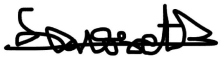
Mémoire présenté le :
pour l'obtention du diplôme
de Statisticien Mention Actuariat
et l'admission à l'Institut des Actuaires

Par : Monsieur Basile Saïsset-Costel

Titre du mémoire :
Décision de résiliation et sensibilité à la prime en assurance automobile

Confidentialité : ☐ NON ☒OUI (Durée : ☐ 1 an ☒2 ans)

Les signataires s'engagent à respecter la confidentialité indiquée ci-dessus.

Membres présents du jury de la filière : Signature :	Entreprise : Generali France Nom : LUO Yufei Signature : 
	Directeur de mémoire en entreprise Nom : Signature :
Membres présents du jury de l'Institut des Actuaires : 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 : 

Remerciements

Je souhaite tout d'abord remercier Christophe Mouren, manager de l'équipe Assurance de Dommages de m'avoir ouvert les portes de son équipe, et pour ses précieux conseils tout au long de la résiliation de ce mémoire.

Je tiens également à remercier Jérémie Togni et Yufei Luo, mes tuteurs au sein de Generali, de m'avoir accompagné au cours de mon alternance. Je tiens particulièrement à remercier Yufei Luo pour son suivi, son engagement et ses précieux conseils lors de la réalisation de ce mémoire.

J'adresse également mes remerciements à toute l'équipe assurance Dommages la direction des partenariats, et plus généralement toutes les personnes que j'ai pu fréquenter lors de mon alternance, pour leur accueil, leur aide et leur bienveillance à mon égard, en particulier je souhaiterais remercier toutes les personnes qui ont contribué à l'élaboration des différents outils et bases de données que j'ai utilisées au cours de mon alternance.

Je souhaite également remercier Olivier Lopez, mon tuteur académique, pour son aide et son suivi au cours de mon mémoire.

Enfin, je remercie ma famille et mes amis qui m'ont accompagné et soutenu durant la rédaction de ce mémoire.

Résumé

L'objet de ce mémoire est la modélisation de la résiliation dans le cadre de l'assurance automobile. Dans un contexte de forte concurrence connaître son portefeuille d'assuré et anticiper les conséquences des variations des tarifs et des modalités de son contrat est de plus en plus important pour les acteurs du marché. Nous avons donc créé et étudié différents modèles estimant la probabilité de résiliation en fonction notamment de la prime payée par les assurés ainsi que l'évolution de celle-ci.

Nous allons, dans un premier temps, utilisé la méthode de la régression logistique (GLM) afin d'avoir un modèle classique qui nous servira de base de comparaison pour la suite. Nous allons ensuite construire des modèles de *Machine Learning*, comme la méthode des forêts aléatoires, ou l'algorithme CatBoost afin de voir si l'utilisation de ces derniers était pertinente ou non.

Aussi, nous avons, à l'aide des modèles GLM, établis des profils de sensibilité à la résiliation. Enfin, nous avons créé un outil permettant d'établir une revalorisation optimale pour les assurés dans l'optique de les conserver en portefeuille.

Mots Clés — Résiliation, Assurance automobile, Régression logistique, GLM, Forêts aléatoires, Catboost, Classification binaire, Machine Learning, Sensibilité prix

Abstract

The topic of this memoir is the modelisation of contract termination in vehicle insurance. In a context of high competition to know the insured portfolio is highly important and so is to anticipate consequences of premium variation. So we built and studied several models expecting termination probability depending on the premium and its evolution.

We first will use a logistic regression in order to get a basis model to allow comparison. We will then use some Machine Learning techniques such as Random Forest or CatBoost to determine whether these are a good addition to this problem.

We also made sensitivity profiles using our GLM coefficients. At last, we made a tool we could use to find the best possible revaluation rate to retain clients in portfolio.

Keywords — Termination, Vehicle insurance, Logistic regression, GLM, Random Forest, Catboost, Binary classification, Machine Learning, Price sensitivity

Table des matières

Introduction

Dans le secteur hautement concurrentiel qu'est l'assurance automobile en France, la direction des partenariats de Generali distribue sous « marque blanche » des contrats d'assurances. Les assurés ayant un large choix d'offres assurantielles, il existe un turn-over important sur ce marché, un client mécontent de son assureur n'a pas de difficulté à en trouver un autre. De plus, le fait d'opérer en marque blanche, prive le porteur du risque d'un certain nombre d'outils lui permettant de mitiger ses risques. Dans ce contexte, il est primordial pour l'Équité comme pour tous les acteurs du secteur d'avoir la plus grande connaissance possible des assurés qu'ils ont en portefeuille.

Ainsi, mettre en place une modélisation des résiliations de contrat apparaît avantageux. Cela présente plusieurs avantages. Premièrement, cela permet de déterminer des profils plus ou moins sensibles à la résiliation et ainsi de mieux comprendre les comportements de nos assurés. Deuxièmement, cela permet de s'assurer que l'on ne conserve pas uniquement de mauvais risques en portefeuille au fil des années. Troisièmement, nous pouvons, à l'aide d'un tel modèle, estimer un résultat technique pour un portefeuille donné.

Le but de notre étude sera donc dans un premier temps de déterminer des profils d'assurés à risque en termes de résiliation et d'anticiper ce risque afin de pouvoir proposer, par la suite, des réponses adaptées à une population ciblée. Dans un second temps, nous créerons un outil permettant d'estimer le résultat technique à la suite des revalorisations décidées en accord avec les partenaires.

Pour ce faire, nous allons utiliser une méthode classique de modélisation : la régression logistique. Par la suite, nous utiliserons des méthodes d'apprentissage automatique (ou *Machine Learning*). Ces dernières sont moins courantes dans la science actuarielle. Malgré une nette percée au cours des dernières années, elles restent sous-utilisées en pratique. Cela est en partie dû au RGPD qui impose un obligation de transparence et d'explicabilité aux assureurs. Ainsi, nous allons mettre en place des outils permettant d'interpréter les modèles de *Machine Learning*.

Les données utilisées proviennent de plusieurs partenaires. Nous y retrouvons des assurés "standard" et "malussé" et comprennent toutes les variables utiles dans le cadre de la tarification d'un contrat d'assurance automobile. Nous avons fait le choix de séparer en deux modélisations distinctes ces périmètres. Ainsi nous pourrons les comparer et observer d'éventuelles différences entre ces derniers.

Nous allons donc modéliser la résiliation sur chaque périmètre à l'aide de différentes méthodes. L'objectif sera alors de définir des profils d'assurés sensibles à la résiliation. Ainsi, nous pourrons adapter les revalorisations annuelles discutées avec les partenaires dans le but d'optimiser le résultat technique obtenu en fin de période d'étude.

Première partie

Contexte du mémoire

Chapitre 1

Marché de l'assurance automobile et résiliation en France

Dans ce mémoire, nous nous concentrerons sur le marché de l'assurance de biens et de responsabilité, et plus précisément sur le marché de l'assurance automobile.

Dans ce chapitre, nous présentons, dans un premier temps le marché de l'assurance de biens et responsabilités dans sa globalité puis nous nous intéressons plus précisément au marché de l'assurance automobile, cadre de notre étude.

1.1 Le marché de l'assurance de biens et responsabilités

Nous verrons ici un certain nombre d'indicateurs clés de l'assurance de biens et responsabilités. Rappelons d'abord que l'assurance de biens et responsabilités est obligatoire en France et que cela explique que ce marché soit très important et drague 63,2 milliards d'euros de cotisations en 2021[1]. Ces cotisations sont réparties comme suit :

Cotisations

Md€	2017	2018	2019	2020	2021	21/20
Ensemble	**54,6**	**56,2**	**58,7**	**60,2**	**63,2**	**+ 4,9 %**
Automobile	21,4	22,1	22,8	23,5	24,1	+ 2,5 %
Biens des particuliers	10,5	10,8	11,3	11,7	12,1	+ 3,4 %
Biens des professionnels et agricoles	7,6	7,9	8,2	8,2	8,7	+ 5,9 %
Responsabilité civile générale	3,6	3,7	3,9	3,8	4,3	+ 11,7 %
Construction	2,1	2,2	2,3	2,3	2,6	+ 16,6 %
Catastrophes naturelles	1,6	1,6	1,7	1,7	1,8	+ 3,4 %
Transports	0,9	0,9	1,1	1,1	1,3	+ 13,2 %
Autres assurances (1)	6,8	7,0	7,5	7,9	8,3	+ 5,5 %

(1) Protection juridique, assistance, pertes pécuniaires et crédit-caution.

FIGURE 1.1 – Répartition des cotisations en Assurance de Biens et Responsabilités (source : FA[1])

Selon la figure 1.1, on observe, une augmentation faible mais constante du montant des cotisations. Celle-ci peut être due à des facteurs exogènes au monde de l'assurance, tels que l'inflation, par exemple. Aussi, si cette

augmentation est faible, cela est probablement lié à la concurrence extrême que se livrent les différents acteurs de ce marché, notamment sur les segments de l'assurance habitation et de l'assurance automobile.

1.2 Le marché de l'assurance automobile

Si l'on se focalise plus précisément sur le marché de l'assurance automobile, on observe la même faible augmentation des cotisations pour l'automobile.

Cotisations selon le type de véhicule assuré

M€	2017	2018	2019	2020	2021	21/20
Cotisations	**21 387**	**22 124**	**22 807**	**23 496**	**24 084**	**+ 2,5 %**
Véhicules de 1ère catégorie (1)	17 109	17 562	18 072	18 523	18 949	+ 2,3 %
Véhicules de 3e catégorie (2)	936	1 066	1 108	1 147	1 199	+ 4,5 %
Autres véhicules (3)	1 262	1 310	1 325	1 414	1 411	- 0,2 %
Flottes d'entreprises	2 080	2 186	2 302	2 412	2 525	+ 4,7 %

(1) Véhicules 4 roues de moins de 3,5 t hors flottes.
(2) Principalement des 2 roues motorisés hors flottes.
(3) Véhicules agricoles, véhicules de deuxième catégorie, missions, remorques, caravanes, engins spéciaux, polices garages, etc…

FIGURE 1.2 – Répartition des cotisations en assurance auto (source : FA[1])

.

Ces données peuvent être mises en relation avec la taille du parc automobile assuré français.

Parc des véhicules assurés

M de véhicules au 31/12	2017	2018	2019	2020	2021	21/20
Véhicules de 1ère catégorie (1)	42,2	42,6	43,0	43,5	44,2	+ 1,4%
Véhicules de 3e catégorie (2)	4,0	4,4	4,5	4,7	4,8	+ 3,7%
Autres véhicules (3)	2,9	2,9	2,9	2,9	3,0	+ 1,4%
Flottes d'entreprises	3,9	4,1	4,4	4,6	4,8	+ 5,0%
Ensemble Automobile	**52,9**	**54,0**	**54,8**	**55,7**	**56,8**	**+ 1,9%**

(1) Véhicules 4 roues de moins de 3,5 t hors flottes. (2) Principalement des 2 roues motorisés hors flottes.
(3) Véhicules agricoles, véhicules de deuxième catégorie, missions, remorques, caravanes, engins spéciaux, polices garages, etc.

FIGURE 1.3 – Parc automobile assuré (source : FA[1])

Nous pouvons constater sur les tableaux 1.2 et 1.3 que les cotisations de la branche assurance automobile n'augmente pas beaucoup au cours des cinq dernières années. De la même façon, le nombre de véhicules assurés connaît une faible augmentation. Ainsi, la faible augmentation du nombre de véhicules assurés est une des raisons qui expliquent la faible augmentation des cotisations en assurance automobile. On peut aussi citer la forte concurrence de ce marché sur lequel près de 100 acteurs opèrent[2].

Nous avons vu dans ce chapitre, que le marché de l'assurance incendies accidents risques divers (IARD) et notamment le marché automobile, sont en faible augmentation. On peut notamment observer cela en s'intéressant aux cotisations ou au nombre d'automobiles assurées en France. Ainsi, le marché de l'assurance automobile n'évolue pas beaucoup sur les dernières années mais voit tout de même apparaître de nouveaux acteurs. Dans ce contexte, les acteurs de ce marché souhaiteront connaître au mieux leur portefeuille d'assuré. Dans ce cadre, l'étude de la résiliation est intéressante car elle permet de comprendre quels profils d'assurés sont davantage sensibles à la résiliation et ainsi de prendre des mesures particulières pour ces derniers si ce sont des profils rentables.

1.3 Résiliation du contrat en assurance automobile

En ce qui concerne la résiliation en assurance automobile, on peut avant tout parler des évolutions dans la législation. Ensuite, nous regarderons quelques chiffres du marché concernant la résiliation.

1.3.1 Changements de législation

Au cours des 20 dernières années les conditions de résiliation ont largement évolué concernant les contrats d'assurance automobile. Tout d'abord en 2005, la *loi tendant à conforter la confiance et la protection du consommateur* dite « loi Châtel »[18] est promulguée. L'objectif de cette loi est de simplifier la résiliation de certains contrats. Rappelons qu'à cette époque la résiliation ne peut se faire qu'à l'anniversaire du contrat, cependant les assurés ne connaissaient pas nécessairement cette date, et se rendaient souvent compte trop tard de leur possibilité de résilier leur contrat. Cette loi impose donc à l'assureur d'envoyer un avis d'échéance au moins 15 jours avant la date limite de résiliation. Ainsi l'assuré est prévenu de l'échéance à venir, et averti que d'une éventuelle hausse de sa prime. Il peut donc plus facilement renoncer à la tacite reconduction de son contrat et le résilier.

En 2014, est promulguée la loi *du 17 mars 2014 relative à la consommation*, dite « loi Hamon »[19]. Cette dernière a pour objectif de rééquilibrer le pouvoirs entre consommateurs et professionnels. Dans le cadre de l'assurance de dommages et responsabilités, elle permet de résilier encore plus simplement qu'auparavant son contrat et offre notamment la possibilité de le résilier à tout moment à partir du premier anniversaire, et plus uniquement lors de l'échéance annuelle du contrat.

La loi Hamon a été l'objet de nombreux mémoires au moment de sa promulgation, car les acteurs du marché de l'assurance de biens et responsabilités, craignaient une forte hausse des résiliations et une plus grande volatilité du portefeuille d'assurés. On peut noter qu'en effet, les chiffres de résiliations ont quelque peu augmenté en 2015 pour se stabliliser à partir de 2016.

Enfin, au cours des années les conditions de résiliations ont été plusieurs fois modifiées dans l'optique de faciliter la résiliation et/ou d'offrir plus de vecteurs permettant de rompre le contrat aux assurés. Nous pouvons notamment citer la résiliation en trois clics apparue en 2023.

1.3.2 Modalités de résiliation

Dans le cadre d'un contrat d'assurance IARD, à la fois l'assureur et l'assuré peuvent rompre le contrat. Néanmoins certaines conditions doivent être réunies et la résiliation est strictement encadrée par le Code des assurances. On verra dans le tableau ci-après que certaines conditions de résiliation sont réservées aux assurés et d'autres aux assureurs.

Ajoutons à cela que certains contrats disposent d'une clause permettant aux assurés de résilier en cas de hausse de tarif. Cependant, avec la possibilité de résilier à tout moment permise par la loi Hamon, une telle clause semble désormais obsolète.

	Possibilité de résilier		Source
Motif de Résiliation	Assureur	Assuré	Code des assurances
Opposition à la tacite reconduction (loi Châtel)	Non	Oui	L113-15-1
À tout moment après le 1^{er} anniversaire (Loi Hamon)			L113-15-2
À l'échéance	Oui		L113-12
Évolution du risque			L113-2 et L113-4
Après sinistre			R*113-10 et L191-6
Changement de situation/décès/aliénation du bien			L113-16, L121-9, L121-10 et L121-11
Non paiement	Oui	Non	L113-3
Omission ou déclaration inexacte			L113-9

TABLEAU 1.1 – Motifs de résiliation disponibles pour les assurés et pour les assureurs

1.3.2.1 Motifs réservés à l'assuré

- Opposition à la tacite reconduction :

L'assureur doit envoyer à tout assuré deux mois avant l'échéance du contrat un avis rappelant la date d'échéance, la future prime et l'informant de la possibilité de résilier à échéance ainsi que les conditions pour ce faire. Cet article du Code des assurances a été introduit par la loi Châtel précédemment évoquée.

- À tout moment après le 1^{er} anniversaire :

L'assuré peut, grâce à cet article, résilier à tout moment son contrat, à partir d'un an de contrat échu. Cette résiliation ne peut entraîner de frais supplémentaires pour l'assuré, qui n'est redevable que de la partie de prime correspondant à la période ou le risque est couvert. Cet article a lui fait son apparition dans le Code des assurances en 2014, avec la loi Hamon évoquée plus haut.

1.3.2.2 Motifs communs à l'assuré et à l'assureur

- À échéance :

Cette disposition permet aux deux parties de ne pas reconduire le contrat après l'échéance d'une année de contrat. Néanmoins, une lettre recommandée doit être envoyée au moins deux mois avant l'anniversaire du contrat. Cet article a plusieurs fois été amendé. Dans sa première version en 1976, il ne permettait de résilier que tous les trois ans, puis à partir de 1990, il permettait de résilier tout les ans. Rappelons que cela a été le seul moyen de résilier son assurance pendant plus d'une décennie.

- Évolution du risque :

Si le risque d'un assuré s'aggrave, alors l'assureur est en droit de dénoncer le contrat, ou de proposer une hausse tarifaire en conséquence de l'aggravation du risque. Néanmoins, ces actions doivent être réalisées 10 jours après la notification de changement de situation. A contrario, si le risque diminue, l'assuré est en droit de demander une baisse de sa prime. Si cette baisse n'est pas accordée, alors l'assuré peut dénoncer le contrat et la résiliation prendra effet 30 jours après la dénonciation.

- Après sinistre :

Plusieurs cas de figures existent en cas de sinistre :

  * Si l'assuré a un sinistre alors qu'il est sous l'emprise de l'alcool ou de stupéfiants, l'assureur pourra résilier le contrat.
  * Si la police le prévoit, l'assureur peut résilier un contrat d'assurance en cas de sinistres. Aussi si l'assureur décide de résilier le contrat, l'assuré, s'il est multi assuré, peut demander la résiliation de ses autres contrats.
  * Enfin des dispositions particulières existent pour les assurés du Haut-Rhin, du Bas-Rhin et de Moselle en France métropolitaine ou encore pour les assurés de certains DROM tels que Wallis et Futuna. Ces dispositions permettent aux assurés de résilier leur contrat après qu'ils aient eu un sinistre.

- Changement de situation :

Si l'assuré déménage, se marie ou change de profession, par exemple, alors l'assuré et l'assureur pourront résilier le contrat si *il a pour objet la garantie de risques en relation directe avec la situation antérieure et qui ne se retrouvent pas dans la situation nouvelle.* Cette résiliation se fait sans frais pour l'assuré qu'il en soit à l'initiative ou non, et doit avoir lieu dans les 3 mois suivants la date de l'évènement.

- Décès de l'assuré, aliénation du bien :

Si l'assuré décède ou qu'il perd son bien (qu'il soit vendu, retiré par la police ou même détruit) l'assurance continue pour l'héritier ou l'acquéreur qui aura à charge d'exécuter les obligations inhérentes à ce contrat. Néanmoins, l'héritier, l'acquéreur et l'assureur peuvent résilier le contrat.

1.3.2.3 Motifs réservés à l'assureur

- Non-paiement de la prime :

Si l'assuré ne paye pas sa prime dans les 10 jours suivants l'échéance du contrat, alors l'assureur peut le mettre en demeure. Puis s'il ne paye toujours pas, 30 jours après la mise en demeure, la garantie peut-être suspendue. 10 jours après cette suspension, l'assureur peut résilier le contrat. Si les primes sont payées, alors le contrat reprend ses droits à midi le lendemain du jour où les primes ont été payées.

- Omission ou déclaration inexacte :

En cas d'omission ou de déclaration inexacte, si la mauvaise foi de l'assuré n'est pas établie, alors l'assureur peut résilier le contrat, ou proposer une hausse de prime qui doit être acceptée par l'assuré.

1.4 Données du marché

Nous pouvons nous intéresser aux données de France Assureurs[2] quant à la résiliation de contrat sur le marché automobile.

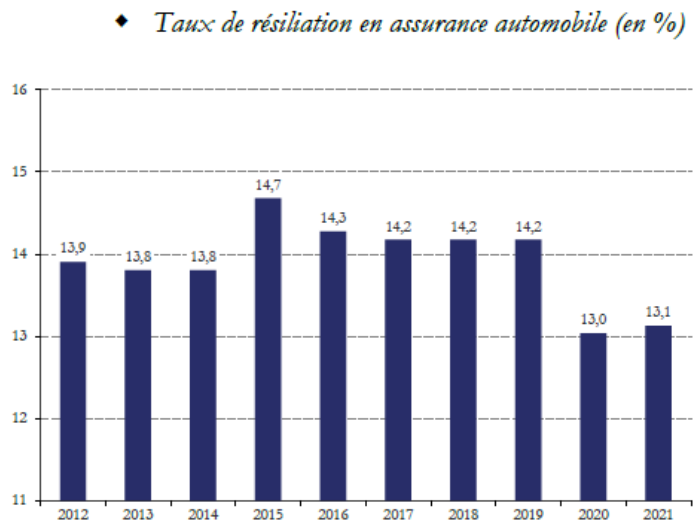


FIGURE 1.4 – Taux de résiliation en assurance automobile des particuliers (source : FA[2])

Nous observons un augmentation du nombre de résiliation en 2015, année de promulgation de la *loi Hamon*. Le taux de résiliation de 2014 est de 13,8 % et de 14,7 % en 2015, puis il se stabilise entre 2016 et 2019 à un niveau légèrement supérieur à 14 %. Nous pouvons noter des taux nettement inférieurs en 2020 et 2021 mais ces données sont à relativiser en raison du contexte particulier de ces années.

Nous avons précédemment évoqué la relative stabilité du parc automobile français, ainsi, il est probable que la majorité des résiliations soient liées à la volonté des assurés de changer d'assureur, et non pas à une volonté d'abandonner la voiture pour s'orienter vers d'autres moyens de locomotions. Nous pouvons aussi noter que le taux de couverture [1] des contrats en 2021 est de 119,7 %[2], le nombre de contrat total augmente donc légèrement et une partie importante des affaires nouvelles est vraisemblablement liées à des résiliations.

Cela met en avant l'importance de mener une politique de rétention efficace pour les assureurs afin de conserver les assurés ayant un profil rentable en portefeuille.

1. Taux d'affaires nouvelles/Taux de résiliations

Chapitre 2

L'assurance de biens et responsabilités à l'Équité

Ce mémoire a été réalisé au sein d'une filiale de Generali France, l'Équité. Cette filiale propose des solutions d'assurance sur mesure à ses partenaires. Ces domaines d'activités comptent notamment l'assurance de personne, l'assurance de dommages, la protection juridique ou encore l'assurance emprunteur. Les partenaires de l'Équité sont assez diversifiés puisque qu'il peut s'agir de professionnels de la distribution d'assurance, ou encore de grandes enseignes joignant des contrats d'assurance à la vente de certains biens, par exemple.

La gestion des contrats et des sinistres étant à la charge du partenaire, le rôle de la direction des partenariats dans le relation assureur/partenaire est de veiller au bon fonctionnement des partenariats existants par multiples suivis, ou la maintenance des documents contractuels. La direction a aussi à charge d'étudier de potentiels nouveaux partenariats.

Le principal moyen d'action de l'Équité, en cas de résultats décevant d'un partenaire, est la revalorisation à échéance des contrats.

2.1 L'assurance auto au sein de l'Équité

L'équipe Assurance Dommages de la direction des partenariats propose aux partenaires commerciaux différents produits d'assurance, notamment multirisque habitation (MRH), produit affinitaires de masse (PAM), protection juridique (PJ) ou encore automobile. Nous allons, dans la suite, développer plus avant les produits auto proposés par l'Équité.

L'Équité propose un large choix de produits d'assurance auto :

- Auto Standard : véhicules quatre roues, engins de déplacements personnels motorisés (EDPM), véhicules sans permis (VSP) etc. ;
- Auto Véhicules de collection : véhicule avec carte grise de collecion ;
- Auto Véhicules haute de gamme : véhicule quatre roues de plus de 50 000€ ;
- Auto Risques aggravés : véhicules quatre roues avec risque alcoolémie, non-paiement, malussé ou forte sinistralité etc. ;
- Taxi/VTC : véhicules quatre roues ;
- Moto Standard : moto deux ou trois roues, quad ;
- Moto Risques aggravées : moto deux ou trois roues, quad avec risque alcoolémie, non-paiement, malussé ou forte sinistralité etc.

Les produits principaux sont les produits auto standard et aggravé ainsi que le produit moto standard. Ils représentent plus de 90 % du chiffre d'affaire de l'Équité. Les deux principaux modes de distribution sont la vente par internet et par grossiste qui représentent près de 75 % du CA.

Pour ce mémoire, nous nous intéresserons particulièrement aux périmètres auto standard et aggravé (ci-après malussé) que nous traiterons de manière distincte. Comme évoqué précédemment, l'objectif de ce mémoire est d'anticiper les résiliations des assurés qui feraient suite à l'augmentation de leurs primes d'assurance à la suite d'une revalorisation annuelle décidée par Generali. Cela permettrait de mieux connaître le comportement des assurés, notamment en établissant des profils plus ou moins sensibles à la résiliation, et d'enlever une part d'incertitude sur les prochains résultats économiques attendus. En effet, s'il est possible d'anticiper la composition du portefeuille en année N+1 selon différents scénarios de hausses de primes, alors il sera plus aisé de choisir les paramètres optimums. Par exemple les profils moins sensibles à la hausse de prime pourrait subir une hausse plus importante que les autres.

Deuxième partie

Éléments théoriques

Chapitre 3

Notions transverses à la création de modèles

Dans cette partie, nous allons présenter les outils théoriques qui seront utilisés tout au long de ce mémoire. Tout d'abord, nous verrons, dans ce chapitre, différents outils et généralités sur les modèles statistiques. Dans un second temps, nous nous intéresserons aux éléments préalables à la modélisation des données. Ensuite, nous verrons la théorie des modèles linéaires généralisés, puis la théorie de différents modèles d'apprentissage automatique. Enfin, nous définirons les différents outils qui serviront à l'évaluation de nos modèles.

3.1 Définition du sous-apprentissage et du sur-apprentissage

Les notions de « *sur-apprentissage* » et de « *sous-apprentissage* » sont fondamentales dans l'étude des modèles prédictifs. Ces notions sont étroitement liées au biais et à la variance du modèle. Nous allons donc définir ces notions et mettre en exergue le lien entre elles.

3.1.1 Définitions

3.1.1.1 Biais/variance

Soit $\hat{f}(x)$ un modèle prédictif, on définit le biais comme étant la différence entre la valeur estimée par le modèle et la vraie valeur observée.

$$Biais(\hat{f}(x)) = \mathbb{E}(\hat{f}(x) - y) \tag{3.1}$$

Le biais mesure donc l'erreur du modèle ; un biais important signifie donc que le modèle est trop simple.
La variance du modèle $\hat{f}(x)$ se définit comme une mesure de la dispersion des valeurs estimées par le modèle.

$$Var(\hat{f}(x)) = \mathbb{E}\left[(\hat{f}(x) - \mathbb{E}(\hat{f}(x)))^2\right] \tag{3.2}$$

On peut interpréter la variance comme étant la capacité d'un modèle à saisir les petites fluctuations du jeu de données.

Nous pouvons illustrer les caractéristiques du biais et de la variance sur la qualité du modèle à l'aide du graphique 3.1 ci-dessous. On observe que le biais du modèle est principalement dû à la qualité de prédiction du modèle, alors que la variance est bien une mesure de la dispersion des prédictions du modèle.

On peut également observer qu'un bon modèle semble être un modèle dont le biais et la variance sont faibles. Nous allons donc chercher à minimiser ces deux variables.

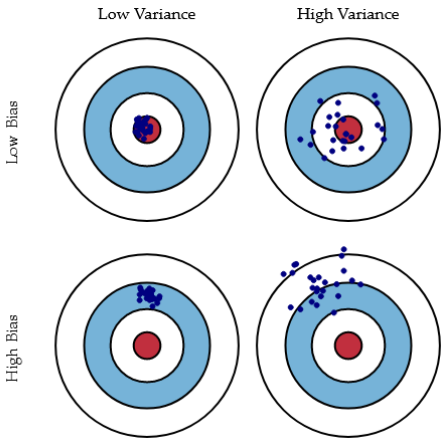


FIGURE 3.1 – Illustration graphique de différentes combinaisons Biais/Variance (source : Scott Fortmann-Roe[17])

3.1.1.2 Compromis biais/variance

Nous cherchons donc à minimiser le biais et la variance du modèle dans le but de minimiser l'erreur du modèle. Néanmoins minimiser une des deux quantités revient à faire augmenter l'autre, ce dilemme est appelé *compromis biais-variance*. Pour répondre à ce modèle, nous pouvons faire varier la complexité du modèle afin de minimiser l'erreur totale du modèle. Un modèle de faible complexité aura un fort biais car il prédira mal les données, cependant sa variance sera faible. A contrario, un modèle très complexe captera bien les particularités des données et aura donc un biais très faible, toutefois, sa variance aura tendance à largement augmenter car le modèle se calquera *trop* bien sur les données et s'adaptera mal aux nouvelles données. Cela peut se comprendre à l'aide de la figure 3.2

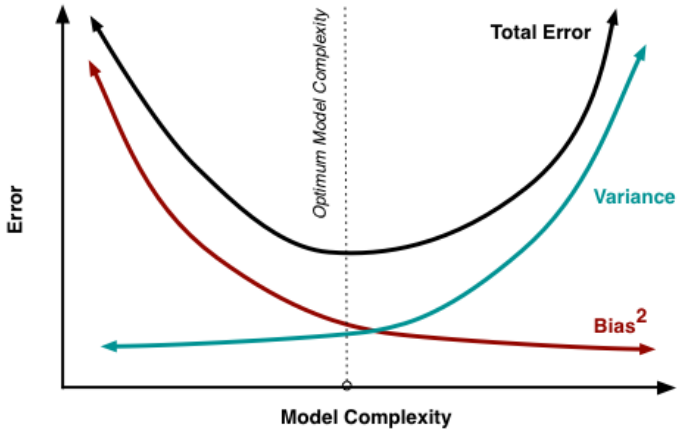


FIGURE 3.2 – Illustration du dilemme biais/variance (source : Scott Fortmann-Roe[17])

Nous allons maintenant faire le lien entre biais et variance et les notions de sous- et sur-apprentissage.

3.1.1.3 Sous-apprentissage

On parle de *sous-apprentissage*, ou *underfitting* en anglais, lorsqu'un modèle ne saisit pas bien la structure sous-jacente des données. Le modèle sera alors peu performant et présenter un biais important. Ce problème intervient lorsque le modèle ne s'adapte pas assez aux données car il n'est pas assez complexe. Le sous-apprentissage peut, par exemple, intervenir si l'on essaye d'ajuster un modèle linéaire à un problème non linéaire.

3.1.1.4 Sur-apprentissage

À l'inverse, on parlera de *sur-apprentissage*, ou *overfitting* en anglais, lorsque le modèle est trop précis par rapport aux données et capte aussi le *bruit*, c'est-à-dire les variations des données qui ne sont pas explicables

par le modèle, du jeu de données. En général, ce problème survient car le modèle construit est trop complexe par rapport au problème sous-jacent. Un modèle souffrant de sur-apprentissage aura une grande variance, car il aura du mal à interpréter de nouvelles données.

3.1.1.5 Résumé

Pour résumer, lors de la création du modèle, il faudra prendre garde aux problématiques de sous- et sur-apprentissage qui peuvent être mesurées une fois le modèle créé. Il faudra donc revenir sur certains modèles pour en changer certains hyper-paramètres dans le but de réduire les phénomènes de sur- et de sous-apprentissage. On peut illustrer cela à l'aide du graphique 3.3 :

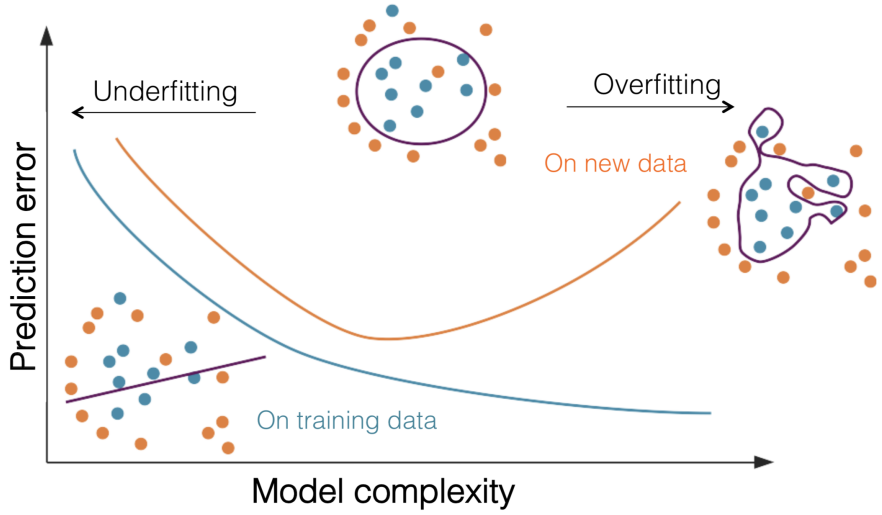


FIGURE 3.3 – Illustration des phénomènes de sur- et sous-apprentissage (source : cours de Mme Boyer [4]

Sur ce graphique, nous pouvons observer un exemple de sous-apprentissage en bas à gauche. Il est clair que le modèle linéaire n'est pas adapté à ces données et donne de mauvais résultats. En haut à droite, nous pouvons observer un exemple de modèle trop bien ajusté, ou le modèle colle parfaitement aux données mais présente une grande complexité. Enfin, en haut au centre du graphique, nous pouvons observer un modèle bien ajusté où l'on tolère une certaine erreur, car deux éléments sont mal classés. Le modèle est, toutefois, bien moins complexe que l'exemple de sur-apprentissage.

En pratique, si le modèle a un mauvais pouvoir prédictif que ce soit sur l'échantillon d'entraînement ou sur l'échantillon de test [1], alors c'est que son biais est élevé et qu'il souffre de sous-apprentissage. Aussi, si l'on observe que l'erreur d'entraînement est bien supérieure à l'erreur de test, alors le modèle a vraisemblablement une grande variance et il souffre de sur-apprentissage.

3.2 Validation croisée

Lors de la création d'un modèle de prédiction, il est important de prendre garde à sa capacité de généralisation. On dit qu'un modèle se généralise s'il est capable de bien prédire des données qui n'ont pas servi à son entraînement. Un modèle qui est en sur-apprentissage pourra, par exemple, avoir une erreur de généralisation très importante puisque, comme nous l'avons vu, il s'adapte trop bien aux données d'entraînement et perd sa capacité de généralisation.

Pour répondre à cette problématique, nous allons utiliser différentes méthodes de « *validation-croisée* », ou « *cross-validation* » en anglais. Le principe général de la *validation-croisée* est de contrôler la qualité d'un modèle en sous-échantillonnant le jeu de données pour permettre une estimation de l'erreur de généralisation. Nous allons détailler, par la suite trois méthodes courantes de *cross-validation*.

1. Dans le cadre de la modélisation, il est important de définir une base d'entraînement du modèle et une base de validation (ou de test),la première va servir à entraîner le modèle et la seconde servira notamment à tester si le modèle est bien calibré et ne souffre pas de sur- (ou de sous-) apprentissage.

3.2.1 Exemple de méthodes de *validation-croisée*

- La méthode la plus classique de *validation-croisée* est la méthode de validation « *Hold-Out* » qui consiste à séparer le jeu de données D_n, de taille n, en deux sous-échantillons indépendants, communément appelés la base d'*entraînement* et la base de *validation* du modèle (respectivement *train* et *test* en anglais). Dans la méthode *hold-out*, le modèle est calibré sur le jeu d'entraînement et son erreur de généralisation est estimée sur le jeu de données de validation.
- La méthode de la *k-fold cross-validation* consiste à créer k sous-échantillons du jeu de données. Chaque sous-échantillon servira à calculer l'erreur de généralisation du modèle ajusté sur les *k-1* autres échantillons. On obtient donc k scores d'erreur. Et l'erreur de généralisation globale du modèle est estimée en faisant la moyenne de ces k erreurs.

FIGURE 3.4 – Exemple de *k-Fold Validation-Croisée* pour k = 10 (source : site web)

- La dernière méthode que nous développerons est la méthode dite « *Leave-One-Out* », qui est un cas particulier de la *k-fold cross-validation*, où l'on crée autant d'échantillons qu'il y a de données ($k = n$).

3.2.2 Application à la sélection de paramètres

On utilisera aussi la *validation-croisée* pour optimiser les paramètres du modèle. Prenons l'exemple des forêts aléatoires, où l'on peut discuter du nombre d'arbres à agréger, par exemple. On pourra alors entraîner le modèle avec différents paramètres et comparer les résultats des modèles en fonction du paramètre étudié en comparant notamment l'erreur d'entraînement à l'erreur de validation. En effet, en traçant les courbes d'erreur en fonction, par exemple, du nombre d'arbres, on pourra observer plus simplement à partir de quelle quantité d'arbres entraînés le modèle est en sur-apprentissage.

Chapitre 4

Étapes préliminaires à la modélisation des données

Dans ce chapitre, nous verrons différents outils utilisés pour préparer les données à la modélisation.

