

Pour la garantie bris de glace, avant l'ajout de la variable d'interaction nous avions un RMSE sur le modèle de fréquence de 1,55325. Après cette étude nous avons les RMSE sur le nouveau modèle de fréquence de 1,55319, qui est inférieur au RMSE initial. Notre but initial était de réduire les erreurs de prédictions, au vu de ces résultats, nous pouvons valider partiellement la démarche sur la garantie bris de glace.

DOM	Fréquence	Coûts
variable issue des résidus modélisés	non	oui
variable issue des résidus bruts	non	non

Pour la garantie dommages tous accidents, avant l'ajout de la variable d'interaction nous avions un RMSE sur la prime pure de 5899,00. Après cette étude nous avons le RMSE sur la prime pure de 5898,9944, qui est inférieur au RMSE initial bien que très proche. Cet ajout de variable d'interaction améliore donc nos modèles.

Conclusion

Dans l'optique d'améliorer les modèles de prime pure automobile existants, nous les avons mis à jour en leur ajoutant provisoirement des informations externes, à terme remplacées par un zonier résumant l'information géographique. Le coeur du mémoire a consisté à intégrer une approche machine learning dans la construction des tarifs, afin de diminuer les résidus des modèles sous-jacents. Pour conserver une interprétabilité des résultats, ces machine learning ont été utilisés pour créer une variable synthétisant les interactions entre variables internes et géographiques. Cette variable sera à ajouter aux modèles de risques pour être valorisée au moyen des modèles linéaires.

La construction de cette variable d'interaction est passé par le lissage des résidus des modèles existants, en sélectionnant le machine learning le plus prédictif. Sur la garantie bris de glace, c'est un alogorithme de gradient boosting qui présente les meilleurs résultats et en ce qui concerne la garantie dommages tous accidents nous retenons une régression elasticnet. Cependant les limites des ressources informatiques ne nous ont pas permis de paramétrer complètement les modèles. Les résultats obtenus sont donc tributaires de ces limites. Ils sont donc discutables et le choix de l'algorithme de stabilisation peut être remis en question. Les modalités d'interaction, qui définissent la variable recherchée, sont finalement constituées des branches d'arbres de profondeur cinq, pour le bris de glace, et six, pour la garantie dommages tous accidents, appliqués sur le meilleur machine learning précédent. Cette optimalité de profondeur a été obtenue par bootstrap de la base, pour arriver à une stabilité des interactions sur les données futures.

Finalement, les modèles de fréquences et coûts moyens par garantie ont été complétés par cette variable d'interaction, afin de voir sa significativité. Une version alternative de variable d'interaction a également été testée, en appliquant directement un arbre aux résidus, sans lissage machine learning préalable. Dans les modèles bris de glace, seule la variable d'interaction alternative a été significative et donc retenue, tandis que sur la garantie dommages tous accidents, la variable d'interaction lissée par elasticnet a été retenue sur le modèle de coût moyen uniquement et les résultats ne se sont pas généralisés à notre base de validation. Les ressources informatiques limitées et la contrainte de mise en production ne nous ont permis ni d'exploiter au maximum le paramétrage des algorithmes de machine learning, ni d'appliquer entièrement la méthode sur d'autres garanties comme la responsabilité civile ou le vol.

La démarche donne des résultats peu probants sur le bris de glace et le dommage tous accidents, où la variable d'interaction n'intervient que partiellement et n'améliore les résidus que dans une très faible mesure.

Bibliographie

[1] Data analytics post. https ://dataanalyticspost.com.

[2] Ffa. https ://www.ffa-assurance.fr.

[3] Insee. https ://statistiques-locales.insee.fr.

[4] ActuIA. L'interprétabilité de l'ia – le nouveau défi des data scientists. https ://www.actuia.com/contribution/jean-cupe/linterpretabilite-de-lia-le-nouveau-defi-des-data-scientists/.

[5] Arthur Charpentier. *Actuariat IARD - ACT2040 Partie 4 - modèles linéaires généralisés*, 2013.

[6] Fédération Française de l'Assurance. Le marché de l'assurance automobile des particuliers en 2018. juin 2019.

[7] Jean-Michel Genuer, Robin Poggi. Arbres cart et forêts aléatoires, importance et sélection de variables. *ffhal-01387654v2*, 2017.

[8] Antoine Guillot. Apprentissage statistique en tarification non-vie : quel avantage opérationnel ?, 2015.

[9] Ron Kohavi. A study of cross-validation and bootstrap for accuracy estimation and model selection. 1995.