Pour des raisons d'interprétabilité des résultats et de facilité de modélisation, nous pouvons faire le choix de regrouper en classes certaines variables présentes dans la base de données. Nous nous intéresserons particulièrement aux corrélations entre variables qualitatives dont nous définirons les méthodes de calcul ci-après. De plus, certaines variables qualitatives présentant un grand nombre de modalités ont aussi pu être regroupées grâce à un algorithme. Nous présenterons en particulier dans ce chapitre : le χ^2, le V de Cramer, la méthode de la classification ascendante hiérarchique (CAH) et enfin l'algorithme des K-Moyennes.

4.1 Analyse des corrélations entre variables qualitatives

Pour analyser les corrélations nous avons choisi d'utiliser la méthode du V de Cramer, qui a pour avantage de fournir une mesure de l'inter-corrélation entre deux variables et de permettre un affichage graphique explicitant facilement ces corrélations.

Pour calculer le V de Cramer, il faut tout d'abord calculer la statistique du χ^2 de Pearson.

4.1.1 χ^2 de Pearson

Le test statistique du χ^2 de Pearson est une mesure déstinée à évaluer à quel point les écarts entre 2 variables est due au hasard. Il se calcule dans notre cas de la façon suivante :

Soit un échantillon de taille n, des variables A et B, indépendamment distribuées avec respectivement r et k modalités.
On a donc les fréquences n_{ij} = du nombre de fois où la valeur (A_i, B_j) a été observé pour $i = 1, \ldots, r$ *et* $j = 1, \ldots, k$, que nous pouvons résumer dans le tableau 4.1.

	B_1	B_2	$\ldots$	B_j	$\ldots$	B_k	Total
A_1	$n_{1,1}$	$n_{1,2}$	$\ldots$	$n_{1,j}$	$\ldots$	$n_{1,k}$	$n_{1,.}$
A_2	$n_{2,1}$	$n_{2,2}$	$\ldots$	$n_{2,j}$	$\ldots$	$n_{2,k}$	$n_{2,.}$
$\ldots$	$\ldots$	$\ldots$	$\ldots$	$\ldots$	$\ldots$	$\ldots$	$\ldots$
A_i	$n_{i,1}$	$n_{i,2}$	$\ldots$	$n_{i,j}$	$\ldots$	$n_{i,k}$	$n_{i,.}$
$\ldots$	$\ldots$	$\ldots$	$\ldots$	$\ldots$	$\ldots$	$\ldots$	$\ldots$
A_r	$n_{r,1}$	$n_{r,2}$	$\ldots$	$n_{r,j}$	$\ldots$	$n_{r,k}$	$n_{r,.}$
Total	$n_{.,1}$	$n_{.,2}$	$\ldots$	$n_{.,j}$	$\ldots$	$n_{.,k}$	n

TABLEAU 4.1 – Contingence du χ^2

On peut donc obtenir la formule de la distance du χ^2 ci-dessous :

$$\chi^2 = \sum_{i,j} \frac{(n_{ij} - \frac{n_{i.}n_{.j}}{n})^2}{\frac{n_{i.}n_{.j}}{n}} \tag{4.1}$$

Si cette mesure est égale à 1, alors les variables sont parfaitement dépendantes. A contrario, si elle est nulle, alors les variables sont parfaitement indépendantes.

Pour calculer le V de Cramer, on a aussi besoin du χ^2_{max} qui se définit comme suit :

$$\chi^2_{max} = n \times min(r-1, k-1) \tag{4.2}$$

4.1.2 V de Cramer

Le V de Cramer permet de calculer la dépendance entre 2 variables, s'il est égal à 1, alors les variables sont parfaitement corrélées, et s'il est égal à 0, elles ne le sont pas du tout. Il se calcule de la façon suivante :

$$V = \sqrt{\frac{\chi^2}{\chi^2_{max}}} \tag{4.3}$$

4.2 Méthodes de classification

Nous verrons ici deux méthodes de classification de variables. La première, la méthode des K-Moyennes va nous permettre de regrouper en classes plus ou moins homogènes les variables continues de notre jeu de données. La deuxième méthode, la classification ascendante hiérarchique, ou CAH, permettra de faire des regroupements hiérarchiques des données.

4.2.1 Méthode des K-Moyennes

La méthode des K-Moyennes est un algorithme de partitionnement des données. L'objectif est de diviser un ensemble de n points en k groupes en minimisant la distance entre les points à l'intérieur de chaque classe.

Soit un ensemble de points $X = x_1, \ldots, x_n$. On souhaite donc partitionner ces n points en k ensembles $S = S_1, S_2, \ldots, S_k$ avec $k \leq n$ en minimisant la distance des points à l'intérieur de la classe :

$$\arg\min_{\mathbf{S}} \sum_{i=1}^{k} \sum_{\mathbf{x}_j \in S_i} \|\mathbf{x}_j - \boldsymbol{\mu}_i\|^2 \tag{4.4}$$

avec μ_i le barycentre des points dans S_i.

4.2.2 Classification ascendante hiérarchique

La CAH fait partie des méthodes de classification automatique de données. Dans cette méthode, les individus sont initialement seuls dans une classe, puis ils sont rassemblés en classes de plus en plus grandes, jusqu'à ce qu'ils soient tous dans une classe unique. On qualifie cet algorithme d'ascendant car l'on part d'un jeu de données parfaitement segmenté avec un individu par classe et que l'on "remonte" en regroupant les données jusqu'à n'obtenir qu'une seule classe. Aussi, les classes ayant des individus en commun sont forcément incluses l'une dans l'autre, c'est pourquoi on parle de méthode hiérarchique. Pour classer les différents individus, la méthode a besoin d'une mesure de dissimilarité entre les individus qui peut, si l'on est dans un espace euclidien, être la distance entre les individus.

Le principe de la classification ascendante hiérarchique est le suivant : On a initialement n classes et on cherche un $nb_{classes}$ qui soit strictement inférieur à n. Cette recherche se fait itérativement en regroupant à chaque étape les 2 classes dont la dissimilarité est la plus petite, on appelle cette valeur l'*indice d'agrégation*. Cet indice est logiquement croissant dans la mesure où l'on agrège d'abord les points les plus proches et que l'on termine par des classes qui peuvent être très éloignées.

Plusieurs méthodes de calcul de la dissimilarité sont disponibles, ici, nous utiliserons la distance de Ward, qui définit la distance entre deux groupes comme la distance entre leurs centres de gravité pondérée par le produit du nombre d'individus dans chaque groupe divisé par leurs sommes. On cherche donc à maximiser l'inertie inter-classe, c'est-à-dire, en considérant deux classes C_1 et C_2 :

$$dissim(C_1,\ C_2) = \frac{n_1 * n_2}{n_1 + n_2}(dissim(G_1,\ G_2)), \tag{4.5}$$

avec n_1 et n_2 le nombre d'éléments de chaque classe et G_1 et G_2 leurs centres de gravité respectifs.

L'un des avantages de cette classification est d'avoir une représentation graphique efficace avec les dendrogrammes (voir figure 4.1). Ces derniers permettent, en effet, d'observer graphiquement un nombre de classes qui serait approprié pour le problème sous-jacent.

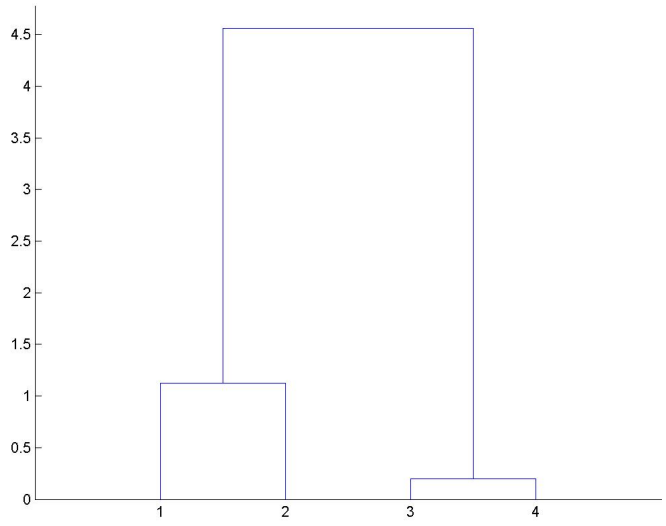


FIGURE 4.1 – Exemple de Dendrogramme

4.2.3 Choix du nombre de classe optimal

Pour choisir le nombre k de classe optimal, nous utiliserons une méthode populaire, appelée « *méthode du coude* ». Cette dernière permet de déterminer graphiquement un k optimal. Pour cela on représente la part de variation expliquée en fonction du nombre de clusters. Le k optimal déterminé par cette méthode est sensible aux utilisateurs car il n'y a pas de « *bonne réponse* ». De plus le coude n'est pas toujours très visible. Cette méthode a tout de même l'avantage d'être simple à mettre en place et de permettre une première comparaison entre les différents nombres de classes.

La mesure de variation choisie est souvent la variance. Nous pouvons, par exemple, choisir de calculer le ratio entre la variance inter-classes et la variance totale, ou le ratio entre la variance intra-classes et la variance inter-classes.

4.3 Méthodes de ré-échantillonnage

4.3.1 Données déséquilibrées

Nous allons maintenant aborder la problématique des données déséquilibrées, qui est un problème commun lorsque nous faisons face à un problème de classification, où il n'est pas rare d'avoir des classes très déséquilibrées. Dans ce cas le modèle pourrait faire le choix d'associer tous les points à la classe majoritaire et ainsi aboutir à une erreur plutôt faible. Prenons l'exemple d'un jeu de données D_n de taille $n = 100$. On cherche à modéliser la variable binaire y. On observe que pour 90 % des observations on a $y = 0$. Dans cet exemple, un modèle prédisant uniquement des 0 aura donc une erreur de prédiction de 10 %. Un taux si faible est souvent considéré comme satisfaisant. Néanmoins, ce modèle n'a aucun intérêt dans la pratique car il ne permet pas d'anticiper quand la variable d'intérêt sera égale à 1.

Il existe différentes méthodes pour éviter cet écueil. Nous pouvons, par exemple, choisir une métrique de la qualité du modèle qui sera adaptée à ce problème. Il est aussi possible de ré-échantillonner le jeu de données de façon à faire apparaître plus d'éléments de la classe minoritaire dans la base d'entraînement et ainsi équilibrer les classes de la variable d'intérêt dans le jeu de données.

4.3.2 Ré-échantillonnage

Dans cette section, nous verrons les différentes méthodes de ré-échantillonnage qui seront utilisées dans ce mémoire.

Tout d'abord, il convient de rappeler que toute méthode de ré-échantillonnage est fondée soit sur le sous-échantillonage, soit sur le sur-échantillonnage, soit sur une combinaison de ces méthodes. Définissons donc ces notions :

- Le sous-échantillonnage consiste à supprimer des individus de la classe majoritaire. Il entraîne donc une perte d'informations dans la mesure où des individus pertinents, pour la modélisation sont supprimés du jeu de données.
- A contrario, le sur-échantillonnage vise à ajouter des individus de la classe minoritaire. Cela peut donc entraîner un biais important dans les données car certaines corrélations pourraient avoir un poids trop important par rapport à la réalité.

Dans la suite, nous allons détailler trois méthodes différentes de ré-échantillonnage.

4.3.2.1 Méthode aléatoire

La méthode aléatoire consiste à tirer aléatoirement des observations. On pourra ainsi, soit augmenter le nombre d'individus de la classe minoritaire en dupliquant certains individus de cette classe, soit supprimer des membres de la classe majoritaire tirés aléatoirement. On peut aussi combiner le sur- et le sous-échantillonnage avec pour objectif d'atteindre une proportion cible de la classe minoritaire dans la base de données finale.

L'avantage de cette méthode est d'être très simple à mettre en place. Néanmoins rien ne garantit la qualité de la base de données ainsi ré-échantillonnée.

4.3.2.2 ROSE

La méthode *ROSE*, pour *Random Oversampling Examples* publiée en 2010 par Menardi et Torelli[14] vise à créer des individus dans le voisinage des points du jeu de données.

Soit $n_j < n$ la taille de Y_j avec $j \in \{0, 1\}$, le nombre d'individus par classe de la variable y. On va chercher à établir des domaines $\mathcal{X}$ avec $f(x)$ une densité de probabilité sur $\mathcal{X}$. L'algorithme ROSE pour créer un nouvel élément consiste à réaliser les étapes suivantes :

1. sélection de $y = Y_j$ avec probabilité $\frac{1}{2}$;
2. sélection d'un individu (x_i, y_i) tel que $y_i = y$ avec probabilité $p_i = \frac{1}{n_j}$
3. création d'un individu (x'_i, y'_i) dans le voisinage de (x_i, y_i) à l'aide d'une fonction K_{H_j} qui est une distribution de probabilité centrée sur x_i et avec H_j une matrice permettant d'affecter des poids aux différentes variables explicatives.

Le nouvel individu est donc généré par estimation de la densité conditionnelle $f(x|y = y_i)$ dans le voisinage des points déjà existants. La taille du voisinage est définie par la matrice H_j.

4.3.2.3 SMOTE

La méthode SMOTE, pour *Synthetic Minority Oversampling Technique*, va créer des nouveaux individus à l'aide de la méthode des k plus proches voisins.

Les entrées de l'algorithme sont les suivantes :

- n le nombre d'individu de la classe minoritaire ;
- s la proportion de sur-échantillonnage souhaitée, *par exemple si l'on fixe s à* 300 %, $3n$ *individus seront créés* ;
- k le nombre de voisins, il est généralement lié à s.

On détermine d'abord les k plus proches voisins de chaque individu de la classe minoritaire. L'objectif sera alors de créer une population synthétique à l'aide de ces voisinages établis.

Soit i un individu du jeu de données, $X = (x_1, \ldots, x_m)$ le vecteur des variables explicatives et k_i le $k^{\text{ème}}$ plus proche voisin de i. L'algorithme va se dérouler ainsi, pour toute variable $x_j \in X$:

1. calcul de la différence entre la valeur de la variable x_j pour i et k_i :

$$D(x_j, i, k_i) = x_j(k_i) - x_j(i) \ ;$$

2. ajout d'un aléa α dans l'algorithme, tel que $\alpha \in [0, 1]$;

3. calcul de la valeur de x_j pour le nouvel individu à créer :

$$x_j(m_i^k) = x_j(i) + D(x_j, i, k_i) \times \alpha \ ;$$

où m_i^k est l'individu créé à partir du $k^{\text{ème}}$ voisin le plus proche de i ;

4. Le nouvel individu sera alors caractérisé par le vecteur $(x_1(m_i^k), \ldots, x_m(m_i^k))$.

On obtient alors $s \times n$ nouveaux individus à la suite de cet algorithme.

Chapitre 5

Modèles linéaires généralisés

Après avoir défini quelques éléments préliminaires à la modélisation, nous allons maintenant définir le modèle le plus classique de l'assurance : le modèle linéaire généralisé, ou GLM. Tout d'abord, nous allons définir le modèle linéaire gaussien. Puis, dans un second temps, nous verrons les principes des GLMs. Enfin, nous détaillerons un cas particulier des GlM : la régression logistique.

5.1 Modèles Linéaires Gaussiens

5.1.1 Définition

Commençons donc par une définition des modèles linéaires. L'objectif de ces modèles est d'expliquer une variable quantitative (c'est-à-dire continue/qui peut prendre une infinité de valeurs), à partir de variables explicatives qui, elles, pourront être quantitatives ou qualitatives.
Soit un modèle avec n observations et p variables explicatives, posons :

- $Y = (Y_1, \dots, Y_n)'$ le vecteur représentant les variables que l'on cherche à expliquer ;
- $X_i = (X_{i,1}, \dots, X_{i,p})$ le vecteur représentant les variables explicatives pour la variable Y_i ;
- X, la matrice de taille $n \times p$ dont les lignes sont les vecteurs X_i
- $\beta = (\beta_1, \dots, \beta_p)'$, le vecteur correspondant aux p paramètres du modèle ;
- ε le terme d'erreur, c'est-à-dire la différence entre l'observé et le prédit : $\varepsilon = Y - \mathbb{E}[Y|X]$.

Ainsi, un modèle est un modèle linéaire gaussien s'il satisfait l'équation :

$$Y = X\beta + \varepsilon$$

$$\text{(ou en forme matricielle dévelopée)} = \begin{pmatrix} y_1 \\ \vdots \\ y_n \end{pmatrix} = \begin{pmatrix} 1 & x_{1,1} & \cdots & x_{1,p} \\ \vdots & \vdots & \ddots & \vdots \\ 1 & x_{n,1} & \cdots & x_{n,p} \end{pmatrix} \begin{pmatrix} a_0 \\ a_1 \\ \vdots \\ a_p \end{pmatrix} + \begin{pmatrix} \varepsilon_1 \\ \vdots \\ \varepsilon_n \end{pmatrix} \tag{5.1}$$

et s'il vérifie les hypothèses suivantes :

($\mathcal{H}_1$) la matrice X est de plein rang, soit $rang(X) = p$;

($\mathcal{H}_2$) le bruit ε est gaussien, centré et ses éléments sont non corrélés, i.e. $\varepsilon \sim \mathcal{N}_p(0, \sigma^2 I)$.

5.1.2 Estimation des paramètres

Il existe plusieurs méthodes pour estimer les paramètres d'un modèle linéaire gaussien. Nous en détaillerons deux dans la suite : la méthode des moindres carrés et la méthode du maximum de vraisemblance.

5.1.2.1 Estimation par maximum de vraisemblance

Pour cette méthode on utilisera la fonction de vraisemblance du modèle :

$$\mathcal{L}(\beta, Y) = \prod_{i=1}^{n} f(y_i, \beta) \tag{5.2}$$

avec $f(y_i, \beta)$ la densité de la loi normale sur Y.

Pour obtenir l'estimateur du maximum de vraisemblance $\hat{\beta}$, on va maximiser sa log-vraisemblance en fonction de β. Pour cela, il faut résoudre le système d'équation suivant, pour lequel $\hat{\beta}$ est solution :

$$\frac{\partial}{\partial \beta_j} ln(\mathcal{L}(\beta_1, \ldots, \beta_k; Y)) = 0 \; pour \; j = 1, \ldots, p \tag{5.3}$$

De la même manière, on peut obtenir un estimateur de σ^2 en maximisant la log-vraisemblance selon σ^2.

5.1.2.2 Estimation par la méthode des moindres carrés

Cette deuxième méthode vise à minimiser la somme des carrés des écarts pondérés entre les observations et les projections du modèle.

$$\hat{\beta} = \underset{\beta \in \mathbf{R}^p}{\arg\min} \|Y - X\beta\|^2 \tag{5.4}$$

De plus, on sait que $\hat{\beta}$ est unique sans biais et se calcule grâce à la formule :

$$\hat{\beta} = (X'X)^{-1}X'Y \tag{5.5}$$

5.2 Modèles linéaires généralisés

Le GLM est une généralisation des modèles linéaires gaussiens vus précédemment. En effet, la variable Y que nous cherchons à prédire n'est plus forcément normale, mais la loi de $Y|X$ doit appartenir à une famille exponentielle. Aussi, plutôt que de chercher directement à modéliser la variable d'intérêt, nous allons modéliser une fonction de l'espérance de cette variable. Cette fonction s'appelle la fonction de lien. L'usage de ce modèle est très courant dans les compagnies d'assurances, notamment en assurance Non-Vie.

5.2.1 Familles exponentielles

5.2.1.1 Définition

Un modèle statistique $(\Omega, \mathcal{F}, (\mathbb{P}_{\theta,\psi})_{\theta \in \Theta, \psi > 0})$ est appelé famille exponentielle si les probabilités $\mathbb{P}_{\theta,\psi}$ admettent une densité f par rapport à une mesure dominante avec :

$$f_{\theta,\psi}(y) = c_\psi(y) exp\left(\frac{y\theta - a(\theta)}{\psi}\right). \tag{5.6}$$

- θ est le paramètre canonique et ψ le paramètre de dispersion, souvent considéré comme un paramètre de nuisance (un paramètre de nuisance est un paramètre qui n'est pas d'un intérêt immédiat mais qui doit être pris en compte dans l'analyse des paramètres d'intérêt) ;
- $a(\theta)$ est de classe C^2 et convexe ;
- $c\psi(y)$ ne dépend pas de θ.

Notons que, comme la définition d'une famille exponentielle fait intervenir une mesure dominante, les lois discrètes peuvent aussi être des familles exponentielles.

5.2.1.2 Quelques exemples

Nous pouvons détailler quelques distributions classiques qui appartiennent à une famille exponentielle en précisant les paramètres canoniques et de dispersion pour ces quelques exemples :

- Loi Normale de paramètres m et σ^2, avec σ^2 connus

$$\theta = m, \; a(\theta) = \theta^2/2, \; \psi = \sigma^2 \; ;$$

- Loi exponentielle ou gamma de paramètres k et λ, avec k connu

$$\theta = -1/\lambda, \; a(\theta) = -klog(-\theta), \; \psi = 1 \; ;$$

- Loi de Poisson de paramètre λ
$$\theta = \lambda,\ a(\theta) = e^{\theta},\ \psi = 1\ ;$$
- Loi de Bernoulli ou binomiale de paramètres n et p avec n connu
$$\theta = log(p/(1-p)),\ a(\theta) = nlog(1+e^{\theta}),\ \psi = 1\ ;$$
- Loi binomiale négative de paramètres p et r, avec r connu
$$\theta = log(1-p),\ a(\theta) = -rlog(1-e^{\theta}),\ \psi = 1.$$

5.2.2 Définition du GLM

Le GLM est caractérisé par trois composantes principales, une loi de probabilité de la famille exponentielle définie ci-dessus, un prédicteur linéaire et une fonction de lien.

5.2.2.1 Composantes du modèle

Loi de probabilité :
Cette composante, aussi appelée *composante aléatoire*, est caractérisée par la loi de probabilité de la variable d'intérêt Y. Elle a donc une fonction de densité de la forme de 5.6

Prédicteur linéaire :
Le *prédicteur linéaire*, est la *composante déterministe* du modèle et est défini par le vecteur de taille n suivant :

$$\eta = X\beta = \beta_0 + \sum_{i=1}^{p} X_i\beta_i \tag{5.7}$$

Comme dans le cas des modèles linéaires non généralisés, β est le vecteur des p paramètres à estimer et X est la matrice des X_i variables explicatives.

Fonction de lien :
La fonction de lien permet d'établir un lien entre le prédicteur linéaire et la moyenne de la fonction de distribution. Cette fonction, notée g, est bijective et on peut la définir mathématiquement de la façon suivante :

$$g(\mu(X)) = g(\mathbb{E}[Y|X]) = X\beta = \eta \tag{5.8}$$

en reprenant les notations utilisées pour la définition du modèle linéaire ci-dessus et en ajoutant $\mu(X) = \mathbb{E}[Y|X]$.

Un modèle est un modèle linéaire généralisé s'il vérifie les hypothèses suivantes :

1. $Y|X = x \sim \mathbb{P}_{\theta,\phi}$ appartient à une famille exponentielle ;
2. $g(\mu(X)) = g(\mathbb{E}[Y|X]) = X\beta$ pour une certaine fonction g bijective, appelée fonction de lien.

5.2.3 Estimation des paramètres

Les paramètres à estimer sont β et ψ, et le seront par maximum de vraisemblance.

$$\begin{cases} \eta_i = X_i'\beta \\ \mu_i = \mathbb{E}[Y_i|X_i] = g^{-1}(X_i'\beta) = g^{-1}(\eta_i) \\ \theta_i = (a')^{-1}(\mu_i) = (a')^{-1}(g^{-1}(X_i'\beta)) = (a')^{-1}(g^{-1}(\eta_i)) \end{cases} \tag{5.9}$$

On peut alors écrire la log-vraisemblance de la façon suivante :

$$l(Y_i;\beta,\psi) = \sum_{i=1}^{n} logf(Y_i;\beta,\psi) = \sum_{i=1}^{n} \underbrace{\left\{ logc_{\psi}(Y_i) + \frac{Y_i\theta_i - a(\theta_i)}{\theta} \right\}}_{:= l_i(\theta_i)} \tag{5.10}$$

où l'on suppose que les Y_i sont indépendants.
Considérons l'EMV de β_j :

$$\frac{\partial l}{\partial \beta_j} = \sum_{i=1}^{n} \frac{\partial l_i(\theta_i)}{\partial \beta_j} = \sum_{i=1}^{n} \frac{\partial l_i}{\partial \theta_i}\frac{\partial \theta_i}{\partial \beta_j} \tag{5.11}$$

D'une part,

$$\frac{\partial l_i}{\partial \theta_i} = \frac{Y_i - a'(\theta_i)}{\psi} = \frac{Y_i - \mu_i}{\psi} \tag{5.12}$$

D'autre part, comme $\eta_i = X_i'\beta$, on peut écrire,

$$\frac{\partial \theta_i}{\partial \beta_j} = \frac{\partial \theta_i}{\partial \eta_i}\frac{\partial \eta_i}{\partial \beta_j} = \frac{\partial \theta_i}{\partial \eta_i} x_{i,j} \tag{5.13}$$

où $x_{i,j}$ est la $j^{ème}$ coordonnée de X_i'.
Enfin,

$$\frac{\partial \theta_i}{\partial \eta_i} = \left(\frac{\partial \eta_i}{\partial \theta_i}\right)^{-1} \tag{5.14}$$

et

$$\frac{\partial \eta_i}{\partial \theta_i} = \frac{\partial \eta_i}{\partial \mu_i}\frac{\mu_i}{\theta_i} = g'(\mu_i)a''(\theta_i). \tag{5.15}$$

Ainsi,

$$\frac{\partial l}{\partial \beta_j} = \frac{1}{\psi}\sum_{i=1}^{n} \frac{x_{i,j}(Y_i - \mu_i)}{g'(\mu_i)a''(\theta_i)} \tag{5.16}$$

Soit D la matrice diagonale dont les coefficients sont égaux à $\frac{1}{g'(\mu_i)a''(\theta_i)}$ alors

$$\frac{\partial l}{\partial \beta_j} = 0 \text{ pour tout } j = 1, \ldots, p \Leftrightarrow X'D(Y - \mu) = 0 \tag{5.17}$$

et donc l'EMV $\hat{\beta}$ est solution de

$$X'D(Y - g^{-1}(X\beta)) = 0 \tag{5.18}$$

Dans le cas général, on ne pourra pas calculer explicitement cette équation. On calculera donc numériquement l'estimateur du maximum de vraisemblance, à l'aide de méthodes itératives, telles que les algorithmes de Newton-Raphson ou de Fisher. On définira l'information de Fisher comme la matrice de terme général :

$$\mathbb{E}\left(\frac{\partial^2 \mathcal{L}(\beta)}{\partial \beta_j \partial \beta_k}\right) = -\sum_{i=1}^{n} \frac{x_{ij}x_{ik}}{Var(Y_i|X = x)}\left(\frac{\partial \mu_i}{\partial \eta_i}\right)^2 \tag{5.19}$$

5.2.4 Mesure de la déviance et qualité d'ajustement

Une fois les paramètres estimés, nous allons nous intéresser à l'évaluation de la qualité du modèle. Pour cela, nous étudierons les différences entre observations et estimations pour voir si notre modèle s'ajuste bien, et décrit bien les valeurs observées.

Nous avons, dans la sous-section précédente, estimé le vecteur des paramètres β par $\hat{\beta}$. Nous pouvons maintenant exprimer la log-vraisemblance du modèle en fonction de $\hat{\beta}$, que nous noterons $\mathcal{L}(\hat{\beta})$. Si le modèle était parfait, alors on aurait la prévision du modèle $\hat{\mu}_i$ tel que, $\hat{\mu}_i = \mathbb{E}(Y_i|X = x)$. Cela signifie que ce modèle possède autant de paramètres que d'observations et estime donc parfaitement les données. On dit alors qu'un tel modèle est saturé, et sa log-vraisemblance est notée $\mathcal{L}_{sat}$. Le modèle estimé est alors comparé avec le modèle saturé à l'aide de la *déviance* des log-vraisemblances $\mathcal{L}(\hat{\beta})$ et $\mathcal{L}_{sat}$, qui est défini mathématiquement, comme suit :

$$D(\hat{\beta}) = -2(\mathcal{L}(\hat{\beta} - \mathcal{L}_{sat}) \tag{5.20}$$

La déviance est donc l'écart de log-vraisemblance entre le modèle saturé d'ajustement maximum et le modèle estimé. Plus la déviance est *faible*, meilleur est le modèle. A contrario, si la déviance est *grande*, alors le modèle sera considéré comme loin du modèle saturé et donc de moindre qualité. Néanmoins, pour définir si la déviance est *petite* ou *grande*, on peut utiliser le test de rapport de vraisemblance. Ce test consiste à étudier le rapport ou

la différence de déviance entre différents modèles où nous aurions ajouté (ou enlevé) des variables explicatives dans l'ajustement du modèle. Pour savoir quel modèle est le meilleur, nous calculons la différence des déviances :

$$D^* = D_2(\tilde{\beta}) - D_1(\hat{\beta}) = -2(\mathcal{L}_2(\tilde{\beta}) - \mathcal{L}_{sat}) + 2(\mathcal{L}_1(\hat{\beta}) - \mathcal{L}_{sat}) = 2(\mathcal{L}_1(\hat{\beta}) - \mathcal{L}_2(\tilde{\beta})) \tag{5.21}$$

avec $\hat{\beta}$ et $\tilde{\beta}$ des vecteurs, de taille p et n, respectivement. Ce critère est relatif et ne permet de comparer que deux modèles emboîtés.

En particulier, l'hypothèse nulle qui établit que le modèle considéré à p paramètres est adéquat peut se reformuler en disant que la diminution de déviance est significative, c'est-à-dire que les p paramètres sont non-nuls, et que les paramètres supplémentaires pour arriver au modèle saturé sont nuls. Sous cette hypothèse, la déviance du modèle suit approximativement une loi du χ^2 à $(n-p)$ degrés de liberté pour les lois à un paramètre, comme les lois binomiales ou de Poisson, et une loi de Fisher pour les lois à deux paramètres, comme les lois gaussiennes. Dans le cas des lois à un paramètre, nous rejetons l'hypothèse nulle H_0 si la différence D^* dépasse un seuil critique α du χ^2 à $(n-p)$ degrés de liberté que nous avons choisi.

D'autres critères existent pour comparer des modèles, citons notamment les critères d'AIC et de BIC qui se basent sur la vraisemblance. Ils permettent notamment de comparer des modèles qui ne seraient pas emboîtés.

- L'**AIC (Akaike Information Criterion)** se définit mathématiquement par $AIC = -2\mathcal{L}(\beta) + 2p$
- Le **BIC (Bayesian Information Criterion**) se définit mathématiquement par $BIC = 2\mathcal{L}(\beta) + plog(n)$

Dans la mesure où le BIC pénalise plus les modèles à grande dimension, il aura tendance à choisir des modèles plus *petits* que le critère AIC. Dans les deux cas, l'objectif est de minimiser la valeur du critère d'information.

Pour réduire la dimension des modèles à l'aide de ces critères, il convient d'utiliser une méthode « *stepwise* » parmi les suivantes :

- Méthode **ascendante** ou **forward** : Nous considérons un modèle ne comportant qu'une constante puis nous ajoutons pas à pas la variable, parmi l'ensemble des variables disponibles, qui permet la meilleure amélioration (selon le critère). Pour arrêter l'algorithme, nous pouvons inclure un seuil à partir duquel on considerera que les variables ajoutées ne sont plus significatives.
- Méthode **descendante** ou **backward** : Pour cette méthode, nous partons d'un modèle complet et nous enlevons, à chaque pas la variable qui entraîne la moins grande perte. Dans ce cas aussi, nous pouvons inclure un critère d'arrêt pour arrêter l'algorithme avant qu'il ne supprime des variables trop importantes.
- Méthode **bidirectionnelle** ou **bidirectional** : Cette méthode est une combinaison des deux méthodes précédentes, qui va tester à chaque pas d'ajouter et de supprimer des variables.

5.3 Régression logistique

La régression logistique est un cas particulier des modèles linéaires généralisés évoqués ci-dessus, l'objectif, ici, est d'expliquer une variable binaire. On retrouve très couramment ces modèles dans le monde de l'assurance. En effet, dans la mesure où, ils permettent de modéliser des variables binaires, on peut s'en servir pour modéliser de nombreux éléments comme la probabilité de fraudes, détecter les sinistres graves, le taux de transformation des contrats, ou le cas qui nous intéresse ici, le taux de résiliation des contrats.

Comme la régression logistique est un cas particulier des modèles linéaires généralisés, elle reprend en grande partie les caractéristiques des composantes du modèle. En effet, les variables explicatives peuvent être quantitatives ou qualitatives, l'estimation des coefficients β se fait grâce à la méthode de la maximisation de vraisemblance. De même, on peut utiliser en plus de la déviance, les critères AIC et BIC pour juger de la qualité des modèles, et faire un choix entre plusieurs modèles.

5.3.1 Définition de la régression logistique

Dans le cadre de la régression logistique, la variable à expliquer Y peut prendre deux modalités, 0 ou 1, et sa loi est donc une loi de Bernoulli. En reprenant les notations précédentes, avec Y la variable à expliquer et X la matrice des variables explicatives. On cherche donc à exprimer l'espérance conditionnelle de Y, $\mathbb{E}(Y|X=x)$.

Cela donne dans le cas d'une loi de Bernoulli,

$$\mathbb{E}(Y|X=x) = 1 \times \mathbb{P}(Y=1|X=x) + 0 \times \mathbb{P}(Y=0|X=x) = \mathbb{P}(Y=1|X=x) \tag{5.22}$$

La fonction de lien canonique est la fonction Logit, elle est de la forme suivante :

$$ln\left(\frac{p}{1-p}\right) \tag{5.23}$$

Ainsi on a, en notant $\pi(x) = \mathbb{P}(Y = 1|X = x)$

$$g(p) = ln\left(\frac{\pi(x)}{1-\pi(x)}\right) = X\beta = \beta_0 + \sum_{i=1}^{p} X_i\beta_i \tag{5.24}$$

L'offset est nul dans le cas de la régression logistique, on peut donc réécrire $\pi(x)$ comme suit :

$$\pi(x) = \frac{e^{\beta_0+\sum_{i=1}^{p} X_i\beta_i}}{1+e^{\beta_0+\sum_{i=1}^{p} X_i\beta_i}} \tag{5.25}$$

5.3.2 Estimation des paramètres

Comme évoqué précédemment, dans les modèles linéaires généralisés, l'évaluation des coefficients β se fait, très souvent, par maximum de vraisemblance.

On cherche à maximiser la fonction suivante :

$$\mathcal{L}(\beta) = \prod_{i=1}^{n} f(y^i|X = x^i) \tag{5.26}$$

que l'on peut aussi écrire,

$$\mathcal{L}(\beta) = \prod_{i=1}^{n} \mathbb{P}(y^i|X = x^i) \tag{5.27}$$

où l'on considère n observations *indépendantes*, et x_i est le vecteur des variables explicatives de la $i^{ème}$ observation et y^i vaut 0 ou 1 dans le cas de la régression logistique binaire qui nous intéresse ici. On remarque que, si, $y^i = 1$, alors $\mathbb{P}(y^i|X = x^i) = \pi(x^i)$. On peut donc réécrire la fonction de vraisemblance en fonction de $\pi(x^i)$:

$$\mathcal{L}(\beta) = \prod_{i=1}^{n} \pi(x^i)^{y^i}(1-\pi(x^i))^{1-y^i} \tag{5.28}$$

En reprenant le résultat de l'équation 5.25, où nous avions exprimé $\pi(x^i)$ en fonction des coefficients β, nous pouvons remplacer $\pi(x^i)$ dans l'équation 5.28 ci-dessus.

On obtient donc,

$$\mathcal{L}(\beta) = \prod_{i=1}^{n} \left(\frac{e^{\beta_0+\sum_{k=1}^{p} X_k^i\beta_k}}{1+e^{\beta_0+\sum_{k=1}^{p} X_k^i\beta_k}}\right)^{y^i} \left(\frac{e^{\beta_0+\sum_{k=1}^{p} X_k^i\beta_k}}{1+e^{\beta_0+\sum_{k=1}^{p} X_k^i\beta_k}}\right)^{(1-y^i)} \tag{5.29}$$

L'estimation du vecteur β se fait par des méthodes numériques itératives en maximisant la fonction ci-dessus, à l'aide d'un logiciel comme R ou SAS, ou encore le langage de programmation Python, par exemple. Pour comparer différents modèles nous pourrons utiliser, comme pour les modèles linéaires généralisés, les critères AIC et BIC.

5.3.3 Rapport des chances

Une des particularités de la régression logistique est la possibilité de calculer des *Rapport des chances*, plus souvent appelés *Odds-Ratio*. Ce rapport permet de calculer l'importance d'un facteur par rapport à un autre, dans notre cas, il peut, par exemple, permettre de comparer deux profils d'assurés et de déterminer lequel aura le plus de chances (ou risques dans notre cas) de résilier son contrat prochainement.

5.3.3.1 Définition mathématique de la cote

On rappelle que nous sommes dans le cas d'une classification binaire, et que l'on a défini $\pi(x)$ comme la probabilité d'avoir $Y = 1|X = x$. On a donc $1 - \pi(x) = Y = 0|X = x$, par complémentarité des évènements $Y = 1|X = x$ et $Y = 0|X = x$. La *cote*, ou *odds* en anglais, permet de mesurer la possibilité de la réalisation d'un évènement par rapport à la non-réalisation de cet évènement.

$$cote = \frac{\pi(x)}{1 - \pi(x)} \tag{5.30}$$

Soit un assuré qui aurait 0.75 de chance de résilier son contrat, et donc 0.25 de chance de ne pas le faire, la cote associée serait de $0.75/0.25 = 3$, il aurait donc 3 fois plus de chance de résilier son contrat que de rester en portefeuille.

5.3.3.2 Définition mathématique du rapport des chances

Le *rapport des chances* permet d'observer l'évolution du rapport des probabilités d'obtenir $Y = 1$ contre celle d'avoir $Y = 0$ lorsque l'on passe d'un profil à un autre.

On peut, par exemple, s'intéresser à la variation d'une modalité de la variable X_i et calculer le *rapport des chances* associé à cette variation. Soit $x_{i,1}$ et $x_{i,2}$ les profils étudiés, pour lesquels seule la variable X_i diffère (le premier profil prend la modalité 1 et le deuxième prend la modalité 2), le *rapport des chances* se calcule ainsi :

$$\begin{aligned} OR &= \frac{\pi(x_{i,1})/(1 - \pi(x_{i,1}))}{\pi(x_{i,2})/(1 - \pi(x_{i,2}))} \\ &= \frac{\pi(x_{i,1})(1 - \pi(x_{i,2})}{\pi(x_{i,2})(1 - \pi(x_{i,1}))} \end{aligned} \tag{5.31}$$

Ce ratio permet d'évaluer le fait qu'un assuré correspondant à un certain profil $x_{i,1}$, et donc à la modalité 1 de la variable X_i dans cet exemple aura α plus de risque de résilier qu'un assuré correspondant au profil $x_{i,2}$. Une variable à k modalités se verra associer $k - 1$ ratios.

5.4 Conclusion

Nous avons vu dans cette partie, tout les éléments nécessaires à la création d'un modèle de régression logistique. Nous avons commencé par définir et expliquer l'estimation des paramètres des modèles linéaires gaussiens classiques, qui sont un préalable aux GLM. Ensuite, nous avons pu définir les modèles linéaires généralisés, en rappelant ce qu'est une famille exponentielle, par exemple, et en finissant par les outils de mesure de la qualité de l'ajustement. Enfin, nous avons défini la régression logistique et mis en exergue le principe des *rapport des chances* (ou *odds-ratio*).

Dans la suite, nous allons nous intéresser aux méthodes d'*apprentissage automatique*, ou *machine learning* en anglais.

Chapitre 6

Méthodes d'apprentissage automatique

L'apprentissage automatique regroupe l'ensemble des algorithmes, basé sur des approches mathématiques et statistiques, qui peuvent « *apprendre* » à partir de jeux de données. Il se divise en 2 parties, l'apprentissage supervisé et l'apprentissage non-supervisé. Dans le premier cas, on aura connaissance des *étiquettes*, ou *label* (en anglais), associées aux données existantes, et il pourra donc utiliser cet apprentissage pour *prédire* l'étiquette associée à de nouvelles données, dans le second cas, on cherchera plutôt à trouver des structures sous-jacentes dans un jeu de données, l'algorithme créera des groupes homogènes permettant de mieux saisir la base donnée notamment.

Nos modèles seront donc des méthodes d'apprentissage supervisé dans la mesure où la variable cible est connue.

6.1 Arbre de décision

Un arbre de décision est un algorithme d'apprentissage automatique qui permet d'expliquer une variable réponse Y, à l'aide de k variables explicatives $X = (X_1, \ldots, X_k)$. Cet algorithme permet de facilement obtenir une représentation graphique de la structure de nos données, il met aussi en avant les variables les plus discriminantes dans l'explication de la variable que l'on souhaite expliquer. Enfin, cet algorithme est non-paramétrique, il est donc relativement simple à mettre en place.

On peut expliciter le fonctionnement d'un arbre à l'aide d'un exemple graphique avant de voir plus en détail le fonctionnement de l'algorithme.

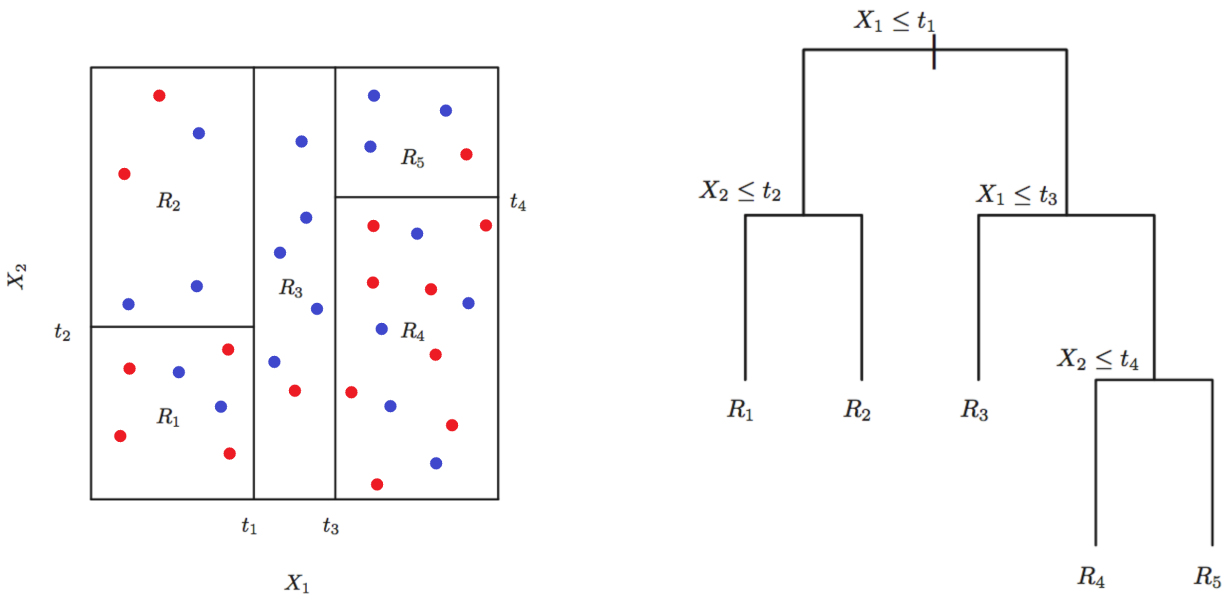


FIGURE 6.1 – Exemple d'arbre de classification (source : Cours de Machine Learning de Mme Boyer[4])

Dans cet exemple, on a un modèle à 2 variables explicatives X_1 et X_2, on cherchera à définir des sous-ensembles partitionnant au mieux les points selon leurs couleurs. On commence avec un repère qui n'est pas

segmenter puis on va considérer le test $X_1 \leqslant t_1$ pour diviser les données en 2 parties l'objectif étant de séparer au mieux les points bleus des points rouges. Après la première division, on obtient donc du côté gauche 6 points rouges et 5 bleus et 10 points rouges et 13 points bleus du côté droit. Nous reproduisons le processus de division jusqu'à obtenir 5 régions. D'après cette division du repère, nous pouvons établir un modèle prédictif, ainsi, nous pouvons adjoindre une couleur à un nouveau point selon la région dans laquelle il sera. Par exemple, un point a de coordonnées : $a(x_1 < t_1; x_2 < t_2)$ sera dans la région R_1 et le modèle prédira donc que c'est un point rouge, car il y a 4 points rouges et 2 points bleus en R_1, la classe majoritaire est donc la classe rouge. De même, un point $b(x_1 > t_3, x_2 > t_4)$ sera dans la région R_5 sera prédit comme étant un point bleu.

Le modèle d'arbre de décision le plus connu est l'algorithme CART (Classification And Regression Tree) publié par Leo Breiman en 1984. On distinguera donc deux types d'arbres en fonction de la nature d' Y :

- Les arbres de classification si l'on cherche à étudier une variable Y qualitative,
- Les arbres de regression si l'on cherche à étudier une variable Y quantitative, et donc continue.

Étant donné la nature de notre problème, nous nous intéresserons particulièrement aux arbres de classification.

6.1.1 Principe de fonctionnement

Un arbre de décision est un ensemble de règles définies successivement avec pour objectif de classer les individus de la base de données en groupes homogènes.

Avant la première itération, tous les individus font partis d'un même groupe et constituent le nœud initial, aussi appelé *racine*. Tous les nœuds, y compris la racine, correspondent à un test sur une variable explicative. Ensuite, les branches partitionnent le jeu de données en fonction du test correspondant au nœud, elles permettent ainsi le passage d'un *nœud père* à un *nœud fils*. Ce *nœud fils* peut être soit un nouveau test soit à une *feuille* de l'arbre, le nom donné aux nœuds terminaux. Les *feuilles* obtenues à l'aide de cet algorithme sont majoritairement composées d'individus d'une seule classe. L'ensemble des règles obtenues à chaque nœud forment alors le modèle prédictif.

6.1.2 Construction de l'arbre

À chaque itération, l'algorithme testera l'ensemble des variables explicatives et en sélectionnera comme étant la plus pertinente pour diviser l'échantillon du *nœud père* en 2 nœuds fils de la meilleure façon possible. L'algorithme s'arrête une fois que les éléments des nœud fils ont tous la même valeur de la variable cible ou lorsqu'une séparation supplémentaire n'améliore plus la qualité de prédiction. Il est cependant possible de paramétrer des conditions d'arrêt que nous verrons par la suite. Cela permet par exemple de réduire le temps de calcul car obtenir une division parfaite n'est pas utile et peut être particulièrement coûteux en temps de calcul, notamment si la base de données est grande. On obtient donc à chaque itération la variable qui explique le mieux la variable réponse.

6.1.2.1 Impureté d'un nœud

Pour définir la meilleure partition à faire à chaque itération, l'algorithme calcule l'impureté d'un nœud dont le calcul sera détaillé dans la suite.

Introduisons d'abord les notations dont on aura besoin.
Soit :

- On veut diviser un nœud N en deux fils N_G et N_D ;
- Cette séparation dépend du test réalisé que l'on caractérise par un couple (variable ; seuil) noté (v, s) ;
- $N_G(v, s) = \{x \in N : x_v < s\}$ et $N_D(v, s) = \{x \in N : x_v \geqslant s\}$ le nombre d'élément respectivement dans le nœud N_G et N_D.

Trouver la meilleure division de N revient donc à trouver le meilleur couple (v, s) en comparant l'impureté de N avec celles de $N_G(v, s)$ et de $N_D(v, s)$ et utiliser une fonction $IG(v, s)$ représentant le gain d'information pour chaque couple (v, s).

Dans le cas d'une classification avec K labels notés $y_i \forall i \in 1, \dots K$. On calcule d'abord la distribution par classe :

$$p_N = (p_{N,1}, \dots, p_{N,K}) \text{ où } p_{N,k} = \frac{\#\{i : x_i \in N, y_i = k\}}{|N|} \tag{6.1}$$

Il faut ensuite définir une mesure d'impureté I. L'indice le plus courant pour mesurer l'impureté d'un nœud est l'indice de Gini qui se définit comme suit :

$$
\begin{aligned}
G(N) = G(p_N) &= \sum_{k=1}^{K} p_{N,k}(1 - p_{N,k}) \\
&= \sum_{k=1}^{K} (p_{N,k} - p_{N,k}^2) \\
&= \sum_{k=1}^{K} p_{N,k} - \sum_{k=1}^{K} p_{N,k}^2 \\
\text{or } \sum_{k=1}^{K} p_{N,k} = 1 \text{ donc } &= 1 - \sum_{k=1}^{K} p_{N,k}^2
\end{aligned}
\tag{6.2}
$$

Ici, nous traitons d'une classification binaire, on peut donc exprimer l'indice de Gini dans le cas où $K = 2$.

$$
\begin{aligned}
G(N) &= \sum_{k=1}^{2} p_{N,k}(1 - p_{N,k}) \\
&= 1 - \sum_{k=1}^{2} p_{N,k}^2
\end{aligned}
\tag{6.3}
$$

Le gain d'information entre un nœud N et ses enfants N_G et N_D se définit comme la différence entre l'impureté du nœud N et les impuretés des nœud N_G et N_D, pondéré par leurs poids respectifs.

$$
IG(v, s) = G(N) - \frac{|N_G(v, s)|}{|N|} G(N_G(v, s)) - \frac{|N_D(v, s)|}{|N|} G(N_D(v, s)) \tag{6.4}
$$

L'objectif est de maximiser le gain d'information à chaque itération de l'algorithme, et donc de minimiser l'indice de Gini. En effet, plus l'indice de Gini est faible plus le nœud est pur.

6.1.2.2 Feuilles et critères d'arrêt

Si un noœud, de l'arbre de décision, est homogène alors il sera un nœud terminal, autrement appelé *feuille*. L'homogénéité d'un nœud peut être vérifiée dans plusieurs cas :

- Si la majorité des individus du nœud sont bien classés et qu'une division supplémentaire n'améliorerait pas la segmentation du nœud ;
- Ou si toutes les variables ont été testées mais qu'aucune n'engendre d'amélioration de la qualité du modèle

Il est aussi possible de définir des critères d'arrêt pour stopper prématurément l'algorithme avec pour but d'améliorer la robustesse du modèle. Parmi les critères d'arrêt les plus communs, on peut notamment citer, la profondeur maximale de l'arbre, le nombre d'individu par nœud, ou encore si l'on considère que l'impureté des feuilles est *assez* petite.

Après création de l'arbre, les individus seront affectés à la feuille à laquelle ils appartiennent et les feuilles se voient donc affecter une classe en conséquence.

Pour déterminer la classe k d'une feuille F, on s'intéresse à la proportion d'individus $p_{F,k}$ précédemment définie comme le nombre d'éléments de F qui appartiennent à la classe k. Le taux d'erreur de F est défini comme la quantité $1 - p_{F,k}$, traditionnellement, on cherche à minimiser ce taux. Par conséquent, on assignera, souvent, la classe majoritaire aux feuilles d'un arbre.

Une fois l'arbre construit, il existe néanmoins d'autres méthodes permettant elles aussi d'améliorer le modèle. Nous parlerons dans la partie suivante de l'élagage.

6.1.3 Élagage

On a vu dans la partie précédente les critères d'arrêt, parfois appelé pré-élagage. L'objectif est d'améliorer la qualité finale d'un arbre de décision au cours de sa construction en anticipant un phénomène appelé *sur-apprentissage* que l'on définira dans une prochaine section. L'élagage comme il est communément entendu, désigne le processus visant à couper des branches d'un arbre après l'avoir construit entièrement dans un premier temps. Dans le cas des critères d'arrêt, on parlera de méthodes « *Top-Down* » et dans le cas de l'élagage "classique", on parlera de méthodes « *Bottom-Up* » en effet, dans le premier cas, on part de la racine et on "descend" jusqu'aux feuilles et dans le second cas, on part des feuilles et on "remonte" jusqu'à la racine.

Le principe de l'élagage est donc de partir des extrémités de l'arbre puis de le réduire, souvent en utilisant un jeu de données de validation qui n'a pas servi à la construction de l'arbre. Si le taux d'erreur de l'arbre à un certain nœud est inférieur à celui obtenu lorsque les fils de ce nœud sont considérés, alors on élaguera au niveau du nœud parent.

6.1.4 Conclusion sur les arbres de décision

Dans cette partie, nous avons pu définir les arbres de décisions et aussi expliciter les principes de ce modèle par le biais de l'algorithme CART qui est un algorithme classique de construction d'arbre de régression et de classification. Nous avons aussi vu différentes méthodes visant à améliorer le résultat final du modèle que ce soit pendant la construction de l'arbre ou une fois qu'il est construit en utilisant une base de validation.

Les arbres de décisions ont de nombreux avantages, on peut insérer des variables qualitatives et quantitatives, ils sont assez simples à comprendre car ils permettent une représentation graphique relativement intuitive etc. Cependant, il a de nombreuses limites, on observe par exemple dans la littérature actuarielle qu'il est souvent décrié comme un modèle trop simple et ne possède pas un grand pouvoir prédictif, aussi il dépend fortement de la base d'apprentissage ce qui rend particulièrement important l'élagage, enfin les arbres de décisions permettent uniquement de trouver un optimal local sans certitude que celui-ci soit global. Pour ces raisons, de nombreuses techniques ont été développées en prenant pour base les arbres de décision, elles offrent néanmoins de meilleurs résultats prédictifs et généralement une meilleure robustesse vis-à-vis de la base d'apprentissage. On verra dans un premier temps la méthode du « *Bagging* » qui constite à aggréger des arbres en grands nombres pour en tirer un arbre "optimal", et dans un second temps la méthode du « *Boosting* » qui est aussi une méthode visant à créer un grand nombre d'arbres différents mais en prenant en compte les résultats des précédents arbres pour améliorer le résultat final.

6.2 Bagging

Le *Bagging* (pour **B**ootstrap **Agg**regat**ing**) est une technique d'apprentissage automatique proposée par Leo Breiman[5]. Plutôt que de créer un modèle très sophistiqué, et donc compliqué à mettre en oeuvre l'idée de la méthode *bagging* est d'agréger un grand nombre de modèles "simple". L'objectif principal étant de réduire la variance [1] des modèles et d'éviter le surapprentissage. Cette méthode est particulièrement utile dans le cadre des arbres de décisions, car elle permet notamment d'en améliorer la précision de ces derniers.

6.2.1 Principe du *bagging*

La première étape d'un algorithme de bagging est de constituer une base de données par la méthode statistique du bootstrap. Soit un jeu de données composé de n individus, on va effectuer un tirage aléatoire avec remise de n' individus. On va ensuite ajuster le modèle à partir de ce "nouveau" jeu de données. Ces opérations sont répétées un nombre B de fois. Ensuite, nous pouvons agréger les B modèles (en utilisant le vote majoritaire dans le cas d'un classifieur). Le nombre B de modèle est à la discrétion de chacun mais il est habituellement fixé autour de 500, de plus, plus B est grand plus le modèle se stabilise.

Algorithme Bagging

for b = 1,..., B **do** :
Tirage avec remise de $D_{n'}^{(b)}$ dans le jeu originel D_n ;
Ajustement du classifieur (resp. regresseur) $\hat{f}^{(b)}$ sur $D_{n'}^{(b)}$.
end for
Agrégation des $\hat{f}^{(1)}$, ..., $\hat{f}^{(B)}$ pour obtenir un seul classifieur (resp. regresseur) par la méthode du vote majoritaire (resp. en utilisant la moyenne).

Cet algorithme permet donc en effet de gagner en stabilité vis-à-vis des données et d'obtenir un modèle plus robuste, en revanche il présente l'inconvénient de perdre en lisibilité.

6.2.2 Erreur *Out-Of-Bag*

L'erreur « *Out-of-Bag* » *(OOB)* est une méthode de mesure de l'erreur de prédiction d'un modèle faisant intervenir le *bagging*. Son calcul se réalise à partir des observations qui n'ont pas été sélectionnées lors du bagging, et en pratique on calcule la moyenne de l'erreur de prédiction sur les données exclues pour chaque modèle créé lors de l'algorithme de bagging.

1. La variance d'un arbre de décision se définit par l'instabilité liée au choix du jeu de données[22]

Soit une observation (x_i, y_i) de la base donnée, on établit la liste de tout les $\hat{f}^{(b)}$, $\forall b \in [\![1, B]\!]$ dans laquelle l'observation (x_i, y_i) n'a pas été élue pour la création du modèle. On dit alors que l'observation (x_i, y_i) est « *Out-of-Bag* ». Ensuite, on calculera, pour tous les modèles **ne** contenant **pas** l'observation (x_i, y_i), la prédiction du modèle pour x_i que l'on notera $\hat{y}_i$. Enfin, le calcul de l'erreur OOB diffère selon le type de modèle :

$$E_{OOB} = \frac{1}{n}\sum_{i=1}^{n}(\hat{y}_i - y_i)^2 \textit{ pour un modèle de régression,} \tag{6.5}$$

$$E_{OOB} = \frac{1}{n}\sum_{i=1}^{n}\mathbb{1}_{\hat{y}_i \neq y_i} \textit{ pour un modèle de classification.} \tag{6.6}$$

avec n le nombre d'individus dans la base de données.

L'erreur « *Out-of-Bag* » peut être utile dans le choix de B le nombre de boostrap réalisé, en effet il se peut que l'erreur OOB augmente pour un B trop grand.

6.2.3 Forêts aléatoires

6.2.3.1 Définition et principe de l'algorithme

Les *forêts aléatoires*, ou *Random Forest* en anglais, sont des méthodes d'apprentissage automatique qui reprennent le principe du *bagging* évoqué dans la partie précédente. En effet, le principe de cette méthode, introduit par Breiman en 2001[6] est de créer des modèles à faible pouvoir prédictif, ou « *weak-learner* » en anglais, ici les arbres de décision, puis d'en agréger un grand nombre. Cependant, cet algorithme va aussi effectuer un tirage aléatoire parmi les variables explicatives à la création des nœuds. Cet aléa supplémentaire a pour but de diminuer la variance en rendant plus indépendant les arbres entre eux, mais cela peut aussi mettre en exergue des variables qui semblent peu importantes lorsque l'on considère l'ensemble de celles-ci mais qui apparaissent dans des échantillons plus petit de variables.

D'après Breiman[7] en 2002, il faut prendre un échantillon de $\sqrt{k}$ variables pour un problème de classification et un échantillon de $\frac{k}{3}$ pour un problème de régression. En pratique, il est recommandé de tester plusieurs valeurs pour observer laquelle est la plus pertinente.

Algorithme Random Forest

for b = 1,..., B **do** :
Tirage avec remise de $D_{n'}^{(b)}$ dans le jeu originel D_n ;
Ajuster un arbre $\hat{f}^{(b)}$ sur $D_{n'}^{(b)}$ en ajoutant un aléa dans la sélection des variables, à chaque nœud, on sélectionne k variables avant la recherche de la division optimale.
end for
Agrégation des $\hat{f}^{(1)}$, ..., $\hat{f}^{(B)}$ pour obtenir un seul classifieur (resp. régresseur) par la méthode du vote majoritaire (resp. en utilisant la moyenne).

6.2.3.2 Quantification de l'importance des variables

L'un des inconvénients de l'agrégation d'arbre, est la perte de l'interprétabilité immédiates des arbres de décision, néanmoins deux outils permettent de déterminer, par le calcul, l'importance des variables, vis-à-vis de la régression ou de la classification.

Admettons que l'on souhaite calculer l'importance d'une variable x_i. Nous détaillerons, ici, deux méthodes différentes :

- Le premier outil est la « *Mean Decrease Accuracy* » qui repose sur une permutation des valeurs de x_i dans l'arbre. Le calcul de la *Mean Decrease Accuracy* fait intervenir l'erreur OOB, en effet, lors de la création d'un arbre, nous allons calculer dans un premier temps la valeur de l'erreur OOB sans permutation, notée $OOB_{sp}^{(i)}$, puis après avoir effectué une permutation des valeurs de x_i dans l'arbre, nous allons recalculer l'erreur OOB, que nous noterons $OOB_{p}^{(i)}$. On calculera ensuite la différence entre $OOB_{sp}^{(i)}$ et $OOB_{p}^{(i)}$, plus la différence est importante plus cela signifie que la variable est importante, dans cet arbre. Pour calculer l'importance globale de x_i, il faut faire la moyenne de $OOB_{sp}^{(i)} - OOB_{p}^{(i)}$ sur l'ensemble des B arbres.
- Le deuxième outil est le « *Mean Decrease Gini* ». Il est basé sur la mesure d'impureté d'un nœud, en effet, il se définit comme étant la somme pondérée des gains d'homogénéité induits lorsque la variable x_i est utilisée dans une division d'un nœud.

6.2.3.3 Avantages et inconvénients des forêts aléatoires

Pour conclure à propos des *forêts aléatoires*, nous pouvons établir quelques avantages et inconvénients de ces modèles. Premièrement, l'agrégation de nombreux *weak-learners* permet souvent une meilleure prédiction. En effet, comme nous l'avons évoqué, le bagging et l'aléa dans la sélection de variable ajouté par l'algorithme des *forêts aléatoires* permettent de réduire la variance des modèles, ce qui est important dans la mesure où un modèle à forte variance aura de grands risques d'être sur-ajusté. On peut aussi noter que le paramétrage de ces modèles est relativement simple. Toutefois, ces avantages et notamment le gain en précision se font en perdant grandement l'interprétabilité [2] du modèle. En effet, les arbres de décisions ont pour principal avantage leur grande interprétabilité, notamment car une représentation graphique est simple à établir. Cette possibilité disparaît avec l'agrégation d'un grand nombre d'arbres. Dans le cas de tels algorithmes non-interprétables en eux-mêmes, on parlera de modèles « *Boîte-Noire* ».

6.3 *Boosting*

Le *boosting* est une méthode faisant partie de l'apprentissage automatique, elle vise à créer des *classifieurs forts*, plus couramment appelés, « *strong-learners* » par agrégation de *classifieur faible (weak-learners)*. On définit un classifieur faible comme un modèle prédisant au moins aussi bien que le hasard, c'est-à-dire que pour un modèle à deux classes par exemple, il prédit bien dans au moins 50% des cas. Il présente donc des similarités avec la méthode du Bagging évoqué précédemment, néanmoins il est considéré comme plus puissant dans de nombreux cas, c'est pourquoi nous allons développer dans la suite une comparaison plus approfondie entre ces deux branches de l'apprentissage automatique. Puis nous verrons les principes généraux de l'algorithme *Adaboost* qui est l'un des premiers algorithmes qui a repris les principes du Boosting. Ensuite, nous nous intéresserons à la méthode de la *descente du gradient* qui est un préliminaire au *gradient boosting*. Enfin, nous détaillerons l'algorithme *CatBoost* qui sera utilisé pour la modélisation.

6.3.1 Comparaison entre *Bagging* et *Boosting*

Ces deux méthodes sont des méthodes dites ensemblistes car elles ont pour principe de regrouper et d'agréger les résultats d'un ensemble de *weak-learners* avec pour objectif de réduire le biais qui est dû à la faible qualité du weak-learner utilisé initialement et la variance liée à la base de données. Comme évoqué, le principe général de ces deux méthodes, est le même, avec pour objectif d'entraîner des modèles à partir de jeu de données $D_{n'}^{(b)}$ qui sont tirés aléatoirement dans D_n, le jeu initial, puis de les agréger. Néanmoins une première différence apparaît car dans le cas de la méthode Boosting, les individus sont pondérés à chaque nouvelle itération de l'algorithme, un individu mal classé se verra affecter un poids plus important avec pour objectif de mieux capter les individus les plus complexes à prédire du jeu de données. Dans la méthode du bagging les classifieurs sont donc construits en parallèle, et totalement indépendamment les uns des autres, alors que dans la méthode du boosting, les classifieurs sont construits les uns après les autres avec des jeux de données qui ont « appris » des arbres précédents. Enfin, lors de l'aggrégation des *weak-learners*, l'agrégation des modèles se fait à l'aide d'une moyenne simple des résultats des arbres créés alors que la méthode du boosting fait intervenir un autre ensemble de poids pour pondérer les *weak-learners* créés en fonction de leurs taux d'erreur de prédiction, plus un learner aura un bon résultat, plus son poids dans l'agrégation finale sera important.

A priori, le boosting semble meilleur car il fait intervenir une amélioration tout au long de la construction du *strong-learner* ce qui n'est pas nécessairement le cas du bagging, néanmoins dans certains cas les algorithmes faisant intervenir du bagging seront plus pertinents pour un jeu de données ou un problème donné que les algorithmes de boosting. En effet, le bagging sera par exemple, *en général*, moins sensible au phénomène de sur-apprentissage car l'avantage de s'améliorer à chaque itération entraîne aussi une plus grande propension à se calquer *trop bien* sur les données et donc à mal prédire de nouveaux individus.

2. La capacité d'un modèle à être facilement compréhensible et explicable

Méthode	Bagging	Boosting
Principe de construction du *strong-learner*	Agrégation de B *weak-learners*	
Échantillonnage de la base de données	Tirage avec remise des individus dans la base de données	Tirage avec remise des individus dans la base de données repondérée à chaque itération de l'algorithme
Agrégation des résultats des B learners	Moyenne des résultats	Moyenne pondérée selon la qualité de prédiction des résultats

TABLEAU 6.1 – Résumé de la comparaison entre les méthodes Bagging et Boosting

6.3.2 Principes de l'algorithme Adaboost

Comme évoqué, le boosting consiste à créer un grand nombre de *weak-learners* pour obtenir un *strong-learner* par agrégation. Nous allons détailler, dans cette section, l'algorithme Adaboost (pour Adaptative Booosting) développé par Freund et Schapire en 1996[15] qui est l'un des premiers algorithmes fonctionnels faisant intervenir le principe du boosting.

On peut décrire plus précisément l'algorithme Adaboost dans le cas d'une classification :
Soit :

- B le nombre d'itérations que l'on souhaite faire ;
- $\omega^(b)$ le vecteur des poids des individus, que l'on initialise à $1/n$ pour tous les individus ;
- D_n le jeu de donnée d'apprentissage de taille n ;
- $\hat{f}^{(b)}$ l'apprenant faible [3] créé à l'étape b ;
- ϵ_b le taux d'erreur de $\hat{f}^{(b)}$, $\epsilon_b = \sum_{i=1}^{n} \omega_i \mathbb{1}_{\hat{y}_i \neq y_i}$ dans un problème de classification par exemple ;
- α_b le poids de l'apprenant $\hat{f}^{(b)}$, $\alpha_b = ln\left(\frac{1-\epsilon_b}{\epsilon_b}\right)$

Algorithme AdaBoost

for b = 1,..., B **do** :
Tirage avec remise de $D_{n'}^{(b)}$ dans le jeu originel D_n en prenant en compte le vecteur des poids $\omega^(b)$;
Ajuster un apprenant faible $\hat{f}^{(b)}$ sur $D_{n'}^{(b)}$;
Calcul de ϵ_b ;
Si $\epsilon_b > 0.5$ ou $\epsilon_b = 0$ alors arrêt de l'algorithme ;
Sinon :
Calcul d' α_b ;
Calcul du poids suivant : $\omega_i^{b+1} = \omega_i^b e^{\alpha_b \mathbb{1}_{\hat{y}_i \neq y_i}}$ (dans le cas d'une classification).
end for
Agrégation des $\alpha_1 \hat{f}^{(1)}$, ..., $\alpha_B \hat{f}^{(B)}$ pour obtenir un seul strong learner.

6.3.3 Méthode de la descente du gradient

L'objectif de cet algorithme est d'optimiser la minimisation d'une fonction. Pour cela, nous nous plaçons dans un espace $\mathbb{R}^p$ muni de la norme $||.||$ et du produit scalaire $< . >$, soit une fonction f deux fois différentiable, $f : \mathbb{R}^p \to \mathbb{R}$ que l'on souhaite minimiser, $\nabla f(x)$ le gradient de f en x tel que $\nabla f(x) = \left(\frac{\partial f}{\partial x_i}(x)\right)_{1 \leqslant i \leqslant p}$, $x_0 \in \mathbb{R}^p$ la valeur initiale de l'algorithme et un critère d'arrêt $\varepsilon \geqslant 0$. L'algorithme va calculer itérativement $x_1, x_2, \ldots$jusqu'à ce que l'algorithme converge vers l'optimum, c'est-à-dire qu'un test d'arrêt soit satisfait.

Algorithme de la descente du Gradient

for k = 1,..., n **do** :
Calcul de $\nabla f(x_k)$;
Si $||\nabla f(x_k) \leqslant \varepsilon||$ alors on arrête l'algorithme ;
Sinon :
On calcule le pas de l'algorithme $\alpha_k > 0$ qui est soit constant soit calculé à l'aide d'une règle pré-

3. traduction de l'anglais weak-leaner

établie et évaluée à chaque itération ;
$x_{k+1} = x_k - \alpha_k \nabla f(x_k)$.
end for.

Le signe moins devant le gradient dans le calcul de x_{k+1} est lié au fait que la fonction f décroit plus fortement au voisinage de x dans le sens opposé à son gradient.

Nous pouvons maintenant appliquer cet algorithme à la fonction de coût [4] du modèle que l'on cherche à optimiser. Notons donc $\ell(y_i, f(x_i))$ la fonction de coût qui permet de comparer la valeur prédite $\hat{y}_i$ à la valeur réelle de y_i, et $\mathcal{L} = \sum_{i=1}^{n} \ell(y_i, f(x_i))$ la fonction de coût global. Nous cherchons donc à minimiser la fonction $\mathcal{L}$ par rapport aux paramètres de la fonction f associée à notre modèle. Ainsi, nous obtenons la formule suivante :

$$\begin{aligned} \hat{f}^{(b)}(x) &= \hat{f}^{(b-1)}(x) - \alpha \nabla \ell(y, f(x)) \\ &= \hat{f}^{(b-1)}(x_i) - \alpha \frac{\partial \ell(y_i, f(x_i))}{\partial f(x_i)}, 1 \leqslant i \leqslant n \end{aligned} \tag{6.7}$$

6.3.4 *Gradient Boosting*

Les modèles utilisant le *gradient boosting* sont construits de manière séquentielle comme les autres méthodes de *boosting*, néanmoins grâce à l'apport de l'algorithme de descente du gradient, il permet d'utiliser toutes les fonctions 2 fois différentiables comme fonction de coût à optimiser.

Dans la plupart des algorithmes d'apprentissage supervisé on cherche à expliquer une variable réponse Y à l'aide de variables explicatives X, grâce à une fonction $F(X)$ qui approxime le mieux possible Y. Pour cela, nous introduisons la fonction de coût du modèle $\mathcal{L}$. Nous allons détailler l'algorithme du gradient boosting.

Algorithme du Gradient Boosting

Cet algorithme cherche une approximation de $F(X)$ comme étant une somme pondérée de B weak-learners $\hat{f}^{(b)}(x)$.
Initialisation : $\hat{f}^{(0)}(x) = \arg\min_{\lambda} \sum_{i=1}^{n} \mathcal{L}(y_i, \lambda)$ (qui est la classe majoritaire dans le cas d'une classification binaire).
for $b = 1,\ldots, B$ **do** :
Calcul des pseudo résidus : $r_{i,b} = -\frac{\partial \mathcal{L}(y_i, f(x_i))}{\partial f(x_i)}|_{f(x_i)=f^{b-1}(x_i)}$, $1 \leqslant i \leqslant n$;
Ajustement du classifieur $h^{(b)}(x)$ sur les données $(x_i, r_{i,b})_{1 \leqslant i \leqslant n}$, ici on utilise les pseudo résidus comme variables réponse ;
Ajustement du classifieur $\hat{f}^{(b)}(x) = \hat{f}^{(b-1)}(x) + \lambda_b h^{(b)}(x)$, avec λ_b la solution du problème d'optimisation $\lambda_b = \arg\min_{\lambda} \sum_{i=1}^{n} \mathcal{L}(y_i, \hat{f}^{(b-1)})(x) + \lambda h_b(x_i)$;
end for
Enfin, on agrège les $\hat{f}^{(b)}(x)$.

Selon le problème rencontré il faudra choisir une fonction de coût appropriée. On rappelle que l'objectif de celle-ci est d'estimer à quel point le modèle prédit bien les données.

Si l'on est dans un problème de régression, il faudra alors utiliser une fonction de coût qui calcule la différence entre la prédiction et la vraie valeur de l'individu, pour cela, il existe plusieurs méthodes dont certaines vont davantage pénaliser les écarts entre la prédiction et la vraie valeur (comme la fonction de l'écart quadratique), et d'autres vont par exemple présenter l'avantage d'être robuste vis-à-vis des points aberrants des données (comme la fonction valeur absolue).

Dans le cas d'une classification, l'objectif de la fonction de coût sera de quantifier la précision des prédictions du modèle pour chaque classe. La fonction la plus courante dans le cas de la classification est la fonction « *Log-loss* » elle se définit comme suit :

On se place dans le cas d'une variable réponse Y qualitative à K modalités. Soit l'indicatrice y_i^k qui est égale à 1 si l'observation $y_i = k$ et 0 sinon ie. :

$$y_i^k = \begin{cases} 1 \text{ si } y_i = k \\ 0 \text{ sinon} \end{cases}, 1 \leqslant i \leqslant n, 1 \leqslant k \leqslant K. \tag{6.8}$$

4. fonction souvent notée $\ell(y, f(x))$ qui mesure à quel point $f(x)$ prédit bien y

Posons aussi $\pi_k(x_i)$ la probabilité conditionnelle que x_i appartienne à la classe k. Nous pouvons alors écrire la fonction *log-loss* ainsi :

$$\ell(y_i, f(x_i)) = -\sum_{k=1}^{K} y_i^k log(\pi_k(x_i)) \tag{6.9}$$

Ainsi dans le cas d'une classification binaire, où $y_i \in 0, 1$, on peut poser $\pi_1(x_i) = p$ et donc $\pi_0(x_i) = 1 - p$ on obtient la fonction de coût suivante :

$$\begin{aligned}
\ell(y_i, f(x_i)) &= -\sum_{k=0}^{1} y_i^k log(\pi_k(x_i)) \\
&= -y_i^0 log(\pi_0(x_i)) - y_i^1 log(\pi_1(x_i)) \\
&= -y_i^0 log(1-p) - y_i^1 log(p) \\
&= \begin{cases} -1 \times log(1-p) - 0 \times log(p) \text{ si } y_i = 0 \\ -0 \times log(1-p) - 1 \times log(p) \text{ si } y_i = 1 \end{cases} \\
\text{cela peut être réécrit en enlevant les indicatrices} &= -(1-y_i)log(1-p) - y_i log(p)
\end{aligned} \tag{6.10}$$

6.3.4.1 *Tree Gradient Boosting*

Pour la suite, nous nous intéresserons particulièrement aux algorithmes de *Gradient Boosting* qui utilisent les arbres de décision comme apprenant faibles, on parle dans ce cas de « *Tree Gradient Boosting* ». Parmi ces algorithmes nous pouvons citer les algorithmes LightGBM, XgBoost ou encore CatBoost que nous allons détailler dans la suite.