[10] Kasséa Kévin Axel Kouo. Tarification automobile : Glm vs rÉseaux de neurones.

[11] L.Breiman. Random forests, 2001.

[12] Matthew Liaw, Andy Wiener. Classification and regression by randomforest. déc 2002.

[13] Louis Maisonnave. Détermination de la rentabilité des agents auto et mrh : modélisation et interprétation via des méthodes d'apprentissage automatique, 2019.

[14] Eric Samuel Nana Njoya. Prédiction des comportements de rachat en épargne individuelle : une approche machine learning, 2016.

[15] Wedderburn Nelder. Generalized linear models. *Royal Statistical Society*, 1972.

[16] Samia Nouar. Méthodes de provisionnement ligne à ligne en assurance non-vie.

[17] Ricco Rakotomalala. *Régression Régularisée*.

[18] Robert Tibshirani. Regression shrinkage and selection via the lasso. 1995.

[19] Trevor Zou, Hui Hastie. Regularization and variable selection via the elastic net. *Royal Statistical Society*, 2005.

Annexes

Table des Annexes

1 Préparation de la donnée **2**

1.1 Formation de tranches SAS 2

1.2 Transformation des variables catégorielles en variables de type "Dummy" sur R 3

1.3 Distribution des résidus des garanties vol et de responsabilité civile matérielle 4

2 Quelques algorithmes de machine learning avec le package R "sparklyr" **5**

2.1 Un exemple de tuning de régression elasticnet avec "sparklyr" 6

2.2 Un exemple de tuning d'arbre CART avec "sparklyr" 7

2.3 Un exemple de tuning de forêt aléatoire avec "sparklyr" 8

2.4 Un exemple de tuning de gradient boosting avec "sparklyr" 9

2.5 Récupération de l'importance des variables et visualisation 10

Chapitre 1

Préparation de la donnée

1.1 Formation de tranches SAS

Un exemple de retranscription de format SAS pour construire des tranches sur une variable quantitative, avant la modélisation.

$Value\ ftTxMortaliteInfantile$

$$
\begin{gathered}
0 - 2.3 = 2.2 \\
2.3 - 2.5 = 2.4 \\
2.5 - 2.7 = 2.6 \\
2.7 - 2.9 = 2.8 \\
2.9 - 3.1 = 3 \\
3.1 - 3.3 = 3.2 \\
3.3 - 3.5 = 3.4 \\
3.5 - 3.7 = 3.6 \\
3.7 - 3.9 = 3.8 \\
3.9 - 4.1 = 4 \\
4.1 - 999999 = 4.2 \\
other = 999999 \\
;
\end{gathered}
$$

1.2 Transformation des variables catégorielles en variables de type "Dummy" sur R

```
#Conversion des variables chr en dummies
a<-Sys.time()

train_sc_bis<-train_sc %>% select(label=var_expliquee,var_explicative,dummies)
head(train_sc_bis)

(dummies2<-as.character(lapply(dummies,function(x) paste(x,'2',sep=''))))


(b<-Sys.time()-a)
```

1.3 Distribution des résidus des garanties vol et de responsabilité civile matérielle

Nous représentons ici des résidus centrés-réduits.

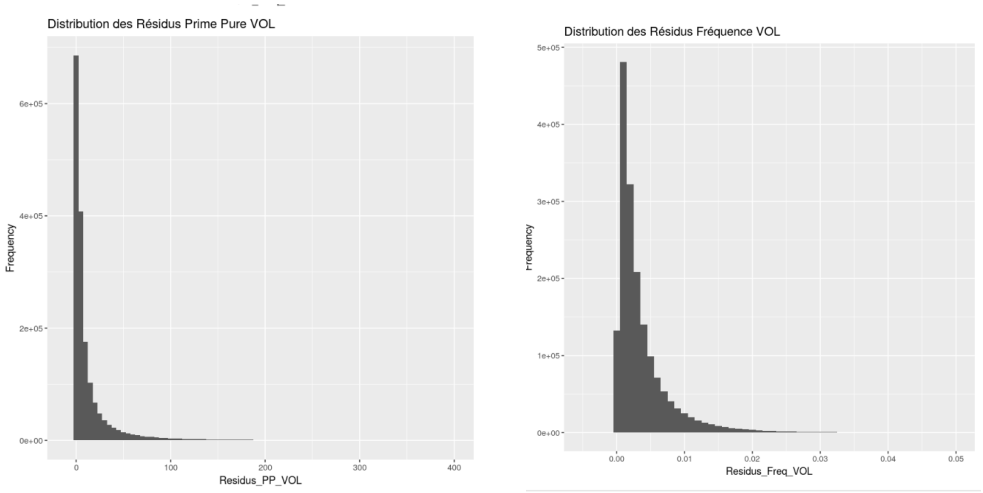

FIGURE 1.1 – A droite les résidus de fréquence VOl, à gauche, résidus de prime pure VOL