6.3.5 CatBoost

CatBoost, pour **Cat**egorical **Boost**ing, est un algorithme publié entre 2017[9] par Anna Veronika Dorogush et al. et en 2018[11] par Liudmila Prokhorenkova et al. Il reprend les principes du Tree Gradient Boosting mais est plus optimisé et est plus performant, d'après ses auteurs, que les autres algorithmes classiques de la littérature comme le LightGBM et le XgBoost par exemple. Il vise à prédire une variable cible catégorielle ou non à l'aide de variables explicatives qui peuvent être qualitatives ou quantitatives en agglomérant un grand nombre d'apprenants faibles [5]. Il se différencie de l'algorithme de *Forêts Aléatoires* car il est construit de manière séquentielle contrairement à ce dernier pour lequel les apprenants sont construits en parallèle.

6.3.5.1 Attention portée aux variables catégorielles

L'un des objectifs principaux de l'équipe ayant développé l'algorithme *CatBoost* était de développer une méthode de Gradient Boosting qui prennent en compte, le plus efficacement possible les variables explicatives catégorielles. Pour cela, ils permettent aux utilisateurs de l'algorithme de choisir entre deux méthodes de traitements pour ces variables, qui doivent être traitées, dans les deux cas, pendant la phase d'entraînement du modèle et non pendant la partie de préprocessing [6]. La première méthode est la méthode la plus commune des deux et est appelée « *One-Hot-Encoding* », la seconde méthode est appelée « *Target Statistics* » par les auteurs de l'article.

6.3.5.1.1 One-Hot-Encoding

Pour rappel, pour la majorité des algorithmes d'apprentissage automatique, il est nécessaire de transformer les variables qualitatives en *étiquette* [7], néanmoins, pour ne pas inclure une fausse relation de grandeur [8] entre les modalités, il est souvent nécessaire d'ajouter une étape appelée le *One-Hot-Encoding*, cette étape permet de transformer une variable à k modalités en k variables binaires. L'inconvénient principal de cette méthode est que le traitement d'une variable catégorielle avec un grand nombre de modalités utilisera un grand nombre de bits dans la mémoire car elle crée artificiellement un grand nombre de variables.

Prenons l'exemple d'une variable *Situation maritale* à 4 modalités *Marié, Célibataire, Veuf* et *Divorcé*. Après l'étape de transformation en une variable catégorielle, la variable *Situation Maritale* deviendra ceci :

5. Les apprenants faibles seront très souvent des arbres de décisions de "faible" profondeur
6. Étape précédant l'entraînement du modèle, où l'on prépare les données pour obtenir de meilleurs résultats, on peut par exemple exclure les variables aberrantes
7. L'étiquette, ou *label* en anglais est une catégorie d'une variable qualitative, ici les étiquettes seront numériques.
8. Nous ne voulons pas que le modèle interprète que la classe 1 soit inférieure à 2 par exemple car les modalités ne sont pas

ID	Situation Maritale	Catégorie_Sit_Mar	Autres Variables	Variable Cible
1	Célibataire	1	...	y_1
2	Marié	2	...	y_2
3	Veuf	3	...	y_3
4	Divorcé	4	...	y_4

TABLEAU 6.2 – Transformation d'une variable catégorielle en étiquette numérique

L'étape *One-Hot-Encoding* va ensuite transformer la variable *Catégorie_Sit_Mar* en quatre variables binaires.

ID	Célibataire	Marié	Veuf	Divorcé	Autres Variables	Variable Cible
1	1	0	0	0	...	y_1
2	0	1	0	0	...	y_2
3	0	0	1	0	...	y_3
4	0	0	0	1	...	y_4

TABLEAU 6.3 – Exemple de *One-Hot-Encoding* sur la variable Catégorie_Sit_Mar

Comme évoqué, cette méthode transforme la variable *Situation Maritale* qui a 4 modalités en 4 variables binaires différentes cela peut donc impliquer un certain coût lors de l'exécution de cet algorithme car cela peut prendre une grande place dans la mémoire alloué au programme.

Pour éviter cet écueil, les auteurs de l'algorithme *CatBoost* proposent une méthode basée sur les *Target Statistics* que nous allons développer dans la suite.

6.3.5.1.2 Target Statistics

Le principe de base de cette méthode est de remplacer la catégorie des variables catégorielles par un nombre égal à une *statistique cible*. Soit, par exemple, x_i le $i^{\text{ème}}$ individu de la variable X catégorielle, on cherchera donc à remplacer x_i par $\hat{x}_i$. Il faut donc déterminer une stratégie de calcul pour $\hat{x}_i$, qui est généralement la moyenne de la variable cible selon la catégorie, ie. $\hat{x}_i = \mathbb{E}(y|x = x_i)$.

L'équation retenue pour le calcul de $\hat{x}_i$ dans l'algorithme est :

$$\frac{\sum_{j=1}^{n} \mathbb{1}_{\{x_j = x_k\}}.y_j + ap}{\sum_{j=1}^{n} \mathbb{1}_{\{x_j = x_k\}} + a} \tag{6.11}$$

où p est appelé *prior*[9], ce paramètre permet notamment de lisser l'estimation lorsque certaines modalités sont peu représentées et $a > 0$ est un hyper-paramètre permettant de régler l'importance de p.

Toutefois, utiliser cette fonction directement conduit à un problème nommé *Target Leakage*, car $\hat{x}_i$ est calculé à l'aide de ce que l'on cherche à prédire initialement, et cela mène à un autre problème appelé *conditionnal shift*[10], cela se manifeste par le fait que les distributions conditionnelles de $\hat{x}_i|y$ sont différentes pour les jeux de données d'apprentissage et de test. Pour éviter ce problème (et prendre en compte l'ensemble des données disponibles), les auteurs de l'article[11] propose une approche appelée *Ordered TS*. Pour celle-ci, il faut introduire une composante de "temps", définie par une permutation σ, ensuite on utilise tout l'historique disponible pour calculer la TS de x_i. Nous allons détailler cette démarche dans l'exemple suivant, où nous considérerons une variable explicative x a 3 modalités (Paris, Lyon, Marseille) et la variable réponse est une variable binaire.

comparables entre elles.

9. Prior signifie antérieur en français
10. Qui se traduit par Décalage conditionnel

ID	Var x	Variable Réponse
1	Paris	1
2	Lyon	1
3	Paris	0
4	Marseille	1
5	Marseille	1
6	Paris	0
7	Lyon	0
8	Marseille	1

TABLEAU 6.4 – Données initiale

Nous allons donc, dans un premier temps effectuer une permutation des individus, puis nous pourrons calculer $\hat{x}$ à l'aide des observations **précédentes**. En effet, la notion d'historique est ainsi introduite, pour calculer la *statistique cible* de la, on prendra par exemple, uniquement les deux premières lignes en compte. Nous allons expliciter ce principe grâce au tableau suivant :

ID	Var x	Variable Réponse	$\hat{x}$
8	Marseille	1	0,05
1	Paris	1	0,05
5	Marseille	1	0,525
7	Lyon	0	0,05
3	Paris	0	0,525
6	Paris	0	0,35
4	Marseille	1	0,683
2	Lyon	1	0,025

TABLEAU 6.5 – Après transformation

Pour rappel, le calcul de la colonne $\hat{x}$, pour la ligne j se fait ainsi :

$$\hat{x}_j = \frac{\sum_{k=1}^{j-1} \mathbb{1}_{\{x_k = x_j\}} \times y_j + ap}{\sum_{k=1}^{j-1} \mathbb{1}_{\{x_k = x_j\}} + a} \tag{6.12}$$

avec, ici, $a = 1$ et $p = 0.05$. Cette équation peut se traduire par l'équation suivante [11] :

$$\frac{countInClass + prior}{totalCount + 1} \tag{6.13}$$

Où :

- *countInClass* est le nombre de fois, **dans l'historique**, où la variable réponse est égale à 1, pour la catégorie de l'individu dont on veut calculer la statistique cible ;
- *prior* est la variable définie précédemment ;
- *totalCount* est le nombre total d'individu de la catégorie étudiée **dans l'historique**.

Détaillons le calcul de l'individu d'ID 3 et dont la variable x_i = Paris. Pour ce dernier, il faut prendre en compte l'individu 1 mais pas l'individu 6 car il se situe après dans "l'historique" représenté par cette permutation des individus. Les autres individus appartiennent à une autre catégorie, et n'interviennent donc pas dans le calcul de ce $\hat{x}_i$. On a donc :

$countInClass = 1$

11. L'équation est tirée de la page web de l'algorithme CatBoost

$prior = 0.05$

$totalCount = 1$

Ce qui donne :

$$\frac{1 + 0.05}{1 + 1} = 0.525 \tag{6.14}$$

De même, nous pouvons calculer la statistique cible de l'individu dont l'ID est 2 et la variable x_i = Lyon. Il faudra donc prendre en compte, dans le calcul l'individu d'ID 7. Nous pouvons donc calculer la statistique cible grâce à :

$countInClass = 0$

$prior = 0.05$

$totalCount = 1$

Ce qui donne :

$$\frac{0 + 0.05}{1 + 1} = 0.025 \tag{6.15}$$

Il est utile de préciser que pour éviter d'avoir une trop grande variance lors du calcul des *Target Statistics*[12], l'algorithme créé un certain nombre de permutations et en choisis une **aléatoirement** lorsqu'il commence à construire un arbre.

6.3.5.2 *Ordered Boosting*

Le concept d'*Ordered Boosting*, est une solution à un problème identifié par les auteurs de l'algorithme CatBoost[11]. Le problème discuté par les auteurs de l'article est le « *Prediction Shift* » que l'on peut traduire par « *Biais de prédiction* » en français, et est inhérent aux modèles de *Gradient Boosting*. Dans la mesure où tous les arbres de décision sont ajustés sur le même jeu de données, le modèle final s'ajustera moins bien sur des nouvelles données.

L'*Ordered Boosting* reprend le principe de l'*Ordered TS*. En théorie, pour éviter le problème de *Prediction Shift*, en se donnant une permutation σ, il faudrait créer n modèles tel que le modèle M_i $i \in \{1, \dots, n\}$ est entraîné uniquement sur les $i^{\text{ère}}$ observations de σ. À chaque étape, on utilise le modèle M_{j-1} pour calculer les résidus du $j^{\text{ème}}$ individu. Nous pouvons écrire, l'algorithme détaillé ci-dessus, en pseudo-code. Soit $\mathcal{L}$ la fonction de coût choisie pour optimiser le modèle et a la valeur de la formule associée au modèle.

Mise à jour des modèles dans le cadre de l'*Ordered Boosting*

Soit D_n notre jeu de données de taille n ordonné selon une permutation σ_r et T le nombre d'arbres.
$M_i \leftarrow 0$ for $i = 1, \dots, n$;
for $t = 1, \dots, T$ **do** :
  for $i = 1, \dots, n$ **do** :
    for $j = 1, \dots, i-1$ **do** :
      $g_j \leftarrow \frac{\partial}{\partial a}\mathcal{L}(y_j, a)|_{a=M_i(x_j)}$;
    Next j ;
    $M \leftarrow LearnOneTree((x_j, g_j) for j = 1, \dots, i-1)$;
    $M_i \leftarrow M_i + M$;
  Next i ;
Next t
return $M_1 \dots M_n$; $M_1(x_1), \dots, M_n(x_n)$

En pratique, cet algorithme est difficilement applicable car il est d'une grande complexité. En effet, il nécessite de construire n modèles pour chaque permutation et de mettre à jour les estimations $M_i(x_1), \dots, M_i(x_i)$ pour chaque modèle M_i. Ainsi, pour réduire la complexité de l'algorithme, une astuce est employée : au lieu de conserver $O(sn^2)$[13] valeurs, l'algorithme n'en conserve que $O(sn)$ car il ne stocke pas tous les $M_{r,j}(i)$ pour chaque permutation $r \in \{1, \dots, s\}$ mais seulement les valeurs $M'_{r,j}(i) := M_{r,2^j}(i)$ avec $j = 1, \dots, \lceil log_2(n) \rceil$ et $\forall i / \sigma_r(i) \leqslant 2^{j+1}$, on peut ainsi réduire la complexité à $O(sn)$.

12. En effet, les premières statistiques cibles seront moins "fines" que les dernières. En particulier, nous pouvons noter, que la première occurrence de chaque modalité de la variable sera toujours égale à ap/a.

13. Ce nombre correspond aux n modèles que l'on doit entraîner et recalculer ($O(n^2)$) pour chaque permutation, donc avec s permutations, on obtient un algorithme de complexité en $O(sn^2)$

6.3.5.3 Implémentation pratique

Pour résumer, l'algorithme de *CatBoost* a un fonctionnement particulier, avec notamment deux modes différents qui sont le mode *Plain* et le mode *Ordered*. Le premier mode est équivalent aux autres algorithmes de Gradient Boosting basés sur des arbres de décisions (*GBDT*[14]), mais inclut le principe des *Ordered Target Statistics*, tandis que le second mode introduit, en plus des *Ordered TS*, le principe de l'*Ordered Boosting*. Nous allons, dans la suite, décrire le fonctionnement de l'algorithme, puis écrire une version en pseudo-code pour permettre une meilleure compréhension de ses différentes étapes.

Au commencement de l'algorithme, ce dernier crée $s+1$ permutations aléatoires, les permutations $\sigma_1, \ldots, \sigma_s$ serviront à évaluer la qualité des divisions des nœuds de l'arbre, et σ_0 sert à définir la classe (ou le coefficient dans le cas d'une régression) des feuilles des arbres ainsi créés. Plus s est grand, moins la variance des méthodes utilisant les permutations (*Ordered TS* et *Ordered Boosting*) sera importante.

6.3.5.3.1 Construction des arbres

Ensuite, dans le processus de construction des *weak-learners*, l'algorithme va construire des « *Oblivious Trees* ». Les *Oblivious Trees* sont des arbres de décision qui utilisent le même critère de division pour tous les nœuds d'un même niveau de l'arbre. L'intérêt de ce choix, selon les auteurs, est que ces arbres seront moins enclins au surapprentissage.

Si le mode *Plain* est choisi, alors le modèle construira l'arbre selon le processus standard des algorithmes GBDT en conservant, tout de même, s modèles de soutien[15] M_r correspondant aux *TS* calculés à l'aide des permutations $\sigma_1, \ldots, \sigma_s$, s'il y a des variables qualitatives parmi les variables explicatives.

Si le mode *Ordered* est choisi, l'algorithme va, là aussi, conserver des modèles de soutien notés $M_{r,j}$ tel que $M_{r,j}(i)$ soit la prédiction actuelle du $i^{\text{ème}}$ individu basé sur les j^{ers} exemples de la permutation σ_r. Ainsi, à chaque itération t, l'algorithme va sélectionner aléatoirement une permutation σ_r et créé un arbre à partir de cette dernière. Cette permutation intervient d'abord dans l'affectation des TS, puis durant la procédure d'apprentissage de l'arbre. En effet, on calcule les gradients pour chaque $M_{r,j}(i) : grad_{r,j}(i) = \frac{\partial \mathcal{L}(y_i,a)}{\partial a}|_{a=M_{r,j}}$. Ensuite, on utilise la similarité cosinus[16] pour approcher le gradient G, où pour tout i on prend le gradient $grad_{r,\sigma(i)-1}(i)$. L'évaluation de la division des nœuds se fait en calculant la valeur de la feuille $\Delta(i)$ en faisant la moyenne des $grad_{r,\sigma(i)-1}$ des individus faisant partis de la même feuille que i.

Une fois que l'arbre est construit, on va booster tous les modèles $M_{r',j}$. La structure de l'arbre crée est invariable par rapport aux modèles mais les ensembles composant les feuilles diffèrent selon r' et j.

6.3.5.3.2 Choix de la valeur des feuilles

Une fois le modèle F construit, on va utiliser la permutation σ_0 créée à cet effet au début de l'algorithme.

6.3.5.3.3 Algorithme du CatBoost

CatBoost

Entrée : D_n, I, α, $\mathcal{L}$, s, Mode
$\sigma_r \leftarrow$ Permutation aléatoire de $[\![1 : n]\!]$ pour $r \in [\![1 : s]\!]$;
$M_0(i) \leftarrow 0,\ i \in [\![1 : n]\!]$;
If *Mode = Plain* **then** :
    $M_r(i) \leftarrow 0$ pour $r \in [\![1 : s]\!]$ et i tel que $\sigma_r(i) \leqslant 2^{j+1}$;
If *Mode = Ordered* **then** :
    for $j \leftarrow 1$ **to** $\lceil log_2 n \rceil$ **do** :
        $M_{r,j} \leftarrow 0$ pour $r \in [\![1 : s]\!]$ et, $i \in [\![1 : 2^{j+1}]\!]$;
for $t \leftarrow 1$ **to** I **do** :
    $T_t, \{M_r\}_{r=1}^{s} \leftarrow BuildTree(\{M_r\}_{r=1}^{s}, D_n, \alpha, \mathcal{L}, \{\sigma_i\}_{i=1}^{s}, Mode)$;
    $leaf_0(i) \leftarrow GetLeaf(x_i, T_t, \sigma_0)$ pour $i \in [\![1 : n]\!]$;
    $grad_0 \leftarrow CalcGradient(\mathcal{L}, M_0, y)$;
    foreach *leaf j in* T_t **do** :
        $b_j^t \leftarrow -\text{moy}(grad_0(i)$ pour i tel que$leaf_0(i) = j)$;
    $M_0(i) \leftarrow M_0(i) + \alpha b^t_{leaf_0(i)}$ pour $i \in [\![1 : n]\!]$;

14. Sigle de Gradient Boosting Decision Tree
15. Ces derniers sont appelés Supporting Models dans l'article de référence
16. Mesure de l'angle entre deux vecteurs non-nuls

return $F(x) = \sum_{t=1}^{I} \sum_j \alpha b_j^t \mathbb{1}_{\{GetLeaf(x,T_t,ApplyMode)=j\}}$.

Où :

- D_n est un jeu de données de taille n, composé d'observations $\{(x_i, y_i)\}_{i=1}^n$;
- I est le nombre d'itérations de l'algorithme ;
- α est un coefficient correspondant à la taille du pas lors de la descente de gradient ;
- $\mathcal{L}$ est la fonction de coût du modèle ;
- s est le nombre de permutations souhaitées ;
- $Mode \in \{Plain; Ordered\}$ est le mode de boosting choisi ;
- M est le vecteur des modèles déjà construit ;
- $leaf_i$ est la valeur de la feuille i ;
- *GetLeaf* est une fonction qui permet de faire coïncider les observations aux feuilles d'un arbre ;
- $grad_i$ est la valeur du gradient du modèle M_i ;
- *CalcGradient* est une fonction permettant de calculer le gradient d'un modèle M ;
- *ApplyMode* est une fonction permettant d'associer les observations aux feuilles en recalculant les TS sur l'ensemble du jeu de données ;
- b_j^t est la valeur de la feuille j de l'arbre T_t.

Fonction BuildTree

Entrée : M, $D_n, \alpha, \mathcal{L}, \{\sigma_i\}_{i=1}^s, Mode$
$grad \leftarrow CalcGradient(\mathcal{L}, M, y)$;
$r \leftarrow random(1, s)$;
If $Mode = Plain$ **then** :
  $G \leftarrow (grad_r(i)$ pour $i \in [\![1 : n]\!])$;
If $Mode = Ordered$ **then** :
  $G \leftarrow (grad_{r,\lfloor log_2(\sigma_r(i)-1)\rfloor}(i)$ pour $i \in [\![1 : n]\!])$;
$T \leftarrow$ empty tree ;
foreach étape de la procédure Top-Down **do** :
  foreach c, candidat pour la division des nœuds **do** :
    $T_c \leftarrow$ ajout de c à T ;
    $leaf_r(i) \leftarrow GetLeaf(x_i, T_c, \sigma_r)$ pour $i \in [\![1 : n]\!])$;
    If $Mode = Plain$ **then** :
      $\Delta(i) \leftarrow$ moy$(grad_r(p)$ pour p tel que $leaf_r(p) = leaf_r(i))$ pour $i \in [\![1 : n]\!])$;
    If $Mode = Ordered$ **then** :
      $\Delta(i) \leftarrow$ moy$(grad_{r,\lfloor log_2(\sigma_r(i)-1)\rfloor}(p)$ pour p tel que $leaf_r(p) = leaf_r(i), \sigma_r(p) < \sigma_r(i))$ pour $i \in [\![1 : n]\!])$;
    $loss(T_c) \leftarrow cos(\Delta, G)$;
  $T \leftarrow \arg\min_{T_c}(loss(T_c))$;
$leaf_{r'}(i) \leftarrow GetLeaf(x_i, T, \sigma_{r'})$ pour $r \in [\![1 : s]\!]$ et $i \in [\![1 : n]\!]$;
If $Mode = Plain$ **then** :
  $M_{r'}(i) \leftarrow M_{r'}(i) - \alpha$moy$(grad_{r'}(p)$ pour p tel que $leaf_{r'}(p) = leaf_{r'}(i))$ pour $r' \in [\![1 : s]\!]$ et $i \in [\![1 : n]\!])$;
If $Mode = Ordered$ **then** :
  for $j \leftarrow 1$ **to** $\lceil log_2 n \rceil$ **do** :
    $M_{r',j}(i) \leftarrow M_{r',j}(i) - \alpha$moy$(grad_{r',j}(p)$ pour p tel que $leaf_{r'}(p) = leaf_{r'}(i), \sigma_{r'}(p) \leqslant 2^j)$ pour $r' \in [\![1 : s]\!]$ et i tel que $\sigma_{r'}(i) \leqslant 2^{j+1}$;
return T, M.

6.3.5.4 Avantages et inconvénients du CatBoost

Pour conclure sur l'algorithme de CatBoost nous allons discuter des avantages et inconvénients de ce dernier.

6.3.5.4.1 Avantages

- Les algorithmes de boosting profitent en général de l'entraînement séquentiel pour avoir de bons résultats prédictifs ;

- Grâce à l'*Ordered Boosting*, l'algorithme du CatBoost est moins sensible au surapprentissage ;
- Cet algorithme a été créé pour mieux prendre en compte les variables qualitatives que les autres algorithmes de boosting :
- Enfin, il est possible de régler très finement les hyper-paramètres du modèle pour optimiser au mieux le modèle, et obtenir de meilleurs résultats en terme biais/variance et de surapprentissage. On peut citer quelques paramètres :
  - Paramètres liés à la complexité du modèle :
    - Le nombre d'arbre entraîné par le modèle ;
    - La profondeur maximale des arbres ;
    - Gain minimum pour autoriser la division d'un nœud.
  - Paramètres liés au bruit du modèle
    - Ratio de sous-échantillonnage de la base d'apprentissage ;
    - Ratio de sous-échantillonnage des colonnes lors de la création d'un arbre.
  - Le taux d'apprentissage, qui est le coefficient utilisé lors de la descente du gradient pour contrôler la vitesse de convergence de l'algorithme ;
  - Pondération possible dans le cas d'une classification pour éviter d'avoir des résultats déséquilibrés selon les classes.

6.3.5.4.2 Inconvénients

- C'est un algorithme *boîte noire*, dont on ne peut pas expliquer directement les résultats ;
- Il est plus complexe que les autres modèles développés ci-dessus et cela se traduit par un temps de calcul plus important ;
- Comme tous les algorithmes de *boosting* il a une sensibilité accrue aux données aberrante.

Chapitre 7

Apprentissage Automatique Interprétable

Au cours de ce chapitre, nous avons vu plusieurs modèles de machine learning, et un des problèmes de ces algorithmes est leur interprétabilité, c'est pourquoi nous allons ici nous intéresser à la problématique du « *Machine Learning interprétable* ». Pour commencer, nous allons discuter de pourquoi il est important de comprendre et de pouvoir *interpréter* les modèles d'apprentissage automatique. Ensuite, nous essaierons de définir précisément le *Machine Learning Interprétable*. Puis nous définirons le principe d'« *analyse Post-Hoc* » et enfin nous détaillerons le principe de quelques outils que nous utiliserons au cours de ce mémoire.

7.1 Intérêt de la recherche d'interprétabilité

Avant de définir le *Machine Learnig Interprétable*, nous allons nous demander quel est l'intérêt de ce champ de recherche. Pourquoi ne peut-on pas construire des modèles, efficaces si possible, et simplement les appliquer sans nécessairement les comprendre, ni comprendre les associations de variables faites à l'intérieur du modèle.

7.1.1 Cas général

Dans le cas général, il est rare qu'un modèle soit entraîné avec comme seul objectif la prédiction, par exemple dans le cas d'un modèle de détection d'une maladie, il est intéressant de savoir qui va être malade mais peut-être encore plus important de savoir quels sont les indices précurseurs de cette maladie. Il est aussi important de vérifier l'absence de biais discriminant dans les modèles. C'est pourquoi de nombreux chercheurs ont introduit cette notion, l'objectif n'est pas seulement d'avoir une bonne prédiction mais il faut aussi pouvoir comprendre, interpréter et si besoin expliquer les décisions prises par les algorithmes de Machine Learning. Nous allons nous appuyer sur le travail Z. C. Lipton[20] pour décrire les propriétés que doivent vérifier les outils d'interprétations et dans quel contexte ils sont nécessaires.

7.1.1.1 Confiance

La confiance dans un modèle d'apprentissage automatique est une notion relativement floue et soumise à la subjectivité de chacun. A priori, on fait plus confiance à un algorithme que l'on peut comprendre, même si la compréhension de toutes les parties n'est pas toujours nécessaire dans l'absolu. Par exemple, un assuré sera plus à l'aise s'il peut comprendre l'algorithme ayant mené à la tarification de sa prime, mais le fait qu'il ne comprenne pas n'empêche pas qu'il soit assuré.

Aussi, pour certains algorithmes, il est important d'avoir une confiance totale dans la capacité prédictive de celui-ci. Par exemple, dans le cas des voitures autonomes, il est primordial d'avoir "*confiance*" dans l'algorithme et que celui-ci ne se trompe pas en percutant un vélo qu'il n'aurait pas reconnu.

On peut donc considérer que l'on peut faire confiance à un modèle lorsque l'on peut lui laisser le contrôle. Ainsi, on ne va pas seulement s'intéresser à la récurrence des bonnes prédictions mais aussi pour quelles observations le modèle prédit bien. Si le modèle prédit bien là où un humain prédirait bien, on peut considérer qu'il est fiable car on peut estimer qu'il n'y a pas de coût à abandonner le contrôle du modèle. Cependant, si le modèle ne prédit pas bien des exemples qui seraient bien prédits par un humain, alors il ne sera pas fiable.

7.1.1.2 Causalité

Les modèles peuvent être utilisés par les chercheurs pour trouver des relations ou des hypothèses qui seraient généralisables au monde réel. C'est ainsi par exemple que la relation entre le fait de fumer et le cancer des poumons a été établie par Wang et al.[13]. Cependant certaines associations apprises par les modèles peuvent

être erronées car causalité n'entraîne pas corrélation. En effet, il peut toujours exister des données non-observées qui expliqueraient la relation sans que ce soit une corrélation.

7.1.1.3 Transférabilité

Comme nous l'avons vu précédemment, il est commun de séparer le jeu de données en base d'entraînement et base de validation pour superviser l'erreur de généralisation, c'est-à-dire la capacité du modèle à s'adapter à de nouvelles données. Toutefois, les capacités humaines vont bien plus loin dans ce domaine. L'exemple fourni par Caruana et al.[10] illustre cela : le modèle qu'ils ont créé visait à prédire la probabilité de mourir d'une pneumonie. Ce modèle attribuait un plus faible risque aux personnes atteintes d'asthmes ce qui s'expliquait par le fait que ces derniers bénéficiaient de traitements plus agressifs. Néanmoins, si ce modèle avait été appliqué, les patients asthmatiques auraient reçu des traitements moins agressifs et cela aurait sans doute invalidé le modèle.

Aussi, il est important que les personnes ciblées par les algorithmes ne puissent pas en influencer le résultat. Par exemple, dans le cas des prêts bancaires, si le modèle indique qu'un client avec trois cartes bancaires à plus de chance de rembourser son prêt que s'il n'en a que deux, alors le client serait incité à prendre une nouvelle carte bancaire, quitte à la rendre une fois le prêt acquis. Il aura alors pu obtenir son prêt sans que sa capacité de remboursement n'ait fondamentalement changé.

7.1.1.4 Informativité

Une autre propriété qui est peut-être profitable est le fait que le modèle puisse fournir des informations utiles plutôt que d'uniquement avoir un fort pouvoir prédictif. Il est important que l'utilisateur de l'algorithme puisse gagner en connaissance pour ensuite prendre une décision plus pertinente car plus éclairée.

7.1.1.5 Prise de décision juste et éthique

Enfin, il est important de rappeler que certains modèles de machine learning ont vocation à être utilisés dans la vie réelle, il est donc extrêmement important qu'ils soient compréhensibles et le plus transparent possible. Pour reprendre l'exemple du prêt bancaire, un modèle prenant en compte des caractéristiques ethniques serait profondément injuste et inéthique, car il inclurait sans doute des biais selon, par exemple la couleur de peau. En Europe, on peut noter que le **R**èglement **G**énéral sur la **P**rotection des **D**onnées (RGPD) impose un devoir de transparence aux entreprises traitant les données de particuliers.

7.1.2 Cas particulier de l'actuariat

L'utilisation de modèles *boîtes noires* a connu un essor certain au cours des dernières années dans le monde de l'actuariat. Ces derniers sont supposés plus performants mais ne sont pas simplement interprétables, ce qui soulève des interrogations à la fois parmi les assureurs et les assurés ce qui a poussé les régulateurs à intervenir sur ce sujet. En effet, le RGPD impose notamment une exigence de transparence, c'est-à-dire que les assureurs doivent être en capacité d'expliquer leur traitement des assurés. L'assureur doit être capable de justifier toutes les décisions prises de manière détaillée et ne pas transférée la responsabilité des actions à la machine [1]. En France, l'ACPR vérifie notamment qu'il n'y a pas de discriminations induites par la modélisation.

Aussi, certains acteurs du marché des assurances rechignent quelque peu à l'utilisation d'algorithme de machine learning. En effet, il est rare qu'une même personne calibre le modèle et l'utilise, par exemple pour vendre un produit d'assurance à un client. L'utilisateur *in fine*, aura sans doute une préférence pour un modèle facilement compréhensible car il sera alors plus simple pour lui d'expliquer au client les raisons du montant de sa prime par exemple.

Pour les raisons évoquées dans cette section, auxquelles on peut ajouter le fait que l'interprétabilité peut aussi permettre de deboguer les modèles, il semble important de pouvoir expliquer les modèles notamment ceux que l'on qualifie de *boîtes-noires*. Cela se traduit par un nombre conséquent d'articles de recherche sur ce sujet ces dernières années. Nous allons donc maintenant essayer de définir le *Machine Learning interprétable*.

7.2 Définition

Pour commencer, il est important de rappeler qu'il n'y a pas de consensus sur la définition à donner au *Machine Learning Interprétable*. Cela s'explique en partie car il est difficile de définir mathématiquement ce problème[21], en effet de nombreuses méthodes (explication graphique, mathématiques, textuelles etc.) différentes ont été rangées dans cette catégorie. En particulier, le modèle d'interprétation à choisir est très sensible non seulement au problème étudié, mais aussi à la personne visée par l'explication. Il ne faudra pas fournir les mêmes explications du modèle selon que l'on ait à l'expliquer à des collègues ou à des clients par exemple.

1. L'article 22 du RGPD donne le droit de ne pas faire l'objet de décisions entièrement automatisées.

Plusieurs définitions non-mathématiques ont cependant été fournies dans les articles de recherches, arrêtons nous d'abord sur celle de Miller (en 2017)[23] : « *Interpretability is the degree to which a human can understand the cause of a decision.* ». Selon lui, l'interprétabilité se définit donc comme le degré auquel un humain peut comprendre la raison d'une décision (d'un modèle). Ainsi, l'objectif du ML interprétable devient de rendre le modèle *assez* compréhensible pour que l'utilisateur puisse le comprendre et expliquer les choix issus de cet algorithme. Nous pouvons aussi nous noter la définition donnée par Murdoch et al. (en 2019)[12] : « *We define interpretable machine learning as the use of machine-learning models for the extraction of relevant knowledge about domain relationships contained in data.* ». Le ML interprétable est donc défini, dans cet article, comme l'ensemble des méthodes permettant de mettre en exergue des relations présentes dans les données. Ainsi, on peut considérer que l'objectif du *Machine Learning interprétable* est de déceler des causalités dans les données et de permettre ainsi une plus grande compréhension des mécanismes de prédiction sous-jacents à l'algorithme.

Maintenant que nous avons donné une définition au concept de Machine Learning interprétable nous allons détailler le principe de l'*analyse Post-Hoc*.

7.3 Analyse Post-Hoc : définition et principe

Lorsque l'on modélise un problème à l'aide d'un algorithme, qu'il soit boîte-noire ou non, il est intéressant d'analyser le modèle après son ajustement, cela peut permettre, par exemple, de déceler des corrélations induites par le modèle et ainsi de les confronter à nos connaissances. Cette méthode s'appelle l'analyse Post-Hoc [2]. Ce type d'interprétation est très utile pour tout les modèles *black-box* dont on ne peut pas obtenir d'information sur les relations qu'il a établi a priori. Notons qu'il existe différents types de techniques d'interprétation Post-Hoc, on peut noter la différentiation entre les méthodes dites « *Model-Agnositc* » qui s'adaptent théoriquement à tous les modèles, et les méthodes qui ne seront faîtes que pour un type de modèle en particulier. Une autre différentiation que l'on peut faire est celle entre les méthodes d'interprétation qui permettent d'obtenir des informations *globales* sur le modèle, en s'intéressant notamment aux relations qu'il a appris par exemple entre une sous-population de la base de données et la variable réponse. On parlera alors de méthodes d'interprétation *globale*, par opposition aux méthodes d'interprétation locale. Ces dernières seront plus concentrées sur les prédictions individuelles du modèle, avec par exemple le calcul d'un score de contribution des variables dans la prédiction.

7.3.1 Exemple de méthode d'interprétation

Dans ce mémoire nous utiliserons principalement deux méthodes d'interprétation pour nos modèles. Une méthode globale et une méthode locale.

7.3.1.1 *Permutation Feature Importance*

Intéressons-nous d'abord à la méthode globale, la Permutation Feature Importance (PFI), ou importance des variables par permutation en français. On parle de méthode globale car cette méthode est calculée sur l'ensemble du jeu de donnée, cela permet par exemple de comprendre comment le modèle prédit les données dans l'ensemble.

La méthode PFI est une méthode dite model-agnostic, car elle est utilisable pour tous les modèles. Cette méthode a été introduite par [16]. L'idée sous-jacente à cette méthode est qu'une variable est d'autant plus importante que l'erreur du modèle augmentera si l'on permute les valeurs de cette variable. En effet, on peut considérer que si la qualité globale du modèle est grandement modifiée par des modifications des valeurs d'une variable alors cela signifie que le modèle est sensible à ces modifications et donc cette variable à un rôle important. Soient f la fonction associée au modèle, une base de données de taille n tel que $X = (x^{(1)}, \dots, x^{(n)})$, les variables explicatives et $Y = (y^{(1)}, \dots, y^{(n)}$ la variable réponse. On note $\mathcal{L}$ la fonction de perte utilisée. Le calcul de l'erreur des k variables se fait d'après l'algorithme suivant :

- Calcul de l'erreur d'origine du modèle : $err_1 = \mathcal{L}(y, f(x))$;
- Pour $j = 1$ à k :
  - On tire aléatoirement une permutation σ de $1, \dots, n$ dans $1, \dots, n$, puis on définit une nouvelle matrice de variables d'entrée $(\tilde{x}_j^{(i)})_{1 \leqslant i \leqslant n, 1 \leqslant j \leqslant k}$ où l'on modifiera les observations de la variable j en laissant les autres inchangées :

$$\forall i \in 1, \dots, n,\ \forall m \in 1, \dots, k,\ \tilde{x}_m^{(i)} = \begin{cases} x_m^{(i)} \ si\ m \neq j, \\ x_m^{(\sigma(i))} \ si\ m = j \end{cases} ;$$

  - On calcule l'erreur commise par le modèle après cette permutation : $err_j = \mathcal{L}(y, f(\tilde{x}))$;

2. Cette dénomination permet de faire la distinction avec les modèles qui sont interprétables en eux-mêmes et n'ont pas besoin d'analyse Post-Hoc pour être interprétable.

- On calcule l'importance de la variable j à l'aide de la formule $FI^{(j)} = \frac{err_j}{err_1}$

- On obtient les $FI^{(1)}, \dots FI^{(k)}$ en sortie de l'algorithme.

Il est possible d'utiliser plusieurs permutations par variable pour obtenir un résultat plus robuste en utilisant la moyenne des importances ainsi obtenues.

7.3.1.2 SHAP

Nous allons maintenant nous intéresser à une méthode d'analyse locale, la méthode SHAP (Shapley Additive Explanations). Cette méthode est basée sur les valeurs de Shapley introduites initialement en théorie des jeux. Elle a été utilisée en *Machine Learning* pour la première fois en 2010[24] 2014[25]. Nous allons tout d'abord expliquer brièvement le principe des valeurs de Shapley puis nous transposerons ces valeurs à notre problème.

Les valeurs de Shapley ont été définies comme la solution du problème suivant. Un ensemble de joueurs coopère pour obtenir un certain gain total issu de cette alliance. Cependant, les joueurs ne sont pas identiques et ne contribuent pas équitablement au résultat du jeu. La coopération est bénéfique dans le sens où cela apporte plus de bénéfice que des actions individuelles. Le problème est donc de répartir l'excédent ainsi obtenu, et les valeurs de Shapley donne une réponse équitable à celui-ci.

Nous pouvons transposer ce problème aux modèles de prédiction. Ainsi, les joueurs deviennent des variables explicatives et la fonction f associée au modèle est le jeu de coopération et le gain de cette alliance est la prédiction du modèle. Ainsi, nous pourrons quantifier l'importance des variables.

Chapitre 8

Évaluation de la qualité des modèles de classification

Après avoir construit nos modèles, il est important de définir des métriques permettant de mesurer la qualité de nos modèles. Pour cela, nous nous intéresserons à divers indicateurs. Dans un premier temps, nous étudierons le concept de la matrice de confusion, etc.

8.1 Matrice de confusion

La matrice de confusion est une matrice de mesure de la qualité d'un système de classification. Le principe est de représenter dans un tableau le croisement entre la prédiction du modèle et la vraie classe des individus. En pratique, on représentera dans les colonnes les classes estimées par le modèle et la vraie classe sur les lignes (ou inversement). Cette méthode permet de visualiser directement, et facilement, la qualité de prédiction du modèle. On peut représenter cette matrice ainsi :

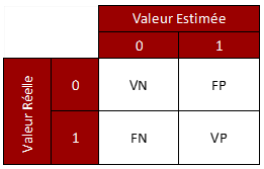


FIGURE 8.1 – Exemple de Matrice de Confusion

où :

- *VN* ou *Vrais Négatifs*, représente les individus de la classe 0 qui ont été bien prédits, dans notre problème, ce sera les non-résiliations bien prédites ;
- *FP* ou *Faux Positifs*, représente les individus qui ont été prédits en tant que 1, alors que ce sont des 0, dans notre cas, ce sera des assurés que l'on prédit comme résilié alors qu'il ne désire pas partir a priori ;
- *FN* ou *Faux Négatifs*, représente les individus qui ont été prédits comme 0 alors que ce sont des 1, dans notre cas, ce sera des assurés résiliés non capté par notre modèle ;
- *VP* ou *Vrais Positifs*, représente les individus de la classe 1 qui ont été bien prédits, dans notre problème, ce sera les résiliations bien prédites.

À l'aide de cette matrice et de ces notations, nous pouvons définir les métriques suivantes :

- L'**exactitude** ou **accuracy** en anglais, décrit le taux d'observations bien prédites ;

$$Exactitude = \frac{VP + VN}{VN + FP + FN + VP} \ ;$$

- Le **taux d'erreur** ou **misclassifiaction rate** en anglais, décrit le taux d'observations mal prédites ;

$$Erreur = 1 - Exactitude = \frac{FP + FN}{VN + FP + FN + VP} \ ;$$

- La **précision** ou **precision** en anglais décrit les éléments vraiment positifs parmi les éléments prédits comme positifs ;

$$Précision = \frac{VP}{VP + FP} \ ;$$

- Le **rappel** ou **recall** en anglais, autrement appelé **sensibilité** (ou **sensitivity** en anglais), désigne la part des positifs bien prédits parmi l'ensemble des vrais positifs ;

$$Rappel = Sensibilité = \frac{VP}{VP + FN} \ ;$$

- La **spécificité** ou **specificity** en anglais représente les éléments vraiment négatifs parmi les négatifs prédits ;

$$Spécificité = \frac{VN}{VN + FP} \ ;$$

- L'**exactitude pondérée**[1] ou **balanced accuracy** en anglais, prend en compte l'éventuel déséquilibre entre les classes et se calcule comme étant la moyenne pondérée de la spécificité et de la sensibilité ;

$$Exactitude\ Pondérée = \frac{Sensibilité + Spécificité}{2} \ ;$$

- Le **score-F_1** ou **F_1-score** en anglais, est la moyenne harmonique entre la précision et le rappel

$$F_1 = \frac{2}{\frac{1}{Rappel} + \frac{1}{Précision}} \ ;$$

- Le **score-F_β** ou **F_β-score** en anglais, est la généralisation du score F_1 où l'on ajoute un coefficient β qui est choisi de telle façon que le rappel soit β fois plus important que la précision.

$$F_\beta = (1 + \beta^2) \times \frac{Précision \times Rappel}{(\beta^2 Précision + Rappel}.$$

Parmi ces métriques, nous aurons une attention particulière pour le score F_2 et l'exactitude pondérée, ces deux métriques présentant l'avantage de réaliser un compromis entre le fait de bien prédire les deux classes. Le fort déséquilibre entre les classes rencontré dans le cadre de la modélisation nous a en effet incités à utiliser ces métriques.

8.2 Courbe ROC et AUC

La courbe ROC est un indicateur graphique, couramment utilisé, permettant de juger de la qualité d'un modèle. Pour tracer cette courbe, il faut représenter le taux de vrais positifs ($\frac{VP}{VP+FN}$) en fonction du taux de faux positifs ($\frac{FP}{FP+VN}$). Cette courbe permet elle aussi un compromis intéressant dans le cas de classes non-équilibrées.

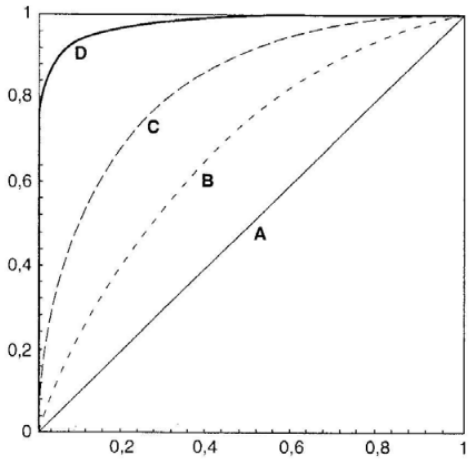


FIGURE 8.2 – Exemple de courbe ROC (source : (source : site web))

Nous pouvons voir plusieurs exemples de courbes ROC sur la figure 8.2. Le modèle correspondant à la courbe A est un modèle qui n'est pas plus performant que l'aléatoire. Le modèle le plus performant est le D, c'est en effet celui qui est le plus proche du coin supérieur gauche.

L'AUC se définit comme étant l'aire sous la courbe ROC, il varie entre 0.5 et 1. Une AUC de 0.5 signifie que le modèle est aussi efficace que l'aléatoire et une AUC de 1 signifie que le modèle est parfait.

1. Aussi appelée "justesse pondérée" dans ce mémoire

Troisième partie

Présentation et étude de la base de données

Chapitre 9

Traitements

Ce mémoire a nécessité un grand nombre de traitements pour nettoyer la base de données. Ces derniers vous seront présentés dans ce chapitre, puis nous verrons quelques éléments de statistiques descriptives dans le chapitre suivant.

Tout d'abord, ces traitements ont été réalisés sur JupyterLab, sous l'environnement de travail Spark[8]. Ce dernier est très performant pour gérer de grosses bases de données permettant des analyses complexes à grande échelle. Il permet en effet d'optimiser et d'accélérer les tâches de l'environnement Hadoop sur lequel il se base.

9.1 Présentation des variables disponibles dans la base de données

La base de données sur laquelle repose ce mémoire fait avant tout traitement, 2 030 871 lignes et 262 colonnes. Elle inclut 11 entités de courtages réparties en 6 groupes, pour un total de 72 polices différentes. Cela comprend des polices dites standard et des polices risques aggravés. Cette base est une concaténation de différentes bases *image* que nous recevons mensuellement de nos partenaires. Néanmoins les bases ayant des architectures différentes d'un partenaire à l'autre, et parfois même d'un mois à l'autre pour un seul partenaire, nous n'avons pas pu intégrer l'ensemble des partenaires de l'Équité. Parmi les 262 colonnes, on peut par exemple retrouver toutes les informations nécessaires à la tarification d'un produit automobile, mais aussi l'historique de sinistres ou les éventuelles aggravations de l'assuré, s'il est malussé.

9.1.1 Variables d'intérêts quant à l'étude de la résiliation

Pour étudier la résiliation, trois variables nous ont semblé intéressantes car elles contenaient des informations sur l'état du contrat et des mouvements qu'il avait subis :

- État qui contient l'état du contrat (En cours, Résilié, Suspendu etc.) ;
- Mouvement_Maille qui contient le mouvement auquel correspond l'image (Affaire Nouvelle, RAS etc.) ;
- Motif_Mouvement qui contient la "raison" du dernier mouvement (Résiliation + motif, suspension + motif etc.).

Notons déjà que selon les différents groupes, les données manquantes parmi ces trois variables, étaient plus ou moins nombreuses. En effet, il s'est avéré que seul la variable Motif_Mouvement contenait assez d'informations pour discriminer les résiliations qui nous intéressaient. Les informations des autres variables étant plus parcellaires, elles ne permettaient par exemple pas de distinguer les résiliations venant de l'assuré, de celle venant de l'assureur. Malheureusement, seulement 3 des groupes partenaires fournissent l'information du Motif_Mouvement.

9.2 Ajout de variables

L'objectif des traitements est d'obtenir une base permettant une modélisation, pour cela il faut faire des choix concernant certaines colonnes :

- certaines variables seront éliminées car non pertinentes pour notre problème ;
- d'autres seront retravaillées pour éviter les valeurs manquantes ou aberrantes ;
- enfin il pourrait être nécessaire d'ajouter certaines variables pour notre modélisation.

Le premier traitement réalisé a été d'uniformiser les encodages pour mieux appréhender chaque variable et ses modalités. Ensuite, pour pouvoir recréer certaines variables nous avons restreint la base à seulement 22 variables, avec notamment les dates de début et de fin d'image et d'effet du contrat, ou les primes détaillées par

garantie, par exemple. Ensuite, chaque partenaire a été traités séparément car chacun nécessitait des traitements adaptés. Par après, certaines variables ont été modifiées, d'autres ajoutées et enfin une jointure avec l'ensemble de la base a été réalisée.

9.2.1 Variables reliées à la prime

Tout d'abord, il a fallu calculer la Prime TTC à partir des primes hors taxe par garantie. Pour cela, nous avons ajouter le taux de commission de chaque partenaire ainsi que le taux de taxe par garantie dans la base. Ensuite, nous avons trouvé intéressant de calculer l'évolution de la prime entre les images.

9.2.1.1 Zoom sur l'évolution de prime

Pour calculer l'évolution de prime, nous avons donc créé des variables PRIME_HT N-1 et PRIME_TTC N-1 contenant respectivement la prime Hors Taxe ainsi que la prime TTC calculée précédemment de l'image précédente. Nous avons ensuite calculé l'écart entre la prime N et la prime N-1 ainsi que l'écart en pourcentage. Néanmoins, nous nous sommes rendu compte à cette étape d'un problème concernant les évolutions de prime, en effet beaucoup sont égales à 0, notamment sur l'image correspondant à la résiliation. Cela s'explique entre autres par le fait que certains partenaires créent une image pour la résiliation (exemple ci-dessous) mais d'autres raisons nous sont apparues plus tard.

id_contrat	date debut image	date fin image	Prime TTC N-1	Prime TTC	Evol Prime TTC	Evol Prime TTC %age
1	01/01/2021	01/05/2021	100	150	50	50
1	02/05/2021	02/05/2021	150	150	0	0

Tableau 9.1 – Exemple du problème rencontré en construisant les variables d'évolution de prime

De plus, certains partenaires ayant de bons résultats n'ont pas eu d'augmentations tarifaires ces dernières années ce qui explique aussi que certaines évolutions de prime soient faibles, voire nulles. En conséquence, nous re-modifierons ces variables plus tard dans le traitement de la base en fonction de divers paramètres.

9.2.2 Ajout d'un *split* annuel

Pour permettre plusieurs calculs, dont celui du taux de résiliation, nous avons choisi de découper les images en "splitant" au 1^{er} janvier lorsque l'image commençait lors d'une année N et finissait en année N+1 ou plus. En procédant ainsi, nous dupliquons donc les lignes et les évolutions de prime que nous avons vues précédemment restent calculées sur les dates de l'image d'origine.

id_contrat	date debut image	date fin image	date debut split	date fin split	Autres infos
1	01/10/2020	01/05/2021	NA	NA	...
1	02/05/2021	10/09/2021	NA	NA	...

Devient

id_contrat	date debut image	date fin image	date debut split	date fin split	Autres infos
1	01/10/2020	01/05/2021	01/10/2020	31/12/2020	...
1	01/10/2020	01/05/2021	01/01/2021	01/05/2021	idem qu'au-dessus
1	02/05/2021	10/09/2021	02/05/2021	10/09/2021	...

Tableau 9.2 – Exemple du split annuel

Cet ajout du split annuel duplique donc les autres variables de la base, et notamment les variables d'évolution de primes créées précédemment. Cela pose problème car, en l'état, on aurait dans la base des évolutions qui ne correspondraient pas entre les différentes images. La possibilité de créer les variables "Split Annuel" avant a été rapidement éliminée car toutes les images nouvellement créées et correspondant à la deuxième partie, i.e. entre le 1^{er} janvier et la date de fin d'image, auraient eu des évolutions nulles. Le problème des évolutions de primes induits par l'ajout du split annuel sera corrigé dans la prochaine section.

9.2.3 Ajout des dates de résiliation

La colonne contenant la date de résiliation est très partiellement remplie, déjà parce qu'évidemment tous les contrats ne sont pas résiliés dans la base, mais aussi parce que la variable n'a pas été bien peuplée à la création de la base. Néanmoins, en comparant les bases images directement envoyées par les partenaires et la base chargée sur Hadoop, on peut s'apercevoir que, pour les contrats résiliés (identifiées grâce aux variables détaillées en 9.1.1), les dates de résiliation et de fin de la dernière image sont les mêmes. De plus, chez certains partenaires, nous pouvons avoir pour un seul contrat plusieurs dates de résiliation, dans ce cas nous avons choisi de prendre la dernière disponible.

9.2.4 Ajout des variables d'âge ou d'ancienneté

Après étude préliminaire de la base, nous nous sommes aperçus que plusieurs variables d'ancienneté avaient des modalités aberrantes, par exemple la variable Ancienneté Contrat était mal remplie, avec notamment des anciennetés non strictement croissante. Pour corriger ceci, nous avons décidé de recalculer ces variables "à la main". Les variables concernées sont les suivantes :

- AGE_CP ;
- Ancienneté_Permis ;
- Ancienneté_Vehicule ;
- Ancienneté_Contrat.

9.2.5 Ajout des variables année et exposition

Dans la mesure où nous avions créé des nouvelles variables de date, il a été nécessaire de créer aussi de nouvelles variables année et exposition pour remplacer les autres. Pour rappel, l'exposition d'une image se calcule en divisant le nombre de jours entre la date de début et la date de fin de la période étudiée par le nombre de jours dans l'année. Ces variables seront notamment utiles pour réaliser les études statistiques sur la base.

9.2.6 Modification des variables liées aux sinistres

Comme nous l'avions évoqué précédemment, la base de données contient des informations sur les sinistres des assurés. Ces dernières viennent pour grande partie de notre base Sinistres et par conséquent, tous les assurés ne sont pas concernés, et les sinistres sont détaillés par garantie. Or nous souhaitons avoir des informations complètes et pertinentes dans notre base, pour ce faire, nous avons décidé de regrouper d'une part les sinistres responsables et de l'autre les sinistres non responsables. Pour les assurés n'ayant pas eu de sinistre, nous avons évidemment remplacé la valeur "None" par 0. Les résultats obtenus par ces calculs correspondent à peu près à la variable "NB_SIN_ANNEE" qui provient elle des bases images.

9.3 Ajout de la variable de modélisation

Pour modéliser la résiliation, nous avons souhaité créer une variable binaire égale à 1 à la dernière image des contrats résiliés, et 0 sinon. Pour ce faire, il a d'abord fallu sélectionner les motifs de résiliation qui nous intéressaient et qui rentraient dans notre périmètre. Parmi les partenaires qui fournissent les motifs de résiliations, nous avions en effet un large spectre de motifs différents. Or, l'objet de ce mémoire est de capter les résiliations de l'assuré, c'est-à-dire, lorsque l'assuré décide de lui-même de changer d'assureur. En effet, il n'est pas très intéressant dans notre cas de se pencher sur l'ensemble des résiliations, d'autant plus que les assurés ayant fait une demande d'assurance mais qui n'ont pas envoyé leurs pièces dans le délai imparti apparaissent quand même dans les bases images de nos partenaires, cela amène donc un grand nombre de personnes "résiliées" mais qui ne nous intéressent pas dans le cadre de notre modélisation.

Dans notre objectif de modéliser les résiliations assurés qui seraient dues à une augmentation du tarif, nous nous sommes plus ou moins restreints aux motifs suivants :

- Résiliation loi Châtel ;
- Résiliation loi Hamon ;
- Résiliation à l'échéance.

Nous avons donc ajouté une variable "Résil_Perim" pour identifier les résiliations qui sont dans notre périmètre. Elle est donc égale à 1 si l'assuré à un motif de résiliation équivalant aux 3 cités ci-dessus et que nous sommes à la dernière image du contrat et 0 sinon.

De plus, pour ne pas introduire de biais par rapport aux résiliations qui ne rentrent pas dans le périmètre, nous avons écarté de la base de modélisation les assurés qui quittent le périmètre d'étude sans avoir résilié selon les critères définis ci-dessus. En effet, les laisser signifierait que l'on considère qu'ils seraient restés s'ils n'avaient pas été résiliés par la compagnie ou le courtier.

9.3.1 Correction des évolutions

Cela nous ramène au problème évoqué en 9.2.1.1, certaines primes sont donc dupliquées alors que l'on devrait parfois observer l'augmentation sur la $1^{ère}$ image et parfois sur la $2^{ème}$ image. En effet, l'objectif du mémoire est de modéliser la résiliation suite à une augmentation du prix, or si l'assuré résilie 6 mois après son anniversaire, on peut raisonnablement penser que la raison de son départ n'est pas la dernière augmentation de primes subite. Ces évolutions de prime dupliquées pourraient donc biaiser notre modèle. Pour remédier à ce problème, nous avons regardé le temps moyen de réflexion avant la résiliation. En effet, une fois que nous aurons cette information, on pourra décider à partir de quels laps de temps on pourra raisonnablement penser que les assurés ont résilié après avoir constaté l'augmentation de leur prime, les autres verront l'évolution de leur prime réduite à 0 [1].

9.3.1.1 Étude du temps de réflexion des assurés

Pour étudier le temps de réflexion des assurés, nous avons comparé le mois de la date de résiliation et le mois de la date d'anniversaire du contrat qui a été définie comme le maximum entre la date effet du contrat et les éventuelles dates d'avenant, comme le montre l'exemple ci-dessous.

DATE_EFFET	DATE_AVT	DATE_RESIL_Corr	Mois_Anniv	Mois_Résil	Écart
19/10/2017	19/10/2018	19/10/2020	10	10	0
08/07/2016	null	08/07/2019	7	7	0
19/10/2016	25/01/2019	19/10/2019	1	10	9

Tableau 9.3 – Exemple de la création de la variable Écart

Nous avons donc créé une variable Écart représentant le nombre de mois entre l'anniversaire et la résiliation. En représentant le nombre de résiliation dans la base par nombre de mois d'écart, et la moyenne pondérée de cette série, on pourra estimer le nombre de mois de réflexion à partir duquel on peut considérer que l'assuré n'a pas résilié à cause de la dernière hausse de sa prime. Après cela, nous pourrons corriger les évolutions de prime.

1. Les différents cas seront décrits ci-après

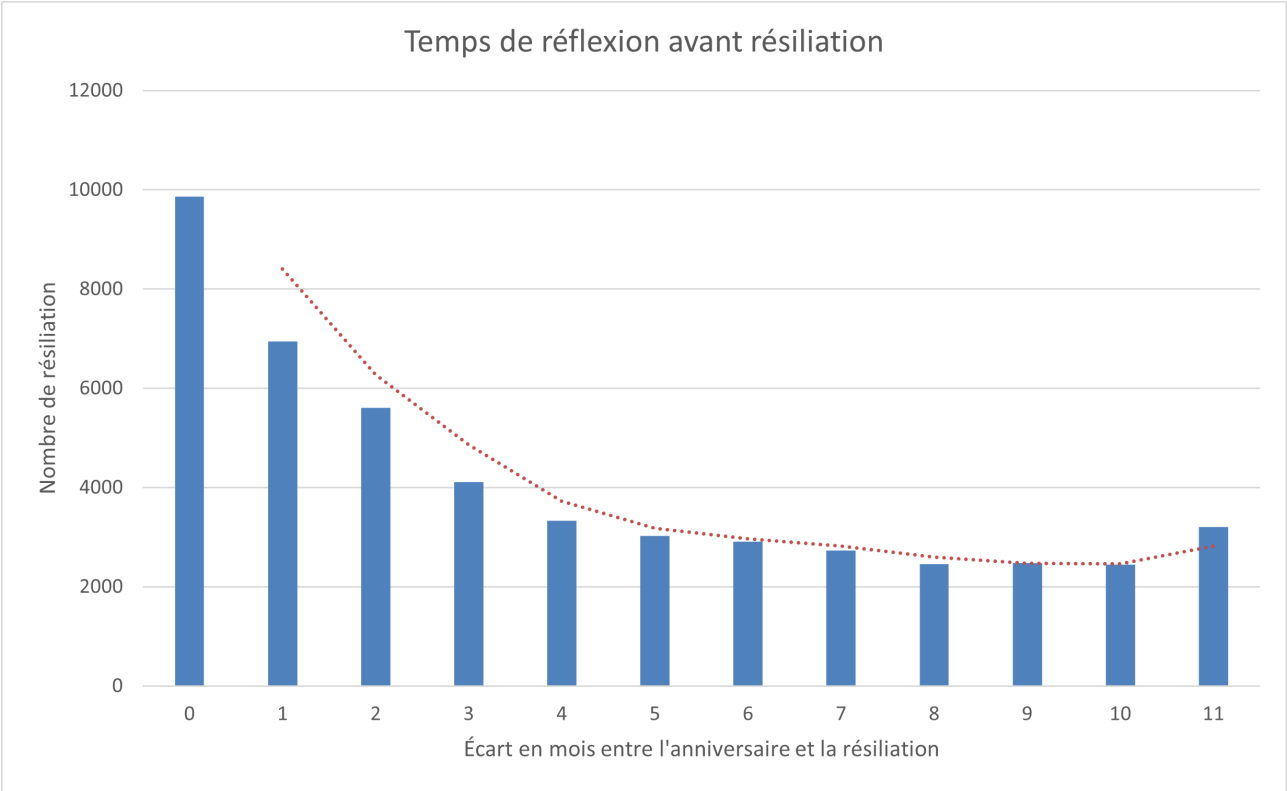


FIGURE 9.1 – Graphique du temps de réflexion

La première chose que l'on peut remarquer est que les résiliations sont concentrées autour des mois suivants l'échéance du contrat. Aussi, on peut noter que le nombre de résiliation remonte 11 mois après l'échéance, cela s'explique sans doute par le fait que les assurés ayant reçu leurs avis d'échéance peuvent anticiper leurs résiliations, après avoir remarqué que leur prochaine prime sera plus élevée.

Après étude du graphique ci-dessus ainsi que de la moyenne pondérée du tableau du nombre de résiliation par mois d'écart, nous avons établi le temps de réflexion moyen à 3 mois. Ce choix a été réalisé car bien que la moyenne de la série soit égale à 4, on peut observer graphiquement un saut assez important entre les barres 3 et 4, alors que le saut entre les barres 4 et 5 est assez faible. Le fait que le nombre de résiliation de 4 à 10 mois d'écart soit relativement constants implique qu'il y ait un comportement "moyen" des assurés pour ces temps de réflexion.

9.3.2 Modification de la base

Pour modifier la base après cette étude, nous avons établi plusieurs cas de figure. Pour commencer, on peut soustraire des cas les lignes qui n'ont pas été dupliquées. Ensuite, pour les images dupliquées où le contrat n'est pas résilié, on décide arbitrairement que les évolutions de la première image seront nulles et que les autres ne changeront pas. Enfin, si l'image qui a été dupliquée, et que l'assuré a résilié à la fin de cette image, alors selon l'écart entre les vrais débuts et fins de l'image, on modifiera l'évolution de la première ou de la seconde partie de l'image "splitée".

Tout ceci est résumé dans le schéma ci-dessous :

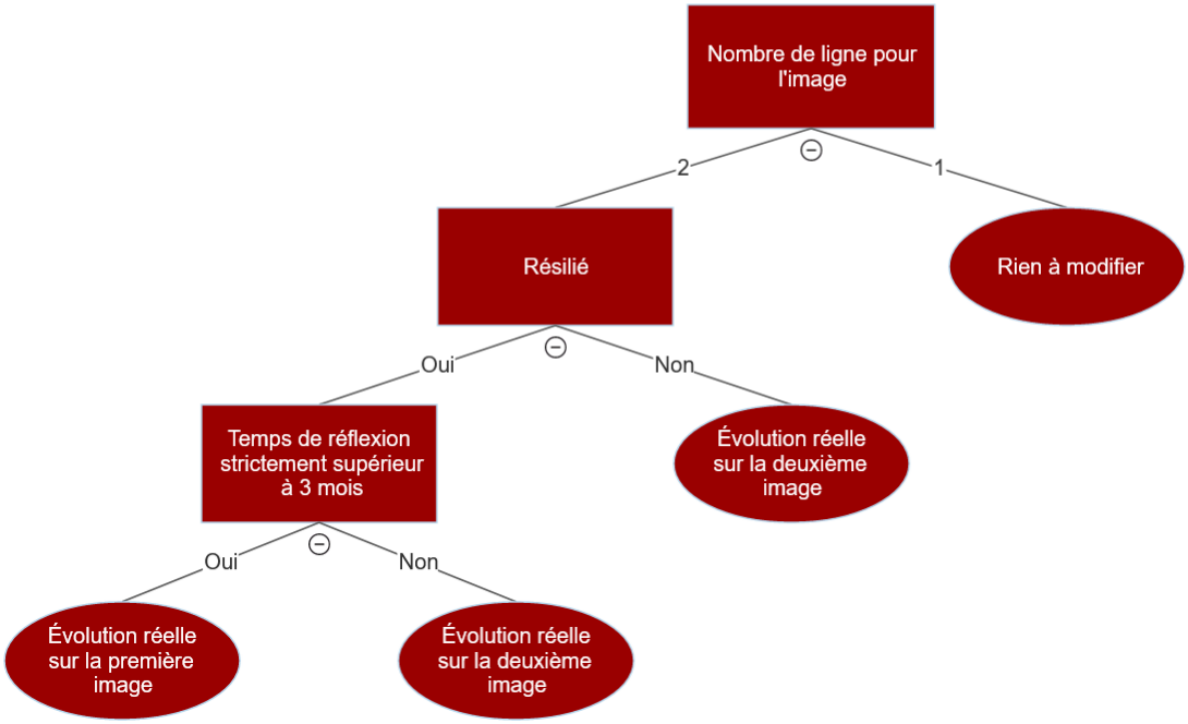


FIGURE 9.2 – Schéma correction des évolutions de prime

9.4 Conclusion

Nous avons donc vu, dans ce chapitre, un grand nombre de modifications réalisées sur notre base de données qu'il était nécessaire de réaliser avant même de pouvoir réaliser des études statistiques sur notre base. Certaines variables ont dû être retravaillées voire recalculées, d'autres ont été ajoutées à la base. Nous allons maintenant pouvoir commencer l'étude descriptive de la base.

Chapitre 10

Statistiques descriptives

Dans ce chapitre, l'objectif sera de comprendre plus avant notre base ainsi que de pouvoir permettre de comparer nos données aux données du marché ou d'autres mémoires d'actuariat. Nous allons d'abord étudier la base dans l'optique de définir les contours de la base qui servira à la modélisation. Par la suite, nous allons étudier la base de modélisation définie avant avec des analyses univariée et bivariée.

10.1 Études préliminaires

Dans cette section, l'objectif sera de réaliser des études préliminaires permettant de définir notre base de données de modélisation.

10.1.1 Limitation temporelle

Le premier objectif est de sélectionner les années qui sont assez représentées dans la base pour pouvoir être utilisée dans la modélisation. La base de données étant composée de bases images mensuelles allant de janvier 2018 à décembre 2021, nous avons donc représenté graphiquement les expositions calculées précédemment, en les regroupant par année et par contrat pour comptabiliser le temps où chaque contrat reste en portefeuille sur l'année. Ainsi une exposition à 1 signifie que l'assuré a été en portefeuille pendant les 365 jours de l'année, et une exposition à 0,5 signifie que l'assuré est resté à peu près 180 jours en portefeuille.

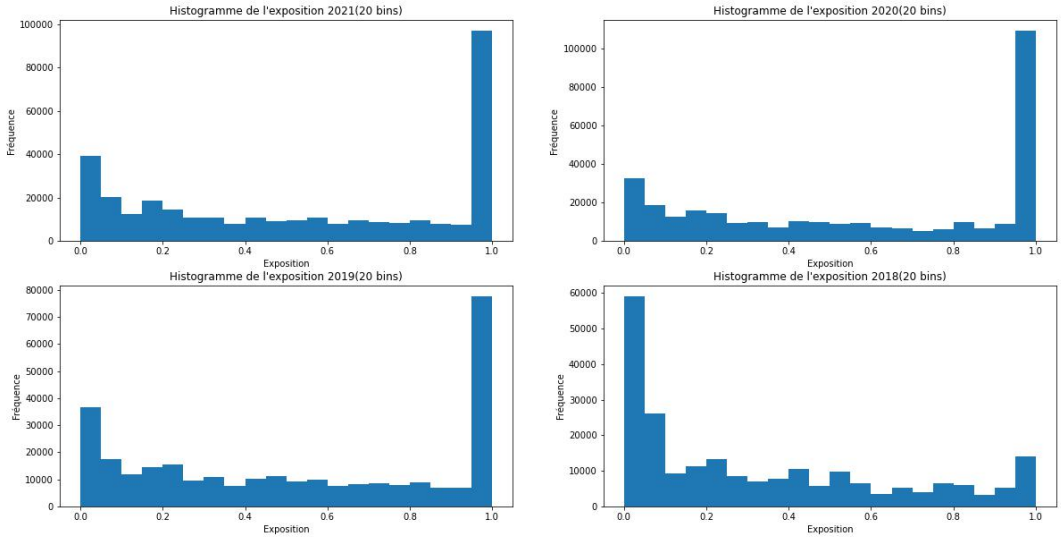


FIGURE 10.1 – Exposition par année du portefeuille

Nous observons directement que les années 2019, 2020 et 2021 semblent mieux représenté dans la base de données que l'année 2018. Cela s'explique aisément par le fait que nos données commencent avec les bases images

de l'année 2018 et que certaines images ne commencent pas au 1er janvier, les contrats n'auront pas forcément l'entièreté de leur année qui sera affichée.

Prenons l'exemple d'un contrat prenant effet le 1^{er} août 2017 et dont la fin sera le 31 juillet 2020. La base a été créée de façon à ne considérer que les contrats ayant eu un mouvement dans le mois de la "partition" [1] et les images ne doivent théoriquement pas dépasser 1 an. En admettant que ce cadre soit respecté et que l'assuré n'ait pas contracté d'avenant au cours de la vie de son contrat, on aurait les images suivantes dans notre base :

DATE_EFFET	DEBUT_IMAGE	FIN_IMAGE	DEBUT_SPLIT	FIN_SPLIT	Exposition	MOIS_OBS
01/09/2017	01/09/2018	31/08/2019	01/09/2018	31/12/2018	0,25	092018
01/09/2017	01/09/2018	31/08/2019	01/01/2019	31/08/2019	0,75	092018
01/09/2017	01/09/2019	31/08/2020	01/09/2019	31/12/2019	0,25	092019
01/09/2017	01/09/2019	31/08/2020	01/01/2020	31/08/2020	0,75	092019

Tableau 10.1 – Exemple des éventuels problèmes d'expositions

Pour ce contrat, l'exposition serait donc de 0.25 en 2018, 1 en 2019 et 0.75 en 2020. Cela aura un impact sur le graphique alors que l'on peut raisonnablement penser (au vu de la date d'effet) que le contrat a aussi une exposition de 1 sur l'année 2018. Néanmoins, on ne peut pas ajouter artificiellement des expositions comme cela sans biaiser notre modèle, voilà pourquoi nous allons, pour notre modélisation, nous restreindre aux années 2019, 2020 et 2021. En effet, ces années là ont des graphiques d'exposition cohérents avec une majeure partie des contrats qui sont tout au long de l'année en portefeuille. On observe un pic plus haut pour des expositions proches de 0, cela est sans doute dû à un problème déjà évoqué dans ce mémoire qui est que nos partenaires nous communiquent les informations des assurés qui n'envoient pas leurs pièces. Ces derniers étant automatiquement résiliés après peu de temps ils ont une exposition très faible dans la base de données. Pour cette raison, nous avons décidé de supprimer, dans la suite, les expositions trop faibles car ces dernières introduiraient un biais important dans les données.

10.2 Analyse univariée

Dans le but, de mieux connaître notre base de données nous avons représenté graphiquement le taux de résiliation en fonction de diverses variables. Cette partie permettra de se représenter, avant la modélisation, d'éventuelle corrélation entre les variables explicatives et la variable réponse. Nous ne représenterons pas toutes les variables dans cette partie du mémoire, nous avons néanmoins fait des graphiques pour chaque variable.

10.2.1 Variables liées aux primes

Pour commencer, nous allons nous intéresser aux variables liées aux primes, à savoir, la prime TTC, l'évolution de la prime entre deux périodes, et l'évolution représentée en pourcentage.

On observe sur le graphique 10.2 que le taux de résiliation est légèrement croissant en fonction du montant de prime. Cette croissance semble constante entre 100 et 2500, avant et après cet intervalle, la faible exposition rendent le taux de résiliation brut très volatil, néanmoins le taux lissé permet tout de même de se représenter la tendance. On observe aussi que la distribution des montants de prime semble plus ou moins suivre une loi normale.

Sur le graphique 10.3, nous pouvons constater une forte concentration autour de 0, cela se justifie par le fait que certains partenaires aient décidé de ne pas augmenter les primes pendant la pandémie de COVID-19, de plus, de nombreux assurés ont quand même résilié malgré une évolution de prime proche de 0, possiblement car ils ont profité de cette période pour changer d'assureur. Par ailleurs, on observe une tendance croissante pour la résiliation en fonction de l'évolution de prime, en effet, si l'on ne prend pas en compte la classe proche de 0, on constate que les courbes de taux de résiliation brut et lissé sont globalement en hausse.