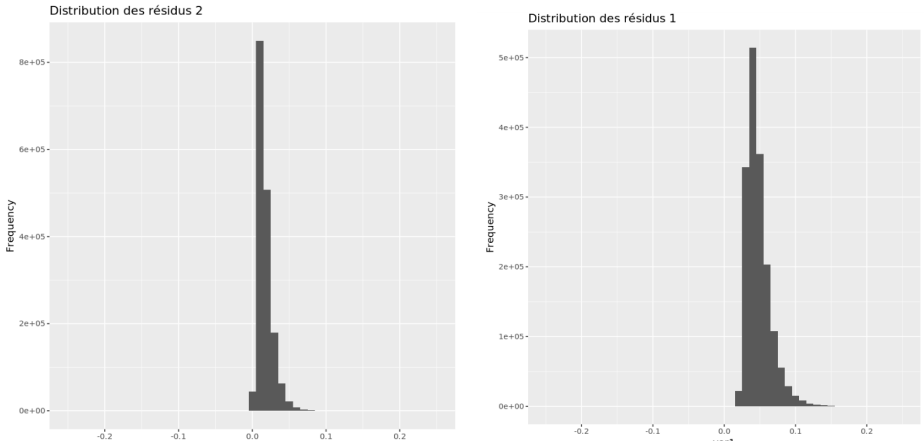

FIGURE 1.2 – A droite les résidus de fréquence , à gauche résidus de fréquence de la garantie responsabilité civile matérielle

Chapitre 2

Quelques algorithmes de machine learning avec le package R "sparklyr"

2.1 Un exemple de tuning de régression elasticnet avec "sparklyr"

```
a<-Sys.time()

pipeline_REG <- ml_pipeline(sc)
   for (i in 1:length(dummies)){
            pipeline_REG<-pipeline_REG %>%ft_string_indexer(input_col = dummies[i], output_col = paste(dummies[i],'1',sep=''))%>%
                                ft_one_hot_encoder(paste(dummies[i],'1',sep=''),paste(dummies[i],'2',sep=''))
               }
pipeline_REG<-pipeline_REG %>%
   ft_vector_assembler(c(var_explicative,dummies2),
                           "features_unscaled") %>%
   ft_standard_scaler(input_col = "features_unscaled", output_col = "features",
                      with_mean = TRUE,with_std = TRUE) %>%

   ml_linear_regression(fit_intercept=TRUE)


# Specify hyperparameter grid
grid <- list(
  linear_regression = list(
      elastic_net_param=c(0,1,0.5,0.25,0.75),
      reg_param=c(0,1,0.5,0.25,0.75)

  )
)

# Create the cross validator object
cv_REG <- ml_cross_validator(
  sc, estimator = pipeline_REG, estimator_param_maps = grid,
  evaluator = ml_regression_evaluator(sc),
  num_folds = 5,
  parallelism = 20
)


# Train the models
cv_model_REG <- ml_fit(cv_REG, train_sc_bis)

# Print the metrics
ml_validation_metrics(cv_model_REG)

(b<-Sys.time()-a)
```

2.2 Un exemple de tuning d'arbre CART avec "sparklyr"

```
a<-Sys.time()

pipeline_TREE<-ml_pipeline(sc)

for(i in 1:length(dummies)){
            pipeline_TREE<-pipeline_TREE %>%ft_string_indexer(input_col = dummies[i], output_col = paste(dummies[i],'1',sep=''))%>%
                           ft_one_hot_encoder(paste(dummies[i],'1',sep=''),paste(dummies[i],'2',sep=''))
               }
pipeline_TREE<-pipeline_TREE %>%
   ft_vector_assembler(c(var_explicative,dummies2),
                       "features_unscaled") %>%
   ft_standard_scaler(input_col = "features_unscaled", output_col = "features",
                      with_mean = TRUE,with_std = TRUE) %>%

   ml_decision_tree_regressor()


# Specify hyperparameter grid
grid <- list(
  decision_tree_regressor = list(
    max_depth = c(5,10,20,30)

  )
)

# Create the cross validator object
cv_TREE <- ml_cross_validator(
  sc, estimator = pipeline_TREE, estimator_param_maps = grid,
  evaluator = ml_regression_evaluator(sc),
  num_folds = 5,
  parallelism = 20
)


# Train the models
cv_model_TREE <- ml_fit(cv_TREE, train_sc_bis)

# Print the metrics
ml_validation_metrics(cv_model_TREE)

(b<-Sys.time()-a)
```