Enfin, sur le graphique 10.4, nous pouvons une nouvelle fois constater un pic important pour une évolution nulle. Nous pouvons aussi observer un pic de résiliation autour d'une baisse de plus de 10% de la prime pour les années 2019 et 2020, cela peut être consécutif à une volonté d'un courtier de mettre en place des mesures de protection de portefeuille qui n'auraient pas eu les effets escomptés. Sur ce même graphique, nous pouvons observer, une concentration des résiliations lorsque l'évolution de prime est faible. Nous remarquons aussi que le taux de résiliation reste stable à partir d'un certain seuil (dépendant de l'année) d'augmentation de prime, cela peut facilement s'expliquer par le fait que l'on capte ici sans doute des assurés qui modifient leurs conditions

1. Mois d'observation de la ligne, équivalent à MOIS_OBS

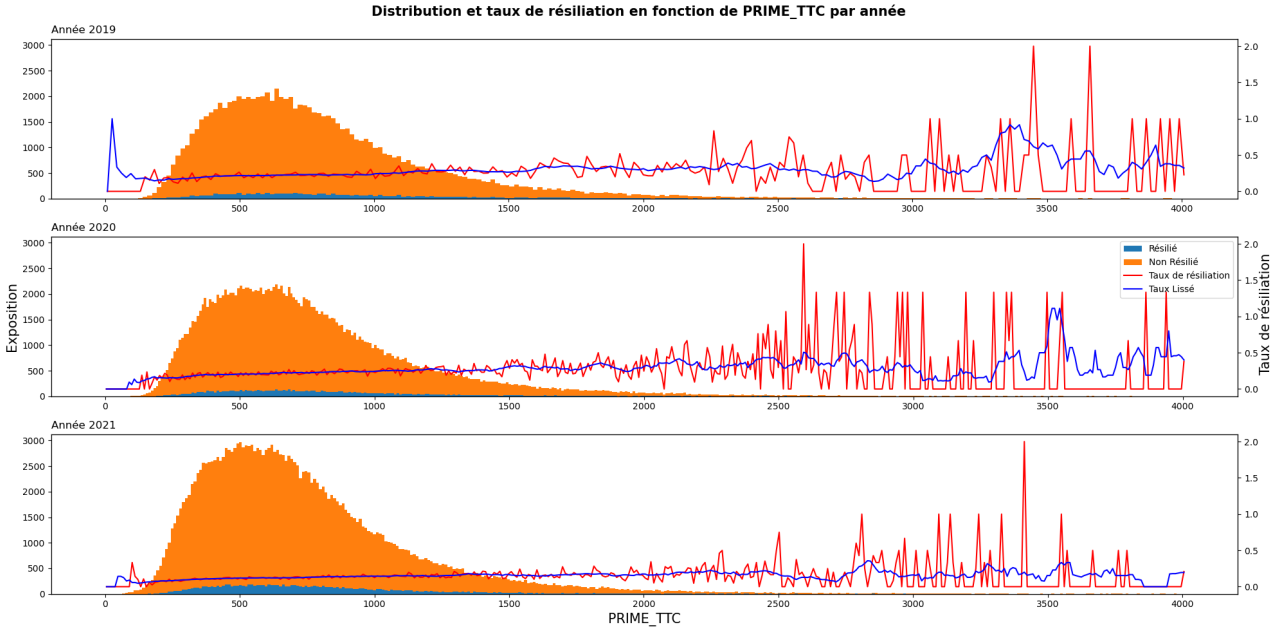


FIGURE 10.2 – Représentation du taux de résiliation en fonction de la variable *PRIME_ TTC*

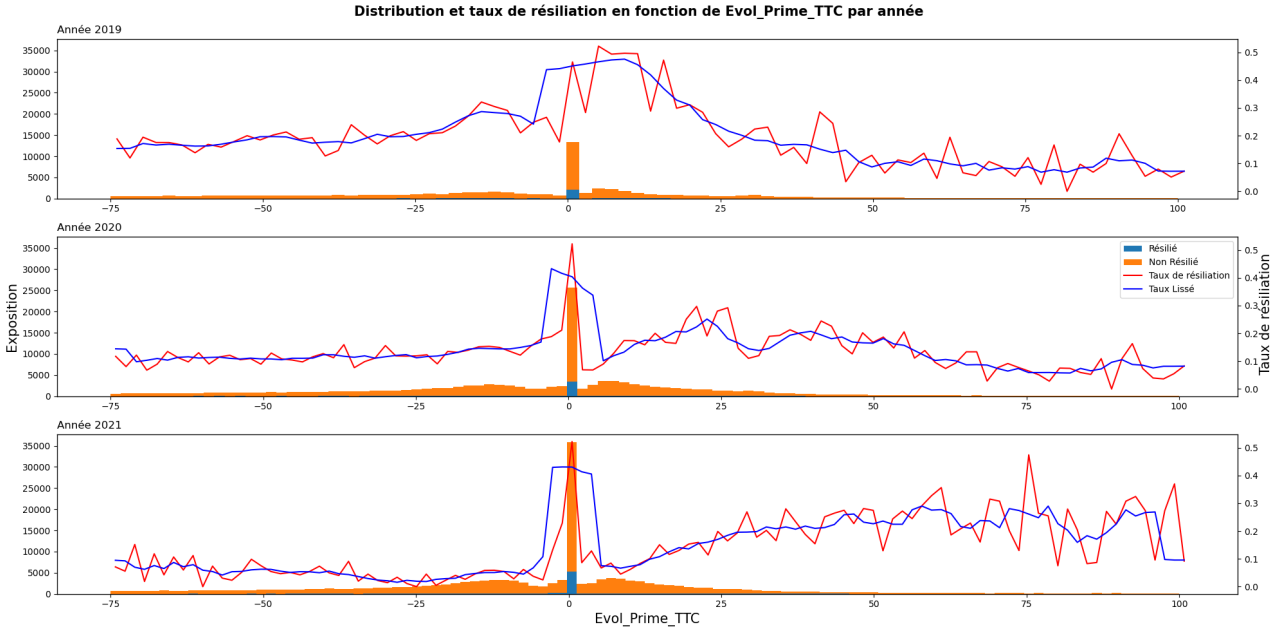


FIGURE 10.3 – Représentation du taux de résiliation en fonction de la variable *Evol_ PRIME_ TTC*

d'assurance (en changeant par exemple de formule ou de voiture) ainsi, l'augmentation est importante mais logique du point de vue de l'assuré.

Pour conclure sur ces graphiques, nous pouvons aussi remarquer que les tendances sont différentes selon l'année prise en compte, notamment pour les variables basées sur l'évolution de la prime. Cela peut s'expliquer par le fait que certaines de ces années ont été atypiques, avec l'émergence de la COVID-19 et les différents confinements qui ont suivi.

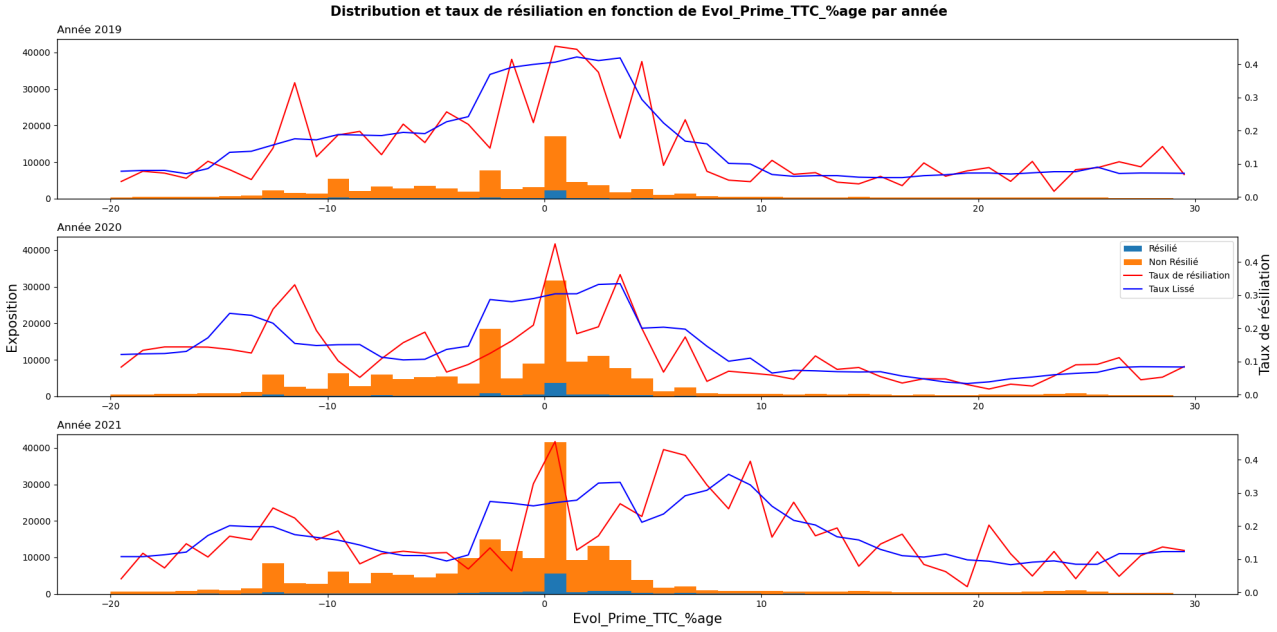


FIGURE 10.4 – Représentation du taux de résiliation en fonction de la variable *Evol_PRIME_TTC_%age*

10.2.2 Variables liées au contrat

Après nous être intéressés aux variables liées à la prime, nous allons nous concentrer sur des variables liées au contrat d'assurance.

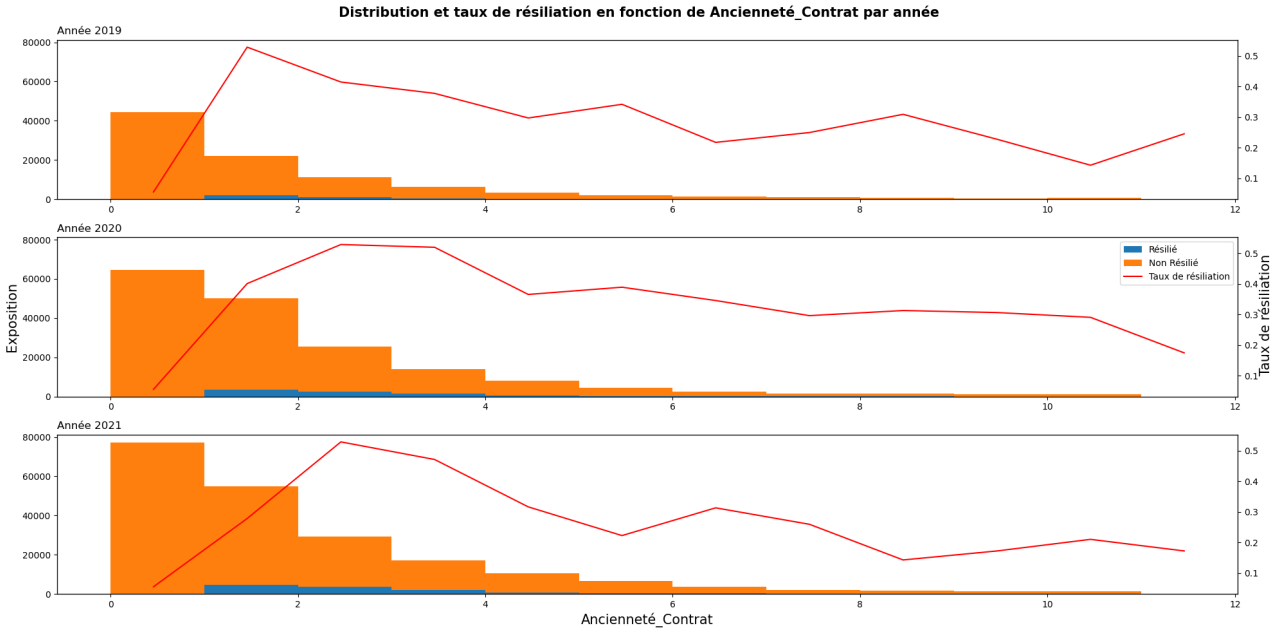


FIGURE 10.5 – Représentation du taux de résiliation en fonction de la variable *Ancienneté_Contrat*

Le graphique 10.5 montre l'évolution du taux de résiliation en fonction de l'âge du contrat. Tout d'abord, nous observons l'absence de résiliation dans la première année, ce qui est logique puisque les conditions permettant de résilier moins d'un an après avoir souscrit à l'assurance ont été exclues de la base de données. Nous pouvons aussi remarquer que le taux de résiliation semble relativement décroissant de l'âge du contrat, après avoir atteint un pic pour les contrats de deux ans en 2019 et les contrats de trois ans pour 2020 et 2021.

Sur le graphique 10.6 on peut constater que les taux de résiliations sont nettement décroissants entre les années. Nous pouvons aussi remarquer qu'une majorité des contrats de notre périmètre d'étude sont des contrats dits *Tiers-Payant* (*RC*), alors que les contrats les moins représentés sont les contrats *Tous-Risques* (*RC* +

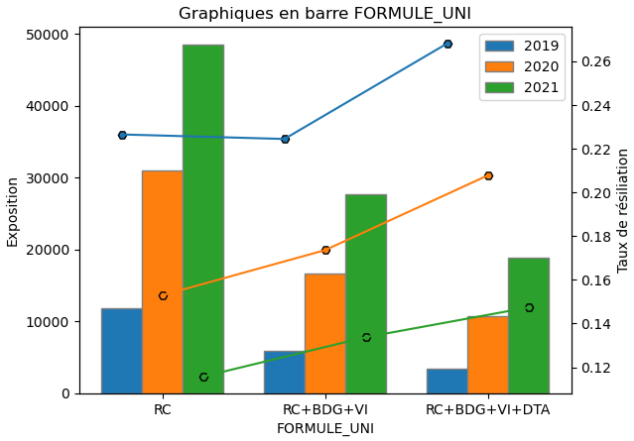


FIGURE 10.6 – Représentation du taux de résiliation en fonction de la variable *FORMULE_UNI*

$BDG + VI + DTA$). On retrouve entre ces deux catégories des contrats *hybrides*, où l'on retrouve évidemment la *Responsabilité Civile*, mais aussi des options telles que le *Bris de glace* ou la protection *Vol/Incendie*. Aussi, à l'aide de ce graphique, nous observons que les assurés *Tous-Risques* semblent plus enclins à la résiliation que les autres assurés.

10.2.3 Variables liées à l'assuré

Nous allons maintenant nous intéresser à quelques variables liées aux assurés.

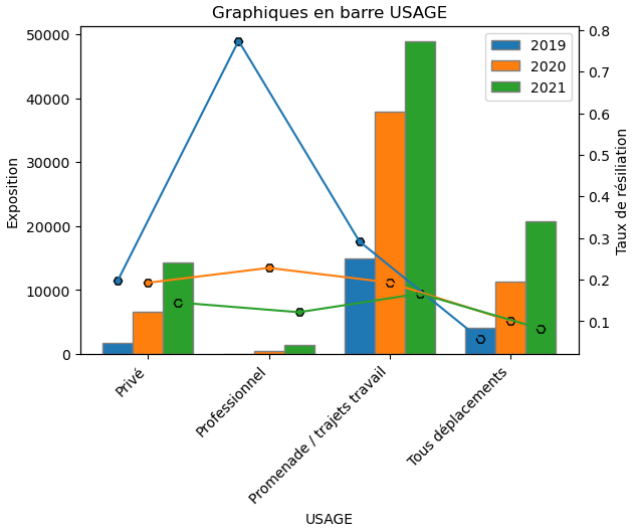


FIGURE 10.7 – Représentation du taux de résiliation en fonction de la variable *USAGE*

Le graphique 10.7 montre avant tout un déséquilibre important entre les classes pour la variable *Usage*. Nous observons néanmoins de légères différences de taux de résiliations entre les différentes classes, ces dernières sont légèrement masquées par le point atypique correspondant à la classe *Professionnel* pour l'année 2019. Le fait d'observer des dissimilatirités entre les taux de résiliations des différentes classes indique que cette variable pourrait permettre d'expliquer une part de la résiliation.

Le graphique 10.8, présente le taux de résiliation en fonction de l'âge du conducteur principal du véhicule. On observe pour ce graphique une tendance relativement similaire pour les courbes de taux de résiliation brut et lissé. Cette tendance montre notamment que les assurés de moins de 30 ans semblent plus prompt à la résiliation que les assurés plus âgés. Ce résultat est intéressant car il suggère que la variable *AGE_CP_OBS* permet d'*expliquer* une partie de la résiliation, ainsi nous pouvons imaginer des mesures de retarification ciblées sur des classes d'âge moins sensibles à la résiliation.

Le graphique 10.9 qui représente la variable *SITUATION_FAMILLE*, met en évidence des disparités vis-à-vis de la résiliation selon les différentes catégories de cette variable, en effet les tendances annuelles sont

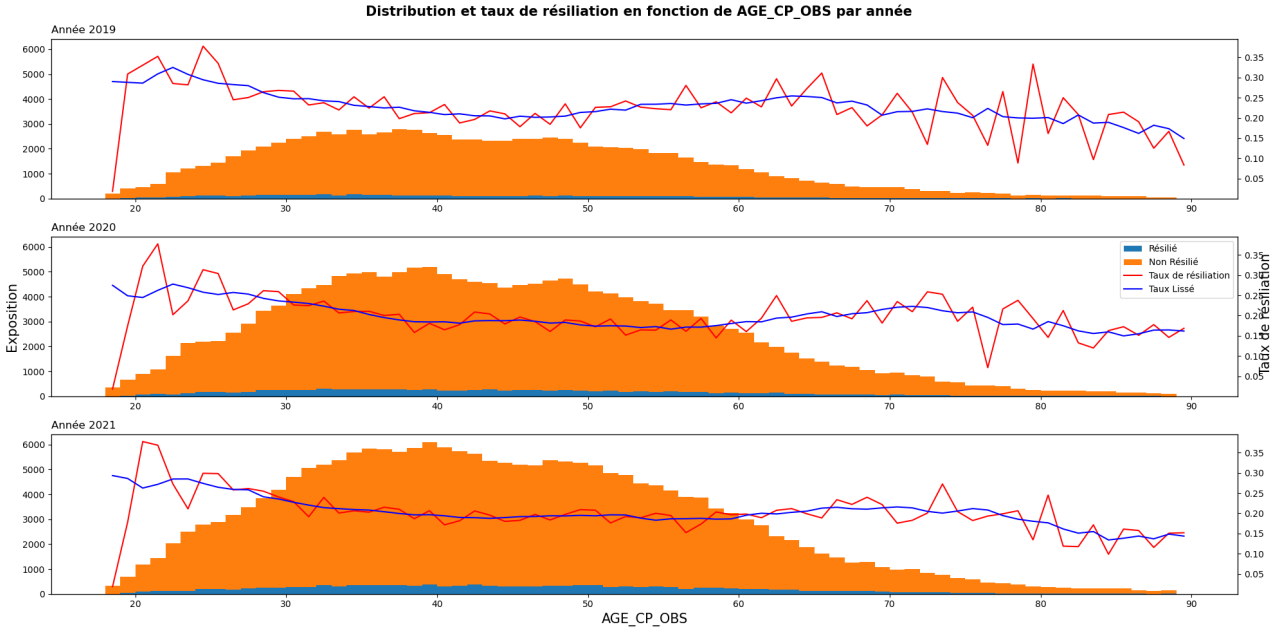


FIGURE 10.8 – Représentation du taux de résiliation en fonction de la variable *AGE_CP_OBS*

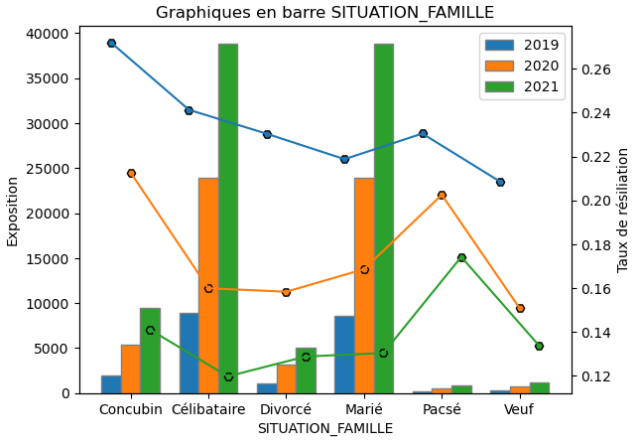


FIGURE 10.9 – Représentation du taux de résiliation en fonction de la variable *SITUATION_FAMILLE*

relativement similaires, mais le taux de résiliation des assurés est différent selon les modalités de cette variable.

Enfin, nous pouvons observer l'évolution du taux de résiliation en fonction du coefficient réduction majoration (CRM) sur le graphique 10.10, on constate que la majorité des assurés semblent avoir atteint le *Bonus 50* ce qui est logique puisque le CRM d'un assuré converge vers 0,5 par définition. Le faible taux de résiliation autour de 1 est possiblement dû au fait que les assurés de cette catégorie sont soit des jeunes conducteurs pour qui le CRM commence à 1, ou alors des anciens assurés malussés dont le CRM est revenu à 1. Ces populations n'ont pas nécessairement, ni la possibilité, ni la volonté de changer d'assurance. On observe néanmoins un pic de résiliation pour les CRM compris entre 0,9 et 0,95 ces assurés sont vraisemblablement des assurés dont le CRM a décru suite à une année sans incident et qui souhaitent voir leur prime réduire davantage que ce que propose les courtiers.

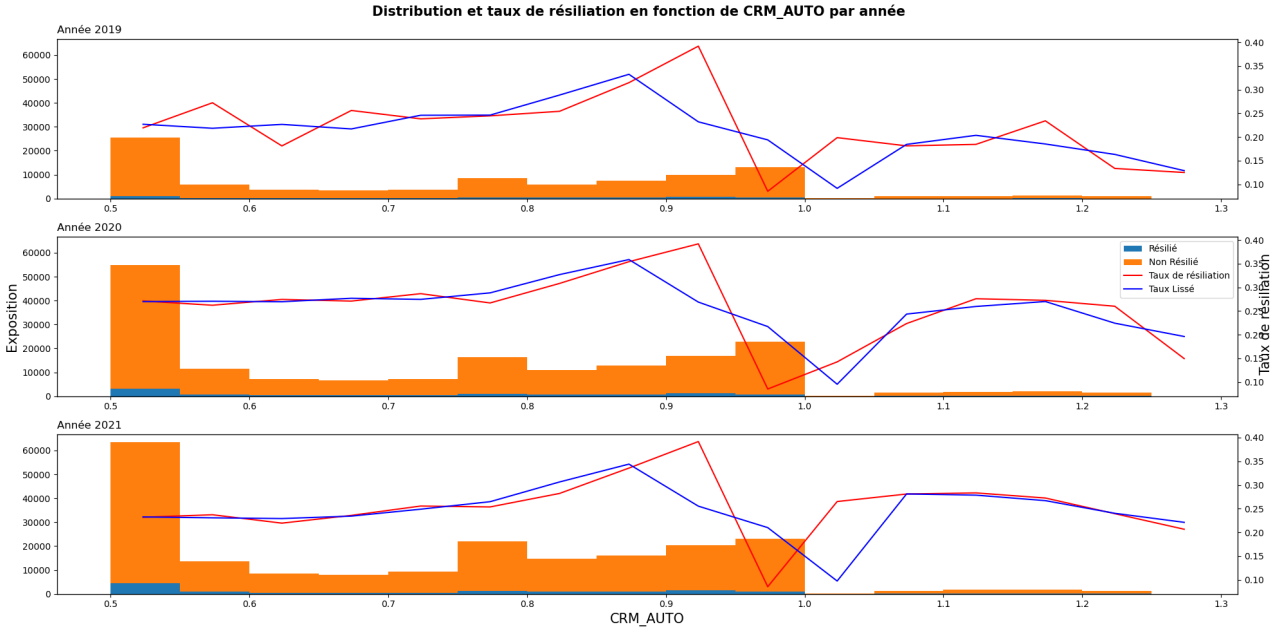


FIGURE 10.10 – Représentation du taux de résiliation en fonction de la variable *CRM_AUTO*

10.3 Analyse bivariée

Nous allons nous intéresser maintenant aux éventuelles relations entre certaines variables de notre base de données. L'objectif est ici de s'assurer que les variables que nous allons étudier ne sont pas corrélées. Il n'est en effet pas nécessaire de conserver des variables fortement corrélées entre elles car ces variables apporteraient les mêmes informations aux modèles. Nous nous concentrerons sur certaines variables pour expliciter la méthode, nous avons néanmoins étudié toutes les variables présentes dans la base pour nous assurer de la pertinence de les conserver.

Nous pouvons voir sur le graphique 10.11 qu'il n'existe pas de relation évidente entre les trois variables *PRIME_TTC*, *Evol_PRIME_TTC* et *Evol_PRIME_TTC_%age*. En effet, même si les variables d'évolution ont été construites à partir de la variable *PRIME_TTC*, nous n'observons pas de relation linéaire ou quadratique sur ces graphiques. Cela s'explique par le fait que l'évolution de prime n'est pas nécessairement liée à la prime initiale (ni a fortiori l'évolution lorsqu'elle est exprimée en pourcentage), ainsi un assuré dont la prime évolue de 100€ n'aura pas la même perception de cette augmentation selon si sa prime initiale est de 250€ ou de 1000€. C'est aussi la raison pour laquelle nous avons décidé de représenter les évolutions de primes en valeur absolue et en pourcentage, ainsi nous pourrons éventuellement observer une différence de comportement selon si l'augmentation de 100€ représente une augmentation de 40% ou une augmentation de 10%.

Le graphique 10.12 représente différentes variables liées à l'âge (du contrat, du conducteur, du permis et du véhicule). A priori, on peut craindre que ces variables soient fortement corrélées, dans la mesure où un assuré de 50 ans pourrait avoir le même véhicule et le même contrat depuis plusieurs années ainsi ces 3 variables seraient fortement corrélées linéairement. Néanmoins, nous n'observons pas, graphiquement, de corrélation particulière entre les variables. En effet, à part les variables AGE_CP_OBS et ANCIENNETE_PERMIS_AUTO qui sont liées car il n'est pas possible d'obtenir son permis avant l'âge de 18 ans, les autres variables ne semblent pas représenter de relation ni linéaire ni quadratique.

Le V de Cramer représenté en figure 10.13 montre que les variables de la base de données sont plus ou moins corrélées entre elles. Nous observons par exemple une corrélation certaine entre les variables Zone_New_Zonier ; Zone_Old_Zonier et STATIONNEMENT_DEP_corrige, il n'est donc pas nécessaire de conserver ces trois variables et seule la variable Zone_New_Zonier sera conservée pour les modélisations. De même les variables LIBELLE_CSD et NB_COND sont fortement corrélées au distributeur de l'assurance, ainsi nous ne les conserverons pas.

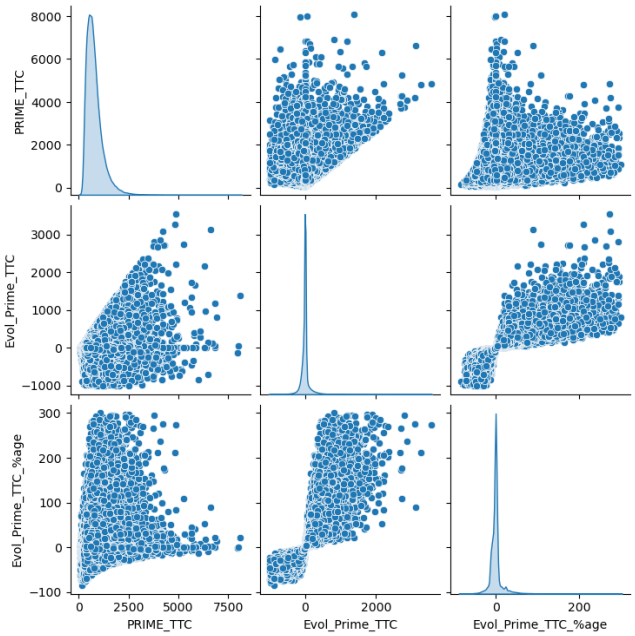


FIGURE 10.11 – Représentation croisée des variables liées aux primes

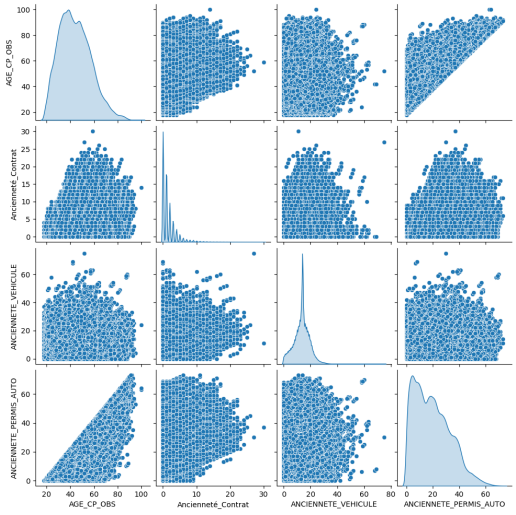


FIGURE 10.12 – Représentation croisée de variables liées à l'âge

FIGURE 10.13 – V de Cramer pour les variables qualitatives

10.4 Création des bases de modélisation

L'étude de la base de données nous permet donc de créer une base par périmètre, **standard** et **malussé**. Ces bases sont construites avec le même fondement, néanmoins le jeu de données lié au périmètre malussé dispose de variables particulières liées, par exemple, aux raisons de l'aggravation du risque de l'assuré.

Nous avons ensuite séparé ces bases en bases d'entraînement et de validation pour chaque périmètre. Nous pourrons ainsi construire et valider tous nos modèles sur la même base de données.

Quatrième partie

Étude de la résiliation

Chapitre 11

La modélisation logistique

Dans cette partie, nous allons mettre en pratique les différents modèles que nous avons développés dans la partie théorique. Pour chaque modèle, nous nous intéresserons séparément aux produits *standards* et *malussés*, cela nous permettra de comparer ces modèles et de discuter de leurs différences. En ce qui concerne les modèles, nous verrons d'abord des modèles linéaires généralisés (GLM), qui sont les modèles classiques de l'actuariat, puis nous verrons des méthodes de *Machine Learning*, en commençant par les forêts aléatoires, enfin nous utiliserons une méthode plus innovante bien que de plus en plus répandue aujourd'hui avec un algorithme de descente de gradient, en particulier nous utiliserons l'algorithme CatBoost.

11.1 Partitionnement des variables préalables à la modélisation GLM

À l'aide des analyses univariées et bivariées, nous nous sommes permis d'éliminer quelques variables, il peut y avoir plusieurs motifs pour cela, par exemple des variables qui sont trop corrélées aux autres, et qui donc n'apportent pas d'informations supplémentaires, ou alors des variables qui ne sont pas assez renseignées pour être utilisées.

Pour les variables quantitatives qui restent, il est généralement recommandé de les découper en classes pour obtenir une meilleure segmentation des résultats. Les variables ayant ainsi été découpées sont les suivantes :

- PRIME_TTC
- Evol_Prime_TTC
- Evol_Prime_TTC_%age
- Ancienneté_Contrat
- AGE_CP_OBS
- ANCIENNETE_PERMIS_AUTO
- CRM_AUTO
- ANCIENNETE_VEHICULE
- CYLINDREE_VEH

Aussi certaines variables qualitatives, possédant un grand nombre de modalités ont été regroupées en classes pour réduire le nombre de modalités de ces variables dans les régressions logistiques. Un autre objectif était de ne pas avoir de modalités trop peu représentées dans la base, ces dernières étant difficiles à modéliser. Les variables ainsi concernées sont :

- CLASSE_SRA
- GROUPE_SRA
- CARROSSERIE_VEH
- CSP_CP

11.1.1 Variables quantitatives

Pour découper les variables quantitatives en classes homogènes par rapport à la résiliation, nous avons fait le choix d'utiliser la méthode des *k-moyennes*. Pour choisir le nombre optimal de classes, nous avons utilisé la méthode du coude détaillée précédemment. Nous allons détailler notre procédé pour déterminer le nombre de classes pour une variable en particulier le raisonnement étant similaire pour les autres variables continues.

Concentrons nous donc sur la variable « *PRIME_ TTC* ». L'objectif est de créer des sous-groupes homogènes par rapport à la résiliation pour cette variable. Nous traçons alors le score de distorsion [1] en fonction du nombre de classes sur la figure 11.1 pour cette variable.

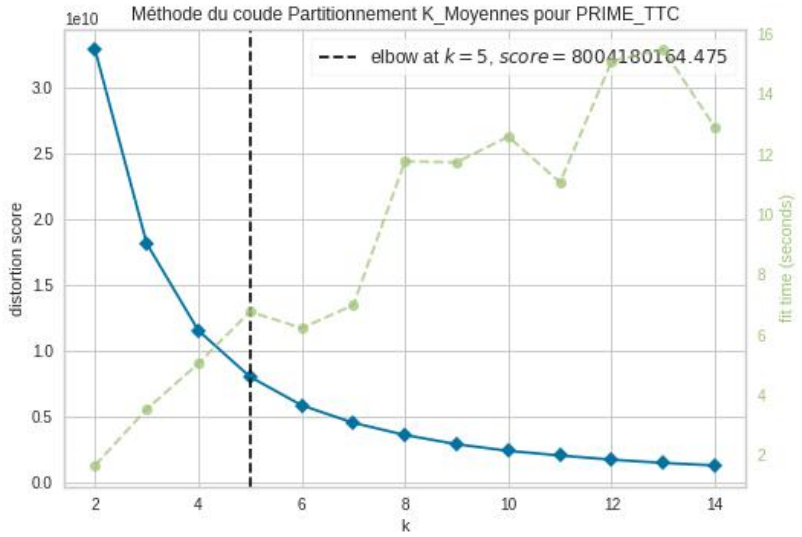


FIGURE 11.1 – Graphique du coude pour PRIME_TTC

Nous observons sur la figure 11.1 que le choix du k optimal, bien que suggéré, n'est pas évident et reste sujet au choix de l'utilisateur. Il peut donc être intéressant de tester plusieurs nombres de classes différents et de comparer les résultats, l'objectif étant de capter le plus d'information possible dans le modèle et donc de ne pas segmenter trop grossièrement les variables. Pour cette raison nous avons découpé certaines variables une première fois pour entraîner un modèle avec ce découpage. Toutefois, quelques modalités de ces variables n'étaient pas significatives, ainsi nous avons réduit le nombre de classes pour ces variables pour obtenir un nombre *convenable* de modalités significatives.

Les autres graphiques sont disponibles en annexe A.

11.1.2 Variables qualitatives

Pour partitionner les variables qualitatives par rapport à la résiliation, nous avons fait le choix de réaliser une classification ascendante hiérarchique. Nous avons utilisé la *distance de Ward* comme mesure de la dissimilitude entre les différentes modalités. Pour choisir le nombre *optimal* de classes, nous avons, ici aussi, utilisé la méthode du coude. Prenons l'exemple de la classification de la variable *CLASSE_ SRA* présenté en figure 11.2, le raisonnement est analogue pour les autres variables, et les graphiques sont disponibles en annexe A.

Sur ce graphique aussi, nous pouvons choisir différents seuils *convenables*. En effet, on observe nettement plusieurs coudes, par exemple pour $k = 6$, $k = 11$ ou $k = 15$, il est donc intéressant de tester plusieurs nombres de classes pour optimiser au mieux le résultat.

1. La distance des points d'un clusteur au centroid de ce dernier, on peut rapprocher cela de la variance intra-classe

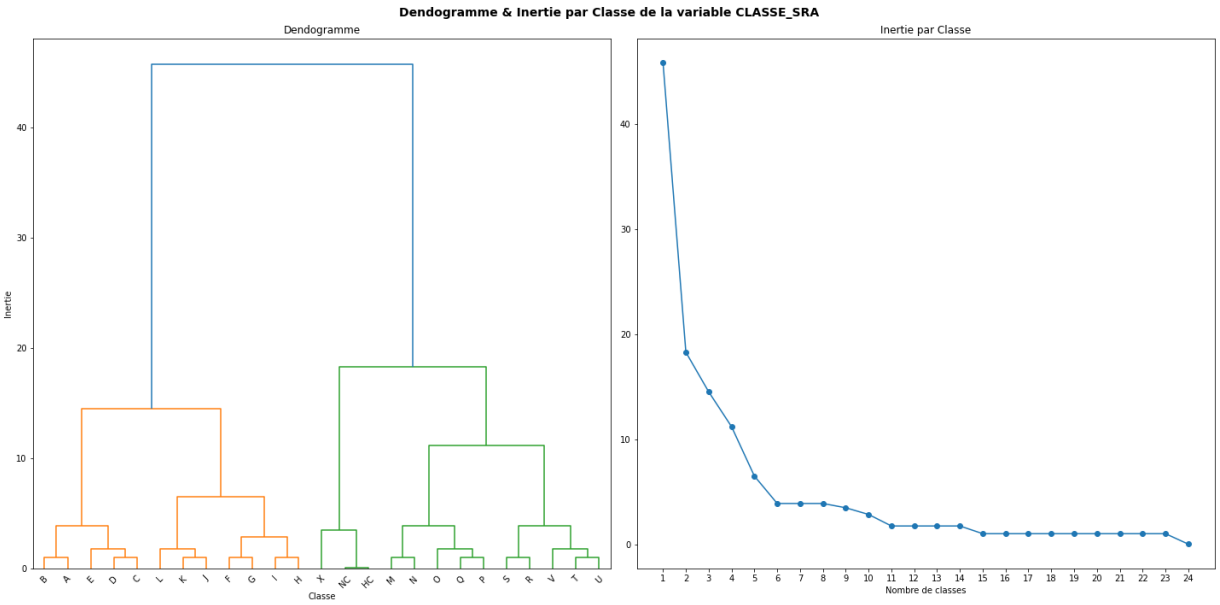


FIGURE 11.2 – Classification Ascendante Hiérarchique pour la variable CLASSE_SRA

11.1.3 Analyse des corrélations des variables

Dans le cadre de la modélisation, le fait d'avoir des variables parfaitement corrélées n'est pas pertinent, en effet, si une variable est parfaitement corrélée à une autre alors elle n'apportera pas d'information supplémentaires. C'est pourquoi nous avons réalisé une analyse des corrélations, à l'aide de la méthode du V de Cramer (car après le partitionnement, toutes nos variables sont qualitatives), pour observer d'éventuelles corrélations entre les variables, notamment avec celles que nous avons modifiées.

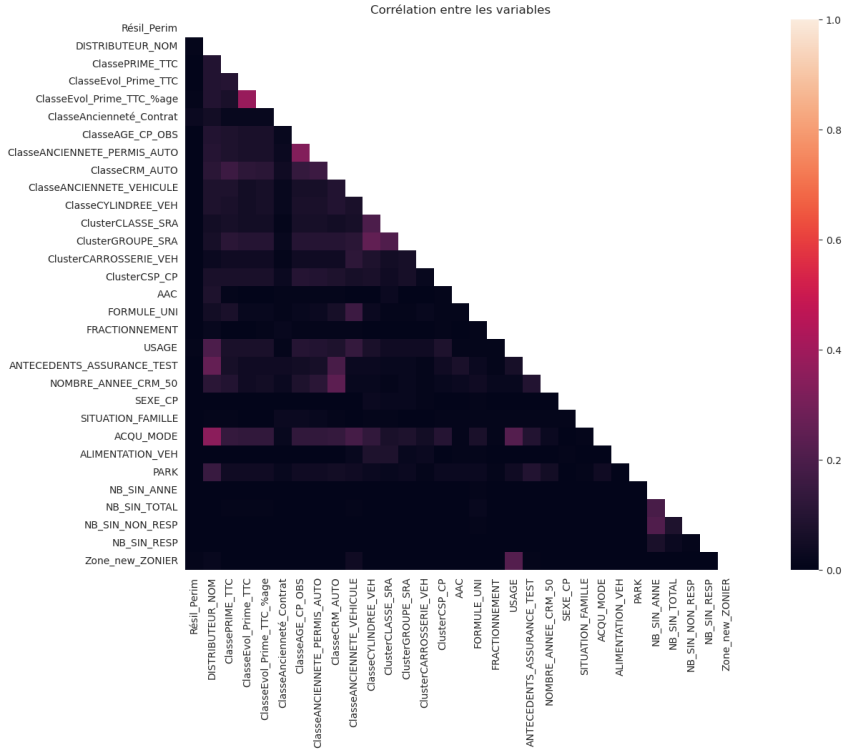


FIGURE 11.3 – V de Cramer pour les variables candidates aux GLM

Nous observons sur le graphique 11.3 que nos variables ne sont pas particulièrement corrélées, nous pouvons ainsi les utiliser dans la modélisation par régression logistique de la résiliation.

11.2 Modèle standard

Nous allons dans cette section, modéliser la variable résiliation à l'aide d'une régression logistique pour le périmètre standard. Pour pouvoir évaluer la qualité des modèles, nous avons préalablement scindé la base en deux, une base d'entraînement représentant 80 % de la base initiale et une base de validation contenant les 20 % restants. Dans cette section, nous allons détailler la méthodologie de travail employée pour obtenir, *in fine*, le meilleur modèle possible pour le périmètre standard.

11.2.1 Premier modèle, premier écueil

Pour commencer, nous avons construit une régression logistique avec toutes les variables candidates et en conservant la proportion de résiliation initiale de la base de données. Cette proportion étant particulièrement faible, approximativement $5,55$ % des lignes de la base de données représentent une résiliation de l'assuré, le modèle obtenu n'est pas très bon au regard des métriques choisies (score F_2 et justesse pondérée ou BA pour Balanced Accuracy). Nous avons donc fait le choix de rééchantillonner la base de données dans l'optique d'avoir une proportion de lignes associées à une résiliation « *satisfaisante* ».

Un autre paramètre doit être réglé, il s'agit du seuil de probabilité à partir duquel on considérera qu'un individu est résilié. En effet, la régression logistique accordera à chaque observation la probabilité que la variable réponse soit égale à 1, pour convertir cette probabilité en Non résilié (0), ou Résilié (1), et ainsi remplir la matrice de confusion, il faut choisir un seuil à partir duquel on considérera que l'individu va résilier son contrat. Ce choix est important et est très lié à la problématique, en prenant un taux inférieur à $0,5$, on augmente le risque de mal classer certains assurés qui n'ont pas résilié leur contrat, néanmoins, on va aussi augmenter la part d'assurés ayant résilié bien prédite. Ainsi, selon si l'on souhaite plutôt un modèle très précis, ou un modèle qui capte bien la variable d'intérêt, il faudra faire varier le seuil de probabilité.

Nous pouvons comparer les résultats de ce modèle selon les différents seuils de probabilité à l'aide du 11.1 et du graphique 11.4.

seuil_proba	F2	BA
0.1	0.3568150138	0.53861945
0.2	0.1528585411	0.13625866
0.3	0.0497777221	0.04080062
0.4	0.0162503186	0.01308699
0.5	0.001601948	0.00128304
0.6	0.0003206567	0.00025661
0.7	NA	NA
0.8	NA	NA
0.9	NA	NA
1	NA	NA

TABLEAU 11.1 – Tableau du score F_2 et de la justesse pondérée en fonction du seuil de probabilité

On observe que le seuil de probabilité offrant les meilleurs résultats est le seuil de 0,1. Le score F_2 est alors de 0,36 et la justesse pondérée est de 0,54. Toutefois, dans la mesure où le meilleur seuil est très faible et que les résultats chutent drastiquement dès qu'il est augmenté, nous avons choisi d'augmenter la proportion de résiliations dans la base de modélisation pour essayer d'améliorer ce modèle.

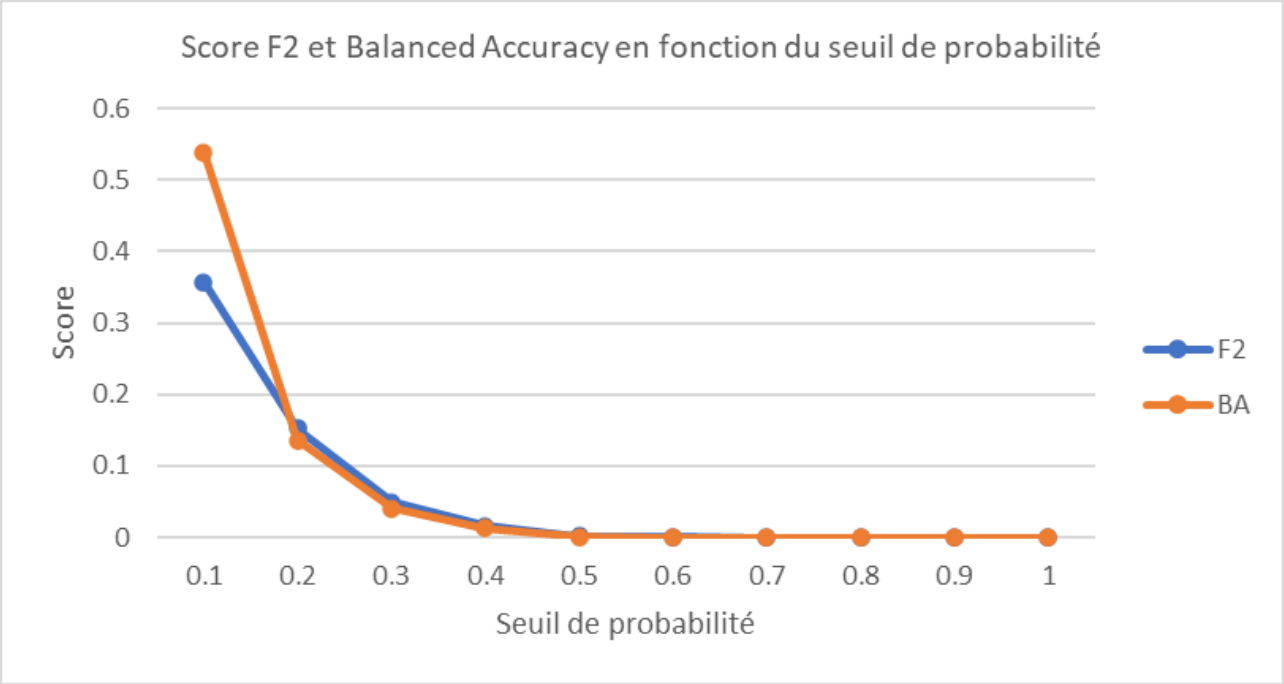


FIGURE 11.4 – Score F2 et BA en fonction de différents seuils de probabilité, pour le périmètre standard

11.2.2 Rééchantillonnage

Pour choisir le taux de rééchantillonnage, c'est-à-dire la part de résiliation dans la base de modélisation, *optimale*, nous avons évalué, pour chaque méthode (SMOTE, ROSE, OVUN) des modèles de régression à l'aide de différentes métriques. Nous avons ajusté ces modèles sur des bases rééchantillonnées à différents taux. L'évaluation des modèles a été faite sur la même base de validation non rééchantillonnée. Pour permettre de choisir un seuil de probabilité pertinent, nous avons décidé d'évaluer les modèles à différents seuils, à savoir 0,3 ; 0,5 ; 0,7. Nous détaillerons, dans un premier temps, la mise en pratique des différentes méthodes de rééchantillonnage. Ensuite, nous pourrons comparer les résultats de ces méthodes vis-à-vis de notre modélisation.

11.2.2.1 Création des bases de données rééchantillonnées

Pour créer ces bases d'entraînements, nous avons utilisé deux langages de programmation différents, *python* et *R*, nous allons détailler, ici, les bibliothèques utilisées ainsi que différents paramètres que nous avons utilisés.

1. **SMOTE** :
   - Bibliothèque : imblearn sur *Python* ;
   - Fonction : SMOTEN, cette fonction est adaptée aux données qualitatives ;
   - Paramètres :
     - sampling_strategy : nombre réel correspondant au ratio désiré entre le nombre d'élément de la classe minoritaire et le nombre d'éléments de la classe majoritaire, ainsi pour obtenir un ratio de 50% d'individus de chaque classe dans la base finale, il faut établir sampling_strategy à 1 [2] ;
     - k_neighbors = 4 : nombre de plus proches voisins utilisés pour construire le voisinage des individus
2. **ROSE** :
   - Bibliothèque : ROSE sur *R*
   - Fonction : ROSE
   - Paramètres :
     - p : taux voulu pour la classe minoritaire après rééchantillonnage
3. **OVUN** :
   - Bibliothèque : ROSE sur *R*

2. De façon générale, si l'on désire un ratio y d'éléments de la classe minoritaire dans la base finale, il faut établir sampling_strategy à $\frac{-y}{y-1}$

- Fonction : ovun.sample
- Paramètres :
  - p : taux voulu pour la classe minoritaire après rééchantillonnage ;
  - method = both : ainsi la méthode combine sur- et sous-échantillonnage.

En ce qui concerne le taux de rééchantillonnage, nous avons décidé de faire neuf nouvelles bases correspondant aux taux suivants : $0,10$; $0,15$; $0,20$; $0,25$; $0,30$; $0,35$; $0,40$; $0,45$; $0,50$, pour pouvoir comparer les modèles entraînés selon chaque taux.

11.2.2.2 Comparaison des méthodes de rééchantillonnage

Nous allons d'abord comparer à l'aide de graphiques les différentes méthodes pour chaque seuil de probabilité. Ainsi, nous pourrons choisir quelle méthode nous semble la plus adaptée. Une fois la méthode choisie, nous pourrons comparer les différents seuils et élire le plus pertinent. Ces résultats sont détaillés sous la forme d'un tableau en annexe B.

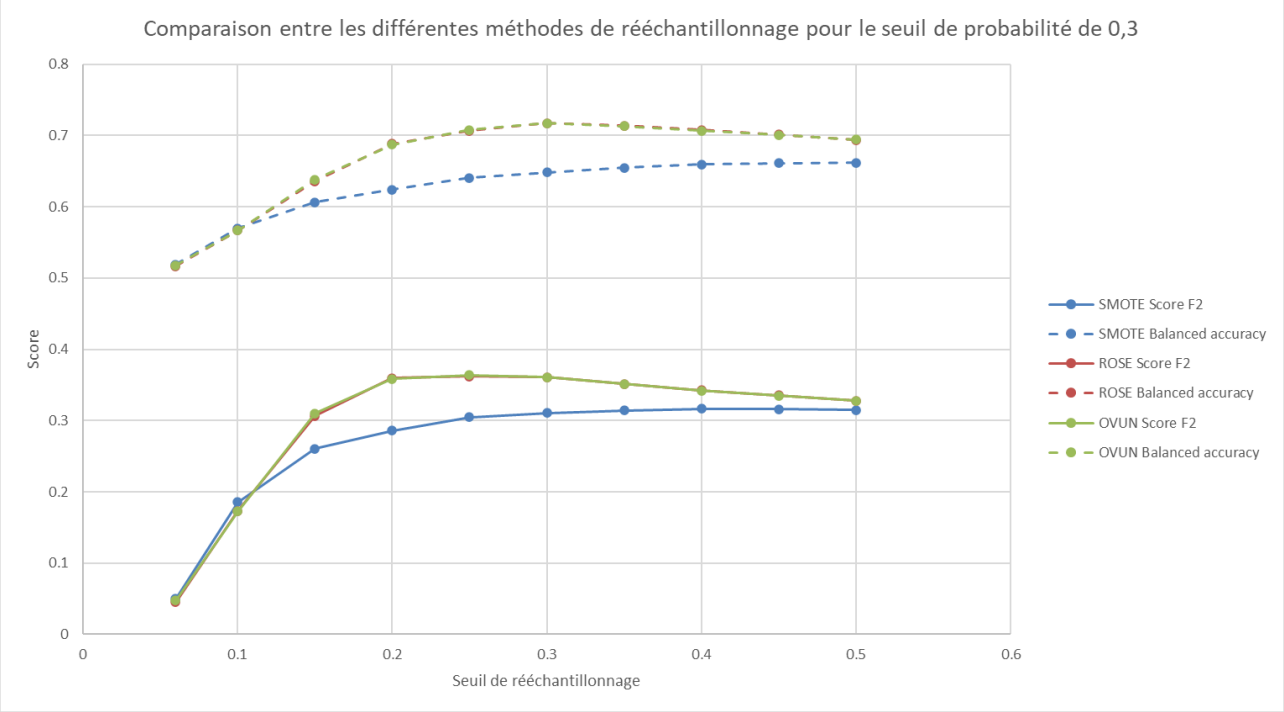


FIGURE 11.5 – Évolution du score F_2 et de la justesse pondérée en fonction du seuil de rééchantillonnage pour chaque méthode, pour le seuil de probabilité de $0,3$

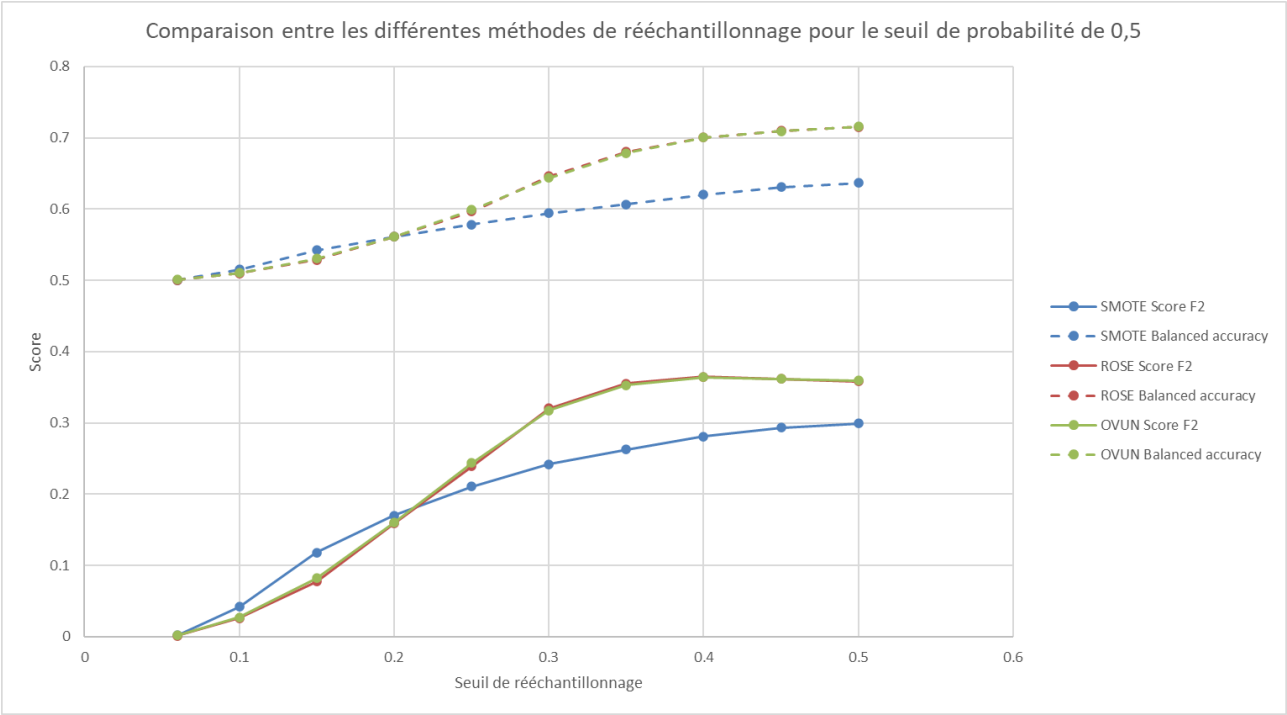


FIGURE 11.6 – Évolution du score F_2 et de la justesse pondérée en fonction du seuil de rééchantillonnage pour chaque méthode, pour le seuil de probabilité de $0,5$

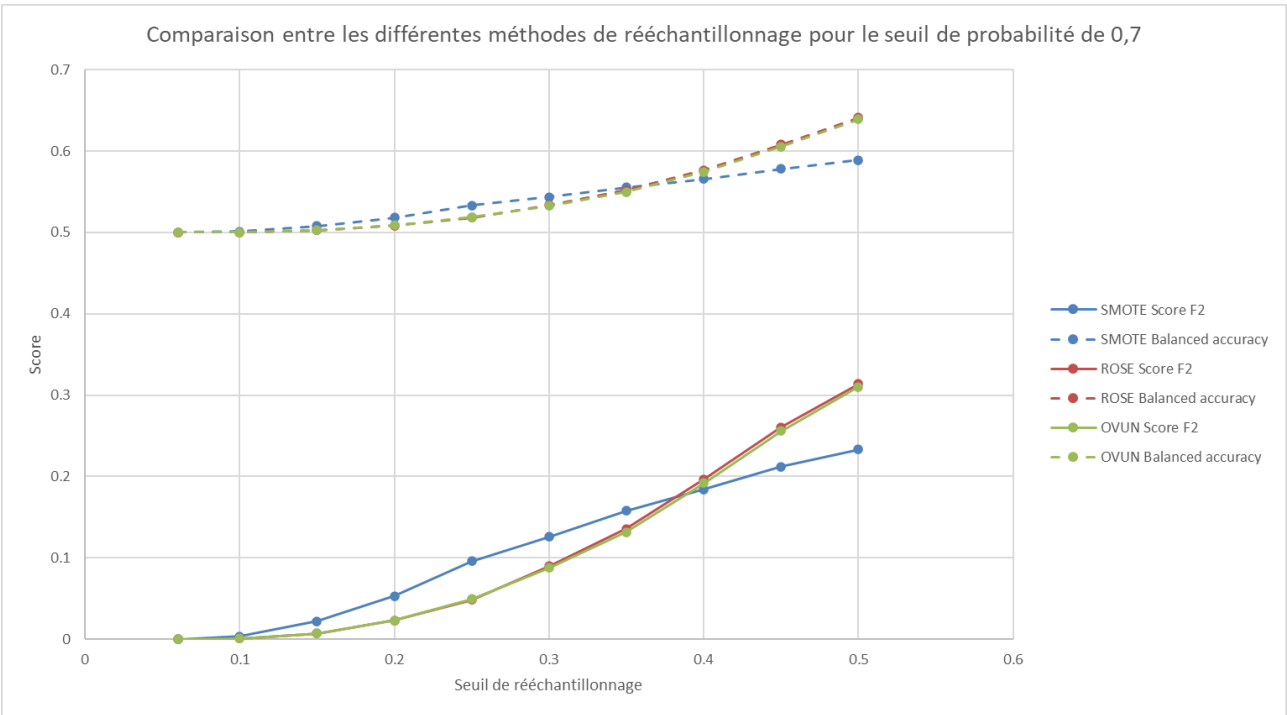


FIGURE 11.7 – Évolution du score F_2 et de la justesse pondérée en fonction du seuil de rééchantillonnage pour chaque méthode, pour le seuil de probabilité de $0,7$

Les graphiques 11.5, 11.6 et 11.7 représentent le score F_2 (les courbes pleines) et la justesse pondérée (les courbes en tiret) des modèles, pour chacune des méthodes (SMOTE en bleu, ROSE en rouge et OVUN en vert) que nous avons construits et ayant été évalué sur notre base de test non-rééchantillonnée. Il semblerait que les méthodes ROSE et OVUN obtiennent des résultats très proches, en effet les courbes de score pour la méthode ROSE sont "cachées" par les courbes de la méthode OVUN. De plus, les résultats de ces méthodes sont, dans cette étude, légèrement au-dessus des résultats de la méthode SMOTE. Le choix entre les méthodes ROSE

et OVUN est complexe car les résultats sont très proches pour ces deux méthodes. Nous nous concentrerons cependant sur la méthode ROSE dans la suite de l'étude.

Nous allons donc maintenant utiliser les résultats que nous avons obtenus pour la méthode ROSE pour choisir le seuil de probabilité adéquat, celui-ci dépend évidemment du taux de rééchantillonnage, on les choisira donc ensemble.

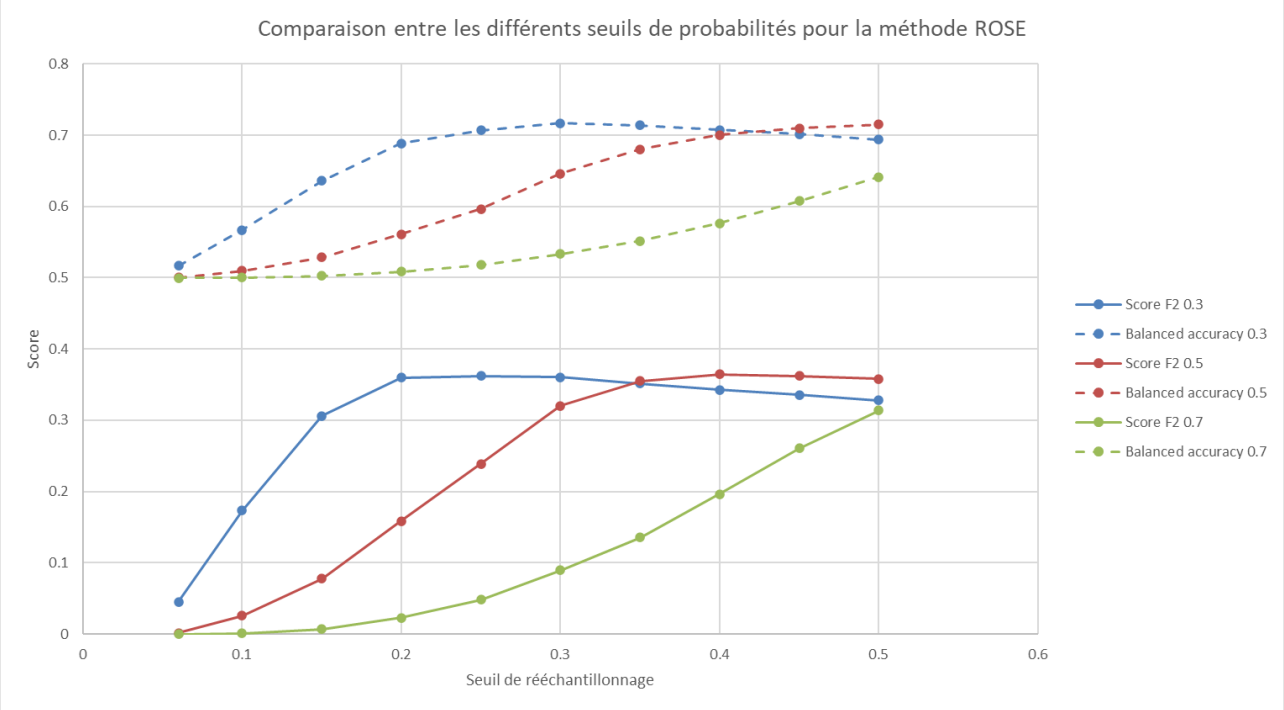


FIGURE 11.8 – Évolution des scores F_2 et BA en fonction du seuil de rééchantillonnage selon les différents taux de rééchantillonnage pour la méthode ROSE

Le graphique 11.8 représente le score F_2 (les courbes pleines) et la justesse pondérée (les courbes en tiret) des modèles, pour chacun des seuils de probabilité (0,3 en bleu, 0,5 en rouge et 0,7 en vert). Une fois de plus, les choix du taux rééchantillonnage et du seuil de probabilité ne sont pas évidents, en effet le maximum du score-F_2 ne correspond pas nécessairement au maximum de la BA. Néanmoins, on peut estimer que le choix de prendre un taux de rééchantillonnage de 0,5, et un seuil de probabilité de 0,5 semble être un choix correct. En effet, même si les scores ne sont pas maximisés pour ces paramètres, ils restent à un bon niveau, et l'on ne sacrifie pas trop l'une des deux métriques par rapport à l'autre, puisque ces dernières sont très proches de leurs maxima. Néanmoins, selon la problématique de l'assureur il peut être intéressant de faire varier ce taux selon l'objectif visé, ainsi si l'on cherche à ne cibler que des gens dont le risque de résiliation est important, il faudra prend un seuil de probabilité élevé pour avoir une grande précision, a contrario, si l'on cherche à éviter le plus de résiliation possible (en acceptant de se tromper en ayant un taux de faux négatif important) on aura tendance à prendre un seuil de probabilité plutôt faible.

Pour la suite de la modélisation GLM sur le périmètre standard, nous allons donc utiliser la base de données rééchantillonnée de façon à avoir 50 % d'individus de la classe minoritaire et 50 % de la classe majoritaire, à l'aide de la méthode ROSE.

11.2.3 Deuxième modèle

Nous avons donc désormais un deuxième modèle qui semble plus efficace que le premier. En effet, le fait de rééchantillonner notre base de données afin d'avoir une plus grande proportion de résiliation dans la base de modélisation a permis d'obtenir de meilleurs résultats en termes de score F_2 et de BA que pour le premier modèle.

Comme nous pouvons le voir sur le tableau 11.2, la base initiale, où il n'y a que 5 % de lignes correspondantes à des résiliations a de moins bon résultats, particulièrement en termes de justesse pondérée. Les scores F_2 sont néanmoins presque égaux.

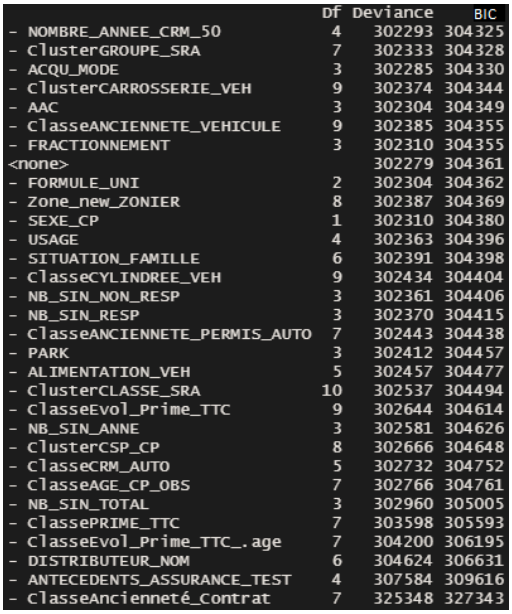
Maintenant que nous avons obtenu un modèle saturé relativement qualitatif, il est important de réduire sa dimension, c'est-à-dire d'utiliser des critères qui permettront de réduire le nombre de variables du modèle.

	Base initiale	Base rééchantillonnée
Seuil de proba	0.1	0.5
F2-Score	0.36	0.36
BA	0.54	0.71

TABLEAU 11.2 – Résultats sur la base de test des GLMs sur le périmètre standard

11.2.4 Réduction de la dimension du modèle

Comme notre modèle a initialement une grande dimension, nous utiliserons le critère BIC qui pénalise mieux ces modèles que le critère AIC. Les résultats de la première étape de cet algorithme sont présentés à l'aide de la sortie R en figure 11.9 :



	Df	Deviance	BIC
- NOMBRE_ANNEE_CRM_50	4	302293	304325
- ClusterGROUPE_SRA	7	302333	304328
- ACQU_MODE	3	302285	304330
- ClusterCARROSSERIE_VEH	9	302374	304344
- AAC	3	302304	304349
- ClasseANCIENNETE_VEHICULE	9	302385	304355
- FRACTIONNEMENT	3	302310	304355
<none>		302279	304361
- FORMULE_UNI	2	302304	304362
- Zone_new_ZONIER	8	302387	304369
- SEXE_CP	1	302310	304380
- USAGE	4	302363	304396
- SITUATION_FAMILLE	6	302391	304398
- ClasseCYLINDREE_VEH	9	302434	304404
- NB_SIN_NON_RESP	3	302361	304406
- NB_SIN_RESP	3	302370	304415
- ClasseANCIENNETE_PERMIS_AUTO	7	302443	304438
- PARK	3	302412	304457
- ALIMENTATION_VEH	5	302457	304477
- ClusterCLASSE_SRA	10	302537	304494
- ClasseEvol_Prime_TTC	9	302644	304614
- NB_SIN_ANNE	3	302581	304626
- ClusterCSP_CP	8	302666	304648
- ClasseCRM_AUTO	5	302732	304752
- ClasseAGE_CP_OBS	7	302766	304761
- NB_SIN_TOTAL	3	302960	305005
- ClassePRIME_TTC	7	303598	305593
- ClasseEvol_Prime_TTC_.age	7	304200	306195
- DISTRIBUTEUR_NOM	6	304624	306631
- ANTECEDENTS_ASSURANCE_TEST	4	307584	309616
- ClasseAncienneté_Contrat	7	325348	327343

FIGURE 11.9 – Premier pas de l'algorithme de réduction du nombre de variables

On observe, à chaque ligne, la déviance et le BIC du modèle entraîné lorsque l'on enlève la variable en début de ligne. La ligne <none> correspond au modèle si l'on enlève aucune variable. Par exemple, après cette première étape, nous pouvons éliminer la variable *NOMBRE_ANNEE_CRM_50* car la déviance et le BIC du modèle s'en voient améliorés.

Cette méthode nous a permis de réduire l'AIC et le BIC du modèle, ainsi que le nombre de variables du modèle. Néanmoins, la déviance a légèrement augmenté, tandis que la *Null deviance* est restée stable. Ces résultats sont résumés dans le tableau 11.3, dans lequel GLM_{STD}^{sat} représente le modèle saturé, doté de toutes les variables candidates, et GLM_{STD}^{rest} représente le modèle obtenu à la suite de l'algorithme.

Modèle	Nombre de variables	Residual Deviance	Null Deviance	AIC	BIC
GLM_{STD}^{sat}	30	302279	389492	302611	304361
GLM_{STD}^{rest}	17	302600	389491	302500	304202

TABLEAU 11.3 – Résultats sur la base de test des GLMs sur le périmètre standard

11.2.5 Importance des variables

Après avoir trouvé un GLM satisfaisant, et avoir réduit la dimension de ce dernier à l'aide du critère BIC, nous avons fait le choix d'étudier l'importance des variables à l'aide de la méthode *PFI* (*Permutation Feature Importance*) qui se base sur le fait que la modification d'une variable aura d'autant plus d'effet sur la performance du modèle que cette variable est importante. Pour créer ce graphique, nous avons considéré la perte de justesse

pondérée comme mesure de la performance des modèles. Cette étude n'est pas strictement nécessaire dans le cas des modèles GLM car ces derniers sont souvent considérés comme intrinsèquement interprétable, même s'il est complexe d'interpréter des centaines de coefficients à la fois. Nous avons toutefois décidé d'utiliser cette méthode pour plusieurs raisons. Premièrement, nous avons de nombreuses modalités, et donc un grand nombre de coefficients ainsi le graphique obtenu grâce à cette méthode permet de saisir plus rapidement les variables qui impactent le plus le résultat de notre modélisation. Deuxièmement, les prochaines méthodes auxquelles nous nous intéresserons ne disposent pas de la même simplicité d'interprétation, ainsi utiliser une méthode *agnostique* simplifie les comparaisons entre les différents résultats obtenus.

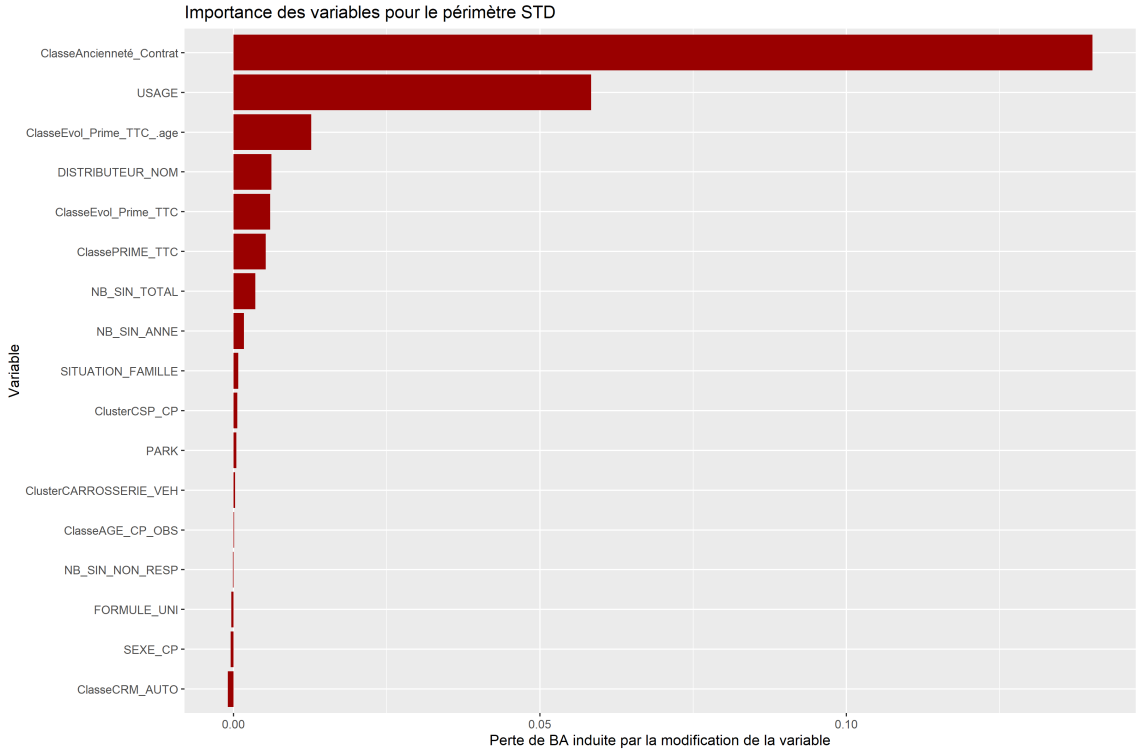


FIGURE 11.10 – Importance des variables pour le modèle GLM standard

Nous observons sur le graphique 11.10 que la variable qui impacte le plus le modèle est la variable relative à l'ancienneté du contrat. Cela est sans doute dû au fait que le périmètre de modélisation choisi exclu les résiliations pour les contrats de moins d'un an, ainsi, dans la mesure où la proportion de contrat *jeune* est importante, cela induit possiblement un biais dans la modélisation. La deuxième variable la plus importante est la variable liée à l'usage du véhicule. Nous pouvons noter que les variables liées à la prime, ou à son évolution contribuent de façon notable à la qualité de notre modèle. Cela justifie notre volonté de modéliser la résiliation des assurés par rapport à ces variables.

11.3 Modèle malussé

Dans cette section, nous allons chercher à créer une régression logistique visant à modéliser les résiliations sur le périmètre malussé. Nous allons procéder de la même façon que pour le périmètre standard, nous allons donc, dans un premier temps, créer un modèle saturé sans rééchantillonnage pour pouvoir comparer les performances de ce dernier aux modèles construits sur des bases de données rééchantillonnées et ainsi discuter de la pertinence du rééchantillonnage. Nous pourrons ensuite nous concentrer sur la réduction de la dimension du modèle puis sur une étude de l'importance des variables.

11.3.1 Premier modèle

Nous avons donc créé un modèle en considérant toutes les variables candidates et en conservant la proportion de résiliation initiale de la base de données. Sur le périmètre malussé, cette proportion est d'approximativement $6,67$ % des lignes de la base de données représentant une résiliation. Pour juger de la qualité des modèles sur ce périmètre, nous allons utiliser le score-F_2 et la justesse pondérée, définis en 8.1. Nous pouvons commencer

par comparer les performances du modèle saturé sur la base initiale selon différents seuils de probabilité. Ces résultats seront présentés dans le tableau 11.4 et du graphique 11.11.

seuil_proba	F2	BA
0.1	0.46360309	0.6612529
0.2	0.34421126	0.3462877
0.3	0.23157082	0.20881671
0.4	0.16026936	0.13805104
0.5	0.07855239	0.0649652
0.6	0.02653471	0.02146172
0.7	0.00506806	0.00406033
0.8	NA	NA
0.9	NA	NA
1	NA	NA

TABLEAU 11.4 – Tableau du score F_2 et de la justesse pondérée en fonction du seuil de probabilité, pour le périmètre malussé

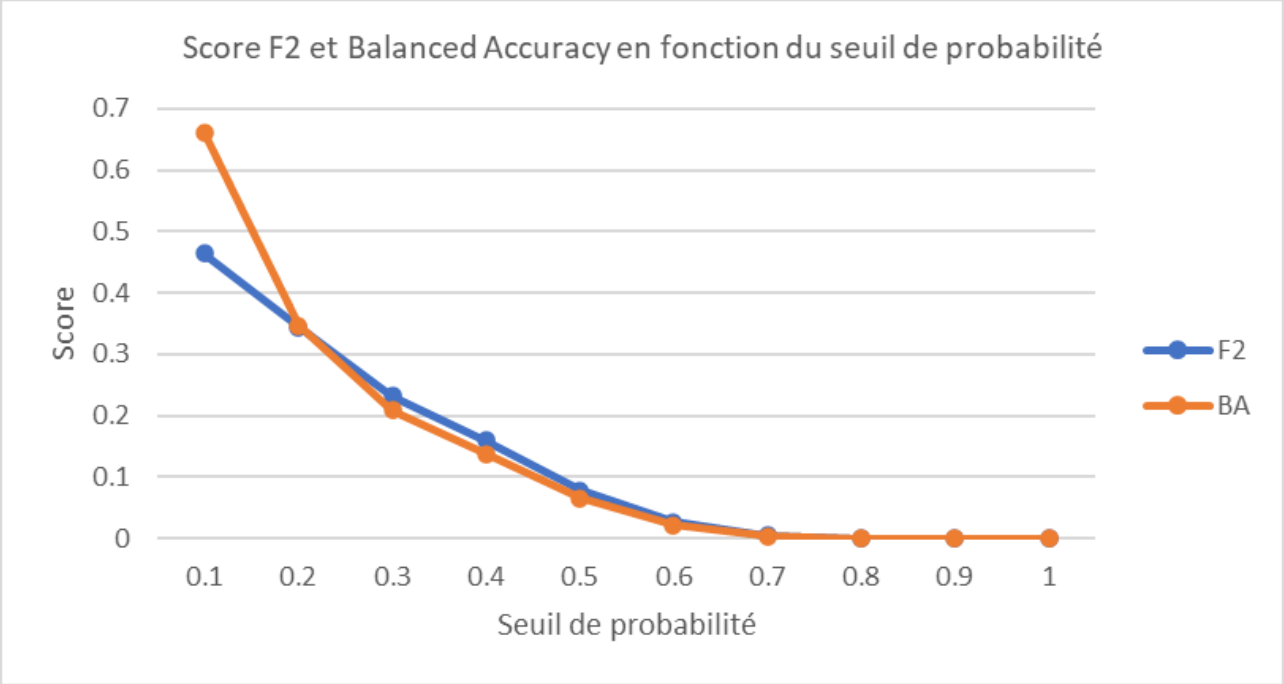


FIGURE 11.11 – Score F2 et BA en fonction de différents seuils de probabilité, pour le périmètre malussé

Pour le seuil de probabilité de 0.1, on observe de plutôt bons résultats avec un score F_2 de 0,46 et une justesse pondérée de 0,66. Il est toutefois important de vérifier dans quelle mesure ces résultats peuvent être améliorés à la suite d'un rééquilibrage des classes de la variable d'intérêt dans la base de données.