2.3 Un exemple de tuning de forêt aléatoire avec "sparklyr"

```
a<-Sys.time()


pipeline <- ml_pipeline(sc)
   for (i in 1:length(dummies)){
            pipeline<-pipeline %>%ft_string_indexer(input_col = dummies[i], output_col = paste(dummies[i],'1',sep=''))%>%
                                ft_one_hot_encoder(paste(dummies[i],'1',sep=''),paste(dummies[i],'2',sep=''))
               }
pipeline<-pipeline  %>%
   ft_vector_assembler(c(var_explicative,dummies2),
                           "features_unscaled") %>%
   ft_standard_scaler(input_col = "features_unscaled", output_col = "features",
                      with_mean = TRUE,with_std = TRUE) %>%

   ml_random_forest_regressor(num_trees = 500)


# Specify hyperparameter grid
grid <- list(
  random_forest = list(
    max_depth = c(6,8,10),
    feature_subset_strategy = c('sqrt','auto')
  )
)

# Create the cross validator object
cv <- ml_cross_validator(
  sc, estimator = pipeline, estimator_param_maps = grid,
  evaluator = ml_regression_evaluator(sc),
  num_folds = 5,
  parallelism = 15
)




# Train the models
cv_model_RF <- ml_fit(cv, train_sc_bis)

ml_validation_metrics(cv_model_RF)

(b<-Sys.time()-a)
```

2.4 Un exemple de tuning de gradient boosting avec "sparklyr"

```
a<-Sys.time()

pipeline_GBM <- ml_pipeline(sc)
   for (i in 1:length(dummies)){
            pipeline_GBM<-pipeline_GBM %>%ft_string_indexer(input_col = dummies[i], output_col = paste(dummies[i],'1',sep=''))%>%
                                ft_one_hot_encoder(paste(dummies[i],'1',sep=''),paste(dummies[i],'2',sep=''))
               }
pipeline_GBM<-pipeline_GBM %>%
   ft_vector_assembler(c(var_explicative,dummies2),
                           "features_unscaled") %>%
   ft_standard_scaler(input_col = "features_unscaled", output_col = "features",
                      with_mean = TRUE,with_std = TRUE) %>%

   ml_gbt_regressor()


# Specify hyperparameter grid
grid <- list(
  gbt_regressor = list(
    max_depth = c(2,4,6),
    step_size = c(0.1,0.01,0.05),
    max_iter = c(20,50,100)
  )
)

# Create the cross validator object
cv_GBM <- ml_cross_validator(
  sc, estimator = pipeline_GBM, estimator_param_maps = grid,
  evaluator = ml_regression_evaluator(sc),
  num_folds = 5,
  parallelism = 20
)


# Train the models
cv_model_GBM <- ml_fit(cv_GBM, train_sc_bis)

# Print the metrics
ml_validation_metrics(cv_model_GBM)

(b<-Sys.time()-a)
```

2.5 Récupération de l'importance des variables et visualisation

Pour cet exemple nous nous plaçons dans le cadre d'un gradient boosting.

```
modalite<-var_explicative
a<-character()
for (i in 0:(length(dummies2)-1)){
    n<-length(ml_stage(cv_model_GBM$best_model,2*i+1)$labels)
    a<-lapply(ml_stage(cv_model_GBM$best_model,2*i+1)$labels[-n], function(x) paste(dummies[i+1],x,sep='_'))
    modalite<-c(modalite,a)
}
#modalite
length(modalite)# Nous ne sommes pas certains que les modalités soient présnetes dans cet ordre là dans les varimp

length(ml_stage(cv_model_GBM$best_model,23)$feature_importances)
my_importance <- as.data.frame(cbind(modalite, ml_stage(cv_model_GBM$best_model,23)$feature_importances()))
colnames(my_importance) <- c('feature','importance')
my_importance  <- my_importance %>% arrange(-as.numeric(importance))
#my_importance


my_importance$importance<-unlist(my_importance$importance)
my_importance$feature<-unlist(my_importance$feature)


p<-ggplot(data=my_importance[1:30,], aes ( x = reorder(feature,importance) , y=importance))+
geom_bar(stat="identity",width=0.2,fill="#A91101")+ggtitle("Importance des variables GBM") +  ylab("Importance") + xlab("Variables")
p + coord_flip()
```