11.3.2 Rééchantillonnage

Comme pour le périmètre standard, nous avons créé un tableau permettant de comparer différentes métriques afin d'évaluer la qualité des différentes méthodes de rééchantillonnage et des différents seuils de rééchantillonnage et de probabilité. Le rééchantillonnage a été réalisé à partir des mêmes outils que pour le périmètre standard (définis ici), nous ne les redétaillerons donc pas ici.

Nous pouvons comparer à l'aide des graphiques suivants les différentes méthodes de rééchantillonnage sur le périmètre malussé. Les résultats présentés ci-après sont représentés en annexe B sous la forme d'un tableau.

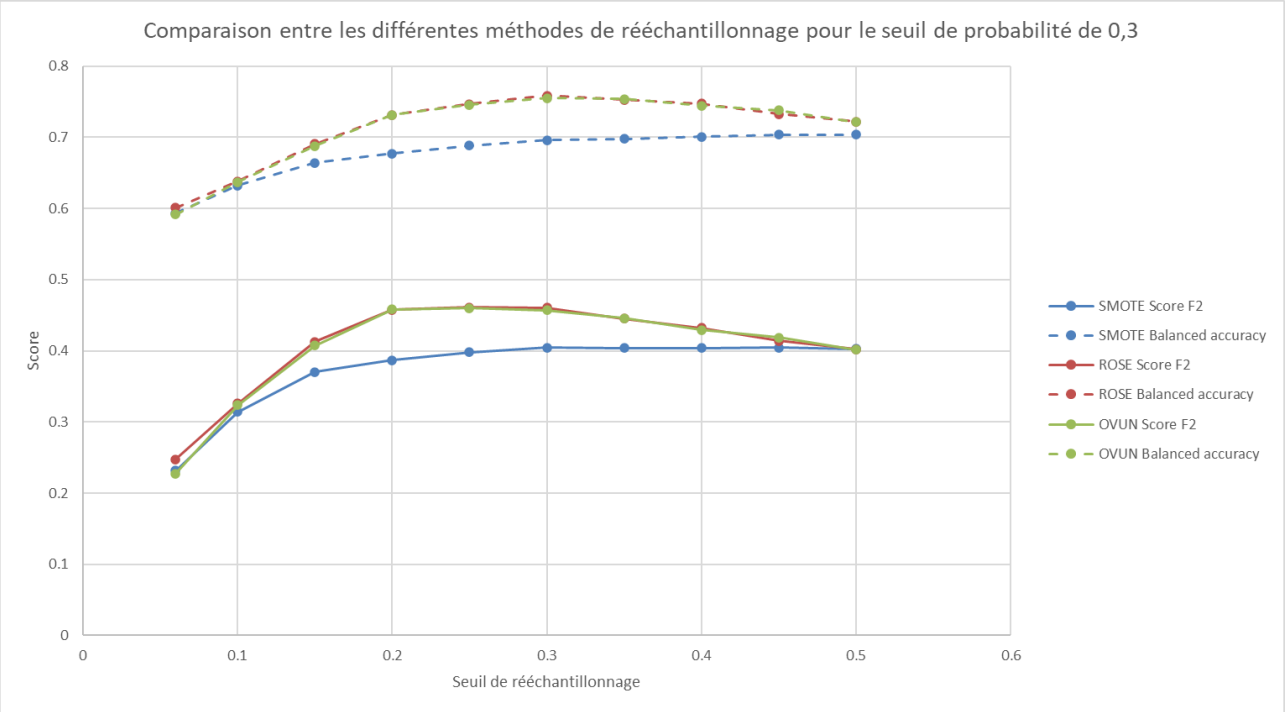


FIGURE 11.12 – Évolution du score F_2 et de la justesse pondérée en fonction du seuil de rééchantillonnage pour chaque méthode, pour le seuil de probabilité de $0,3$, sur le périmètre malussé

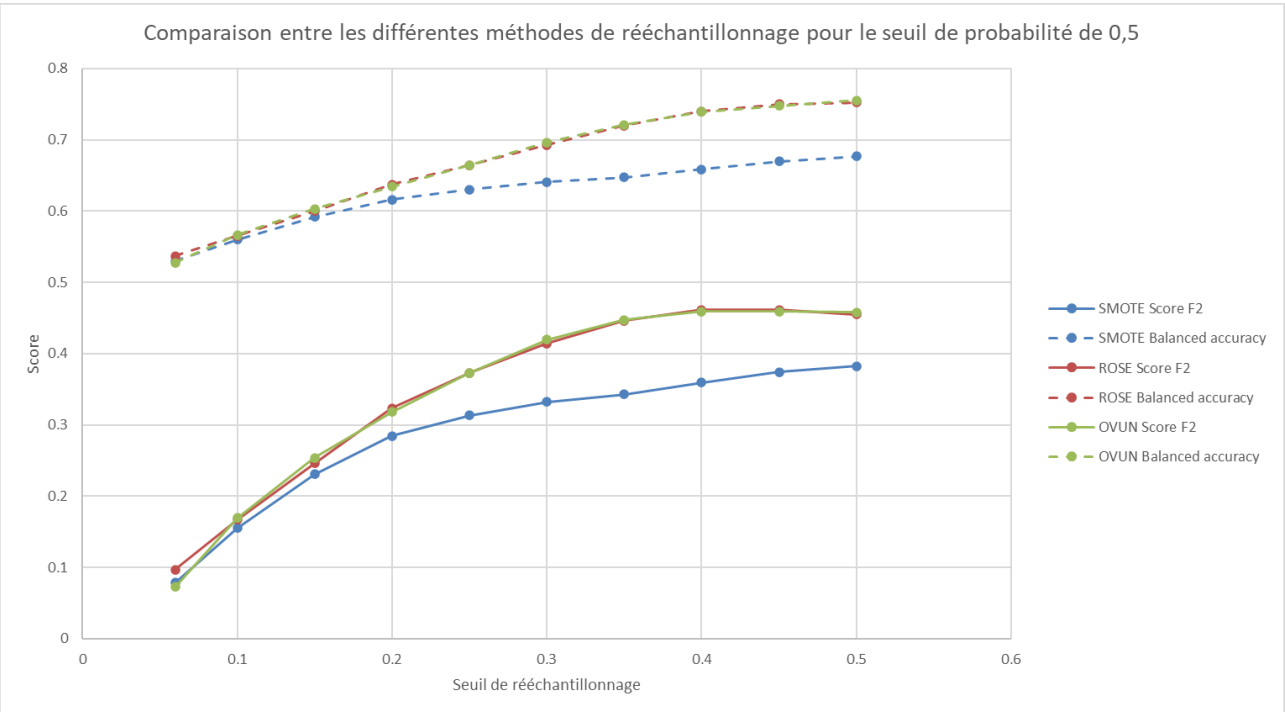


FIGURE 11.13 – Évolution du score F_2 et de la justesse pondérée en fonction du seuil de rééchantillonnage pour chaque méthode, pour le seuil de probabilité de $0,5$, sur le périmètre malussé

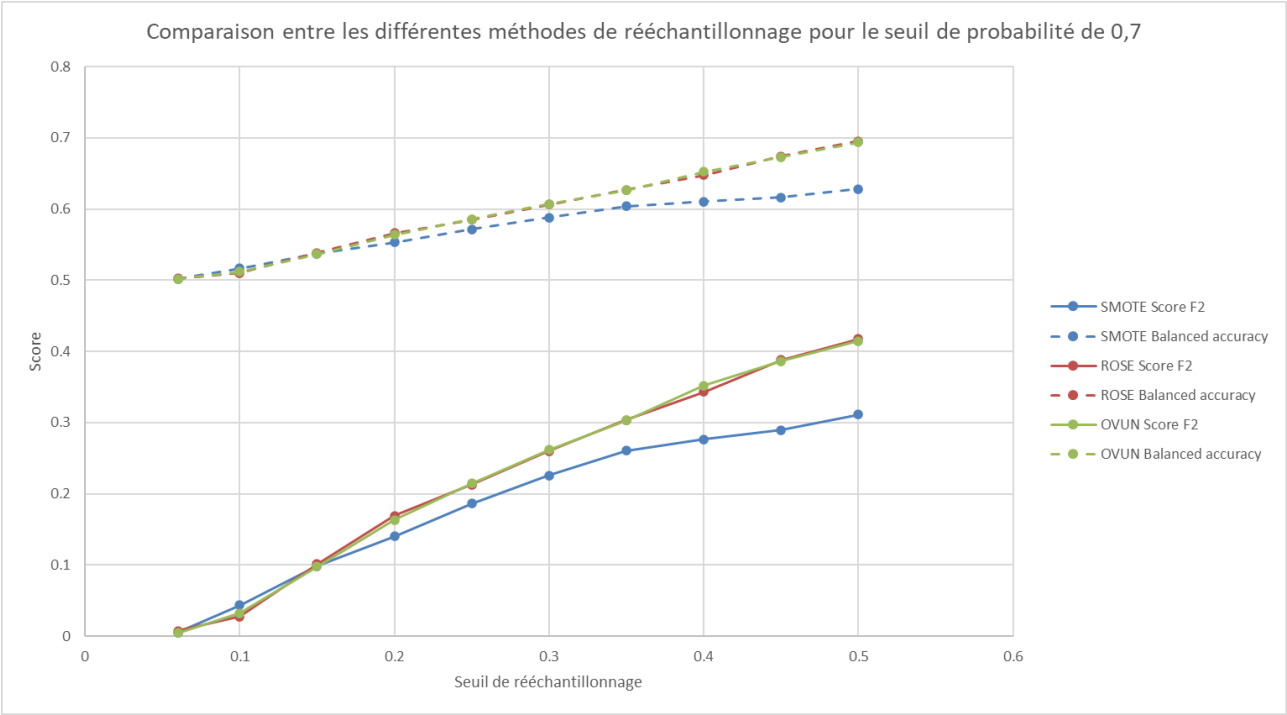


FIGURE 11.14 – Évolution du score F_2 et de la justesse pondérée en fonction du seuil de rééchantillonnage pour chaque méthode, pour le seuil de probabilité de $0,7$, sur le périmètre malussé

Les graphiques 11.12, 11.13 et 11.14 représentent le score F_2 (courbes pleines) et la justesse pondérée (courbes en tiret) des modèles, pour chacune des méthodes (SMOTE en bleu, ROSE en rouge et OVUN en vert) que nous avons construits et ayant été évalué sur notre base de test non-rééchantillonnée. Pour ces modèles aussi, il semblerait que les méthodes ROSE et OVUN obtiennent des résultats très proches et légèrement meilleurs que la méthode SMOTE.

Nous nous concentrerons, dans la suite de l'étude du périmètre malussé, sur la méthode ROSE pour le choix du seuil de probabilité et du taux de rééchantillonnage.

Le graphique 11.15 représente le score F_2 (courbes pleines) et la BA (courbes en tiret) des modèles, pour chacun des seuils de probabilité (0,3 en bleu, 0,5 en rouge et 0,7 en vert). Les choix du taux rééchantillonnage et du seuil de probabilité ne sont, encore une fois, pas évidents, en effet le maximum du score-F2 ne correspond pas nécessairement au maximum de la BA. Néanmoins, nous pouvons ici aussi estimer que le choix de prendre un taux de rééchantillonnage de 0,5, et un seuil de probabilité de 0,5 semble correct. En effet, même si les scores ne sont pas maximisés pour ces paramètres, ils restent à un bon niveau, et l'on ne sacrifie pas trop l'une des deux métriques par rapport à l'autre, puisque ces dernières sont très proches de leurs maxima.

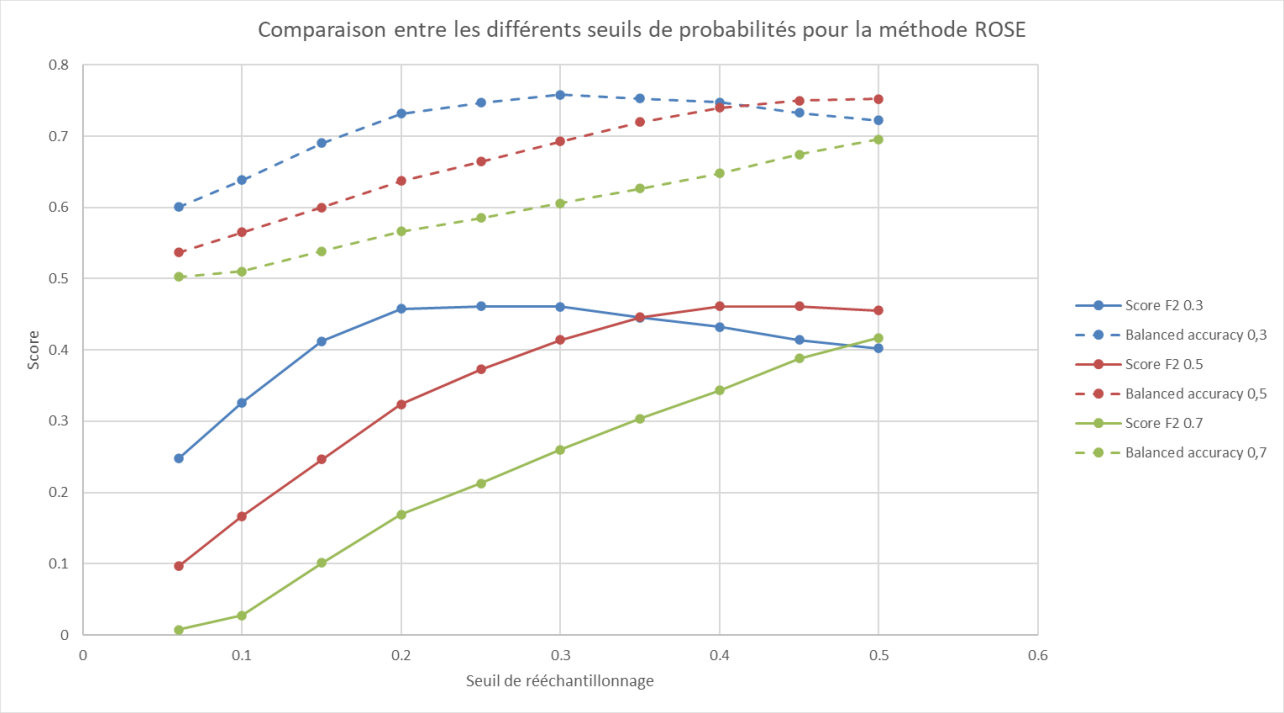


FIGURE 11.15 – Évolution des scores F1 et AUC en fonction du seuil de rééchantillonnage selon les différents taux de rééchantillonnage pour la méthode ROSE, sur le périmètre malussé

11.3.3 Deuxième modèle

Nous construisons donc un deuxième modèle à la suite de ce rééchantillonnage. Nous pouvons alors comparer ses résultats avec notre premier modèle, entraîné sur une base non rééchantillonnée.

	Base initiale	Base rééchantillonnée
Seuil de proba	0.1	0.5
F2-Score	0.46	0.46
BA	0.66	0.75

TABLEAU 11.5 – Résultats sur la base de test des GLMs sur le périmètre malussé

Nous pouvons constater que les résultats obtenus sont meilleurs notamment en termes de justesse pondérée. Aussi, le seuil de probabilité de 0,5 obtenu pour le modèle après rééchantillonnage est une valeur plus classique pour le seuil de probabilité.

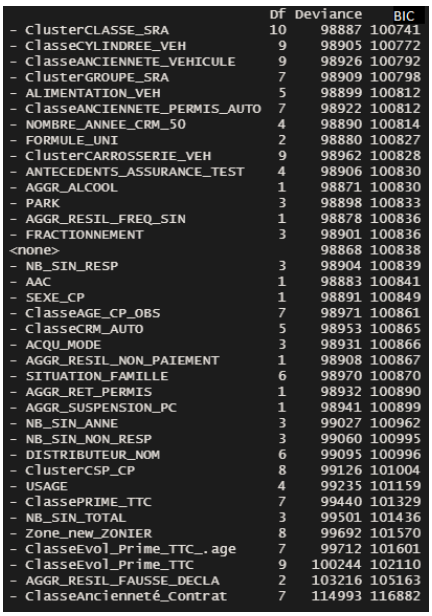
Nous pouvons maintenant essayer de réduire la dimension du modèle saturé pour obtenir un modèle ayant une valeur du BIC plus faible.

11.3.4 Réduction de la dimension du modèle

Comme dans notre étude sur le périmètre standard, nous allons utiliser le critère BIC qui pénalise davantage les modèles aux nombreuses variables. Nous pouvons retrouver le résultat de la première étape de l'algorithme de réduction de dimension pour le périmètre malussé sur la figure 11.16.

Pour cette étape, le modèle va retirer la variable *ClusterClasse_SRA*, ce qui amènera à une amélioration du BIC, qui passerait alors de 100838 (comme indiqué sur la ligne <none>) à 100741. On observe néanmoins une légère augmentation de la déviance du modèle.

Les résultats de l'algorithme sont résumés dans le tableau 11.6. On y observe que le nombre de variables diminue fortement au prix d'une légère augmentation de la déviance résiduelle et de l'AIC. Toutefois, le BIC du second modèle est plus faible que pour le modèle saturé.



```
                                Df Deviance    BIC
- ClusterCLASSE_SRA             10    98887 100741
- ClasseCYLINDREE_VEH            9    98905 100772
- ClasseANCIENNETE_VEHICULE      9    98926 100792
- ClusterGROUPE_SRA              7    98909 100798
- ALIMENTATION_VEH               5    98899 100812
- ClasseANCIENNETE_PERMIS_AUTO   7    98922 100812
- NOMBRE_ANNEE_CRM_50            4    98890 100814
- FORMULE_UNI                    2    98880 100827
- ClusterCARROSSERIE_VEH         9    98962 100828
- ANTECEDENTS_ASSURANCE_TEST     4    98906 100830
- AGGR_ALCOOL                    1    98871 100830
- PARK                           3    98898 100833
- AGGR_RESIL_FREQ_SIN            1    98878 100836
- FRACTIONNEMENT                 3    98901 100836
<none>                                98868 100838
- NB_SIN_RESP                    3    98904 100839
- AAC                            1    98883 100841
- SEXE_CP                        1    98891 100849
- ClasseAGE_CP_OBS               7    98971 100861
- ClasseCRM_AUTO                 5    98953 100865
- ACQU_MODE                      3    98931 100866
- AGGR_RESIL_NON_PAIEMENT        1    98908 100867
- SITUATION_FAMILLE              6    98970 100870
- AGGR_RET_PERMIS                1    98932 100890
- AGGR_SUSPENSION_PC             1    98941 100899
- NB_SIN_ANNE                    3    99027 100962
- NB_SIN_NON_RESP                3    99060 100995
- DISTRIBUTEUR_NOM               6    99095 100996
- ClusterCSP_CP                  8    99126 101004
- USAGE                          4    99235 101159
- ClassePRIME_TTC                7    99440 101329
- NB_SIN_TOTAL                   3    99501 101436
- Zone_new_ZONIER                8    99692 101570
- ClasseEvol_Prime_TTC_.age      7    99712 101601
- ClasseEvol_Prime_TTC           9   100244 102110
- AGGR_RESIL_FAUSSE_DECLA        2   103216 105163
- ClasseAncienneté_Contrat       7   114993 116882
```

FIGURE 11.16 – Premier pas de l'algorithme de réduction du nombre de variables pour le périmètre malussé

Modèle	Nombre de variables	Residual Deviance	Null Deviance	AIC	BIC
GLM_{MAL}^{sat}	36	98868	139501	99210	100838
GLM_{MAL}^{rest}	21	99290	139500	99500	100491

TABLEAU 11.6 – Résultats sur la base de test des GLMs sur le périmètre standard

11.3.5 Importance des variables

Nous allons maintenant étudier l'importance des variables à l'aide de la méthode *PFI*, ce qui permet de créer le graphique 11.17. Ce graphique permet une visualisation rapide des variables qui contribuent le plus aux performances de notre modèle, ce sont donc vraisemblablement à ces variables qu'il faudra faire attention au moment d'établir les nouveaux tarifs d'assurance.

Nous notons, tout d'abord, que la variable la plus importante selon cette méthode est la variable relative à l'ancienneté du contrat. En observant les coefficients, nous constatons que le coefficient le plus faible est celui lié aux contrats de moins d'un an, or les résiliations pour les contrats de moins d'un an ne rentrent pas dans le cadre de notre étude. Le coefficient le plus élevé est cependant celui correspondant aux contrats de 3 ans d'ancienneté, cette valeur peut s'expliquer, en partie, par le fait que le malus s'annule au bout de trois ans sans sinistres, ainsi un assuré ayant attendu que son coefficient de réduction majoration soit revenu à 1 aura tendance à chercher un contrat plus avantageux pour lui. Ensuite, nous remarquons que les comportements face à la résiliation sont différents selon le partenaire, cela indique qu'il est sans doute possible de travailler avec les partenaires les moins performants dans ce domaine pour leur permettre d'améliorer leur rétention. Enfin, nous constatons qu'une des variables liées au caractère d'aggravation du contrat est la troisième variable la plus importante selon cette méthode cela justifie en partie de séparer la modélisation des assurés selon si leur profil est « *a*ggravé » ou non. Enfin, les variables liées à la prime payée par l'assuré sont relativement importantes d'après cette méthode.

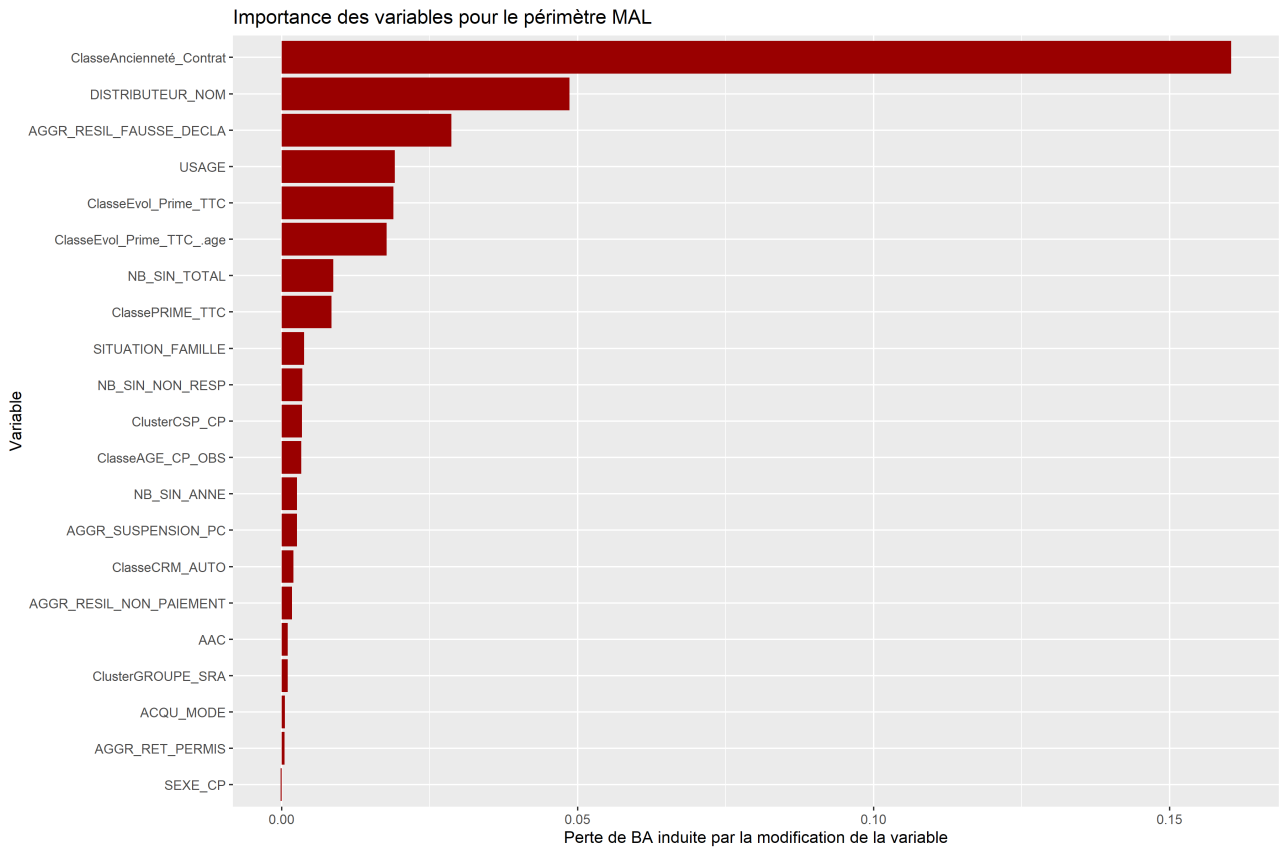


FIGURE 11.17 – Importance des variables pour le modèle GLM malussé

11.4 Résumé de la modélisation logistique

L'un des avantages de la méthode GLM est que l'on peut déterminer simplement à l'aide des coefficients obtenus des probabilités de résiliation pour chacune des modalités et ainsi nous pouvons établir des profils plus ou moins sensibles à la résiliation.

11.4.1 Probabilité et score de résiliation

Nous avons présenté, dans le tableau 11.7, est présenté le résultat d'un GLM ajusté sur les trois variables les plus importantes d'après le graphique 11.10. Ce tableau nous permettra donc de détailler la méthode de calcul des probabilités de résiliation ainsi que le score en découlant.

Modalité	β	Probabilité	Score
Constante du modèle	-2.36106	8.62%	
ClasseAncienneté_Contrat[0.0 ;1.0[	0	8.62%	0
ClasseAncienneté_Contrat[1.0 ;2.0[	1.29298	25.58%	30.59
ClasseAncienneté_Contrat[2.0 ;3.0[	1.3347	26.38%	31.58
ClasseAncienneté_Contrat[3.0 ;5.0[	1.18468	23.57%	28.03
ClasseAncienneté_Contrat[5.0 ;8.0[	0.88112	18.54%	20.85
ClasseAncienneté_Contrat[8.0 ;12.0[	0.80392	17.41%	19.02
ClasseAncienneté_Contrat[12.0 ;16.0[	0.74453	16.57%	17.62
ClasseAncienneté_Contrat>16	0.29831	11.28%	7.05
ClasseEvol_Prime_TTC_%age<-34.27	-0.42133	5.83%	0
ClasseEvol_Prime_TTC_%age[-34.27 ;-15.89[	-0.26091	6.77%	3.79
ClasseEvol_Prime_TTC_%age[-15.89 ;-4.34[	0	8.62%	9.97
ClasseEvol_Prime_TTC_%age[-4.34 ;12.04[	0.82179	17.66%	29.41
ClasseEvol_Prime_TTC_%age[12.04 ;42.25[	0.58039	14.42%	23.70
ClasseEvol_Prime_TTC_%age[42.25 ;91.5[	0.29983	11.29%	17.06
ClasseEvol_Prime_TTC_%age>=91.5	0.02293	8.80%	10.51
USAGEInconnu	0	8.62%	0
USAGEPrivé	1.55097	30.79%	36.70
USAGEProfessionnel	1.23059	24.41%	29.12
USAGEPromenade/trajetstravail	1.64762	32.88%	38.99
USAGETousdéplacements	1.24947	24.76%	29.57

TABLEAU 11.7 – Exemple de probabilité et de score

11.4.1.1 Probabilité de résiliation

La probabilité de résiliation se calcule à l'aide de la fonction de lien du modèle. Par exemple, la probabilité de résiliation d'un assuré dont le contrat a moins d'un an, l'évolution de la prime est comprise entre -15.89% et -4.34% et dont l'usage n'est pas connu (soit la constante du modèle) sera :

$$\begin{aligned}\mathbb{P}(constante) &= g^{-1}(\beta_{constante}) = \frac{exp(\beta_{constante})}{1+exp(\beta_{constante})} \\ &= g^{-1}(-2.36106) = \frac{exp(-2.36106)}{1+exp(-2.36106)} \\ &= 8,62\%\end{aligned} \tag{11.1}$$

Pour obtenir la probabilité de résiliation de la modalité ClasseAncienneté_Contrat[3.0 ;5.0[, le calcul est le suivant :

$$\begin{aligned}\mathbb{P}(Anciennete\_Contrat \in [3,5[) &= g^{-1}(\beta_{constante} + \beta_{AncienneteContrat}) \\ &= \frac{exp(\beta_{constante} + \beta_{AncienneteContrat}}{1+exp(\beta_{constante} + \beta_{AncienneteContrat})} \\ &= \frac{exp(-2.36106 + 1.18468)}{1+exp(-2.36106 + 1.18468)} \\ &= 23,57\%\end{aligned} \tag{11.2}$$

Si l'on souhaite maintenant calculer la probabilité de résiliation d'un assuré dont les modalités ne sont pas présentes dans la constante du modèle, il faut additionner les coefficients β correspondants à celui de la

constante. Ainsi, soit un assuré qui a une ancienneté de contrat comprise entre 3 et 5 ans, une évolution de prime comprise entre -15.89 % et -4.34 % et dont l'usage est "Professionnel". On appellera ce profil, *Profil 1.* La probabilité de résiliation de ce profil est :

$$\begin{aligned}\mathbb{P}(Profil1) &= \frac{exp(\beta_{constante} + \beta_{AncienneteContrat} + \beta_{Evolutiondeprime\%} + \beta_{Usage})}{1 + exp(\beta_{constante} + \beta_{AncienneteContrat} + \beta_{Evolutiondeprime\%} + \beta_{Usage})} \\ &= \frac{exp(-2.36106 + 1.18468 + 0 + 1.55097)}{1 + exp(-2.36106 + 1.18468 + 0 + 1.55097)} \\ &= 59,26\%\end{aligned} \tag{11.3}$$

De la même façon, on peut définir un second profil où l'ancienneté du contrat est comprise entre 1 et 2 ans, l'évolution de prime est comprise entre 12.04 % et 42.25 % et l'usage est "Tousdéplacements". La probabilité de la résiliation associée à ce second profil est :

$$\begin{aligned}\mathbb{P}(Profil2) &= \frac{exp(-2.36106 + 1.29298 + 0.58039 + 1.24947)}{1 + exp(-2.36106 + 1.29298 + 0.58039 + 1.24947)} \\ &= 68,17\%\end{aligned} \tag{11.4}$$

Nous pouvons donc facilement comparer ces deux profils. Ainsi, un assuré du profil 2 aura une probabilité de résiliation $\frac{68.17}{56.26} = 1.15$ fois plus importante qu'un assuré du premier profil.

11.4.1.2 Échelle de score

À l'aide des coefficients β nous pouvons, aussi, établir une échelle de score pour les différents profils d'assuré. Plus ce score sera élevé, plus le risque de résiliation sera important. Pour obtenir ce score, nous avons établi le β_{min} et le β_{max} de chaque variable du modèle, puis calculé l'écart entre ces bornes. Ainsi, nous pouvons obtenir le poids de l'ensemble des variables.

Reprenons l'exemple de notre modèle à trois variables dont les résultats sont exposés dans le tableau 11.7, nous obtenons donc le tableau 11.8

Variable	β_{min}	β_{max}	Δ
Ancienneté Contrat	0	1.3347	1.3347
Évolution de prime (%)	-0.42133	0.82179	1.24312
Usage	0	1.64762	1.64762
			4.22544

Tableau 11.8 – Exemple création de l'échelle de score

Nous pouvons ensuite calculer le score d'une modalité de la façon suivante :

Soit une variable v et une de ses modalités m, on note $S(m, v)$ le score associé

$$S(m, v) = 100 \times \frac{\beta_m - \beta_{v,min}}{\Delta_{total}} \tag{11.5}$$

L'ensemble des scores ainsi calculé pour les modèles standard et malussé sont disponibles en annexe B

11.4.1.3 Rapport des chances

Comme nous l'avons décrit dans la partie théorique nous pouvons calculer des rapports des chances à l'aide des coefficients obtenus à la suite d'un modèle linéaire généralisé. Ces derniers permettent, par exemple, de mesurer l'effet d'une modalité par rapport à une autre, ou la probabilité de résiliation d'un profil par rapport à un autre.

Ce calcul se fait d'après la formule suivante, en notant p et q la probabilité des deux évènements dont on veut calculer le rapport des chances :

$$OR(p/q) = \frac{\frac{p}{1-p}}{\frac{q}{1-q}} \tag{11.6}$$

Ainsi, le rapport des chances de la modalité Évolution de prime entre -4.34 % et 12.04 % (Evol1) et Évolution de prime entre -34.27 % et -15.89 % (Evol2) est le suivant :

$$OR(Evol1/Evol2) = \frac{\frac{0.1766}{1-0.1766}}{\frac{0.0677}{1-0.0677}} = 2.95 \tag{11.7}$$

Nous constatons donc qu'un assuré dont la prime évolue entre -4.34 % et 12.04 % aura 2,95 fois plus de chances de résilier qu'un assuré dont l'évolution de prime est comprise entre -34.27 % et -15.89 %.

De la même façon, en reprenant nos profils définis plus haut, le rapport des chances entre ces derniers est $OR(Profil1/Profil2) = \frac{0.5926/(1-0.5926)}{0.6817/(1-0.6817)} = 0.68$, on peut donc en conclure qu'un assuré du profil 1 aura moins tendance à résilier qu'un assuré du profil 2, puisque le ratio est inférieur à 1. Un assuré du profil 2 aura donc $1/0.68 = 1.47$ fois plus de chance de résilier qu'un assuré du profil 1.

11.4.2 Profils sensibles à la résiliation

À l'aide des scores que nous avons définis, nous pouvons établir des profils plus ou moins sensibles à la résiliation pour chacun de nos périmètres. Ces profils sont résumés dans les tableaux 11.9 et 11.10.

Ainsi, nous pouvons constater qu'il existe plusieurs différences entre les modèles standard et mallusé. Premièrement, les modalités retenues lors de la sélection de variables ne sont pas les mêmes, en plus des variables liées au risque malussé et qui n'apparaissent donc pas dans la modélisation standard, certaines variables comme AAC ou Groupe SRA sont présentes dans le modèle malussé mais pas dans le modèle standard, a contrario certaines variables comme CarrosserieVeh ou Formule sont présentes dans le modèle standard mais absentes du modèle malussé. Deuxièmement, pour les variables conservées dans les deux modèles, nous n'observons pas toujours les mêmes modalités comme étant les plus (ou moins) risqués par rapport à la résiliation. Par exemple, la modalité la moins sensible à la résiliation pour la variable relative au coefficient Bonus/Malus, représente les plus mauvais CRM pour les assurés malussés, ce qui est cohérent car ces derniers ont sans doute davantage de difficultés à trouver un assureur en raison de leur spécificité, en ce qui concerne les assurés du périmètre standard, ce sont les assurés entre 0.75 et 0.83 de CRM qui sont les moins enclins à résilier leurs contrats, cela peut s'expliquer par le fait qu'ils souhaitent d'abord atteindre un CRM plus faible pour pouvoir obtenir une prime plus intéressante.

Périmètre :	Standard		Malussé	
Nom Var	Modalité	Score	Modalité	Score
ClasseAGE_CP_OBS	>=67	0	>=75	0
ClasseAncienneté_Contrat	[0.0 ;1.0[	0	[0.0 ;1.0[	0
ClasseCRM_AUTO	[0.75 ;0.83[	0	>=1.81	0
ClasseEvol_Prime_TTC	<-694.91	0	<-1655.22	0
ClasseEvol_Prime_TTC_%age	<-34.27	0	<-15.15	0
ClassePRIME_TTC	[68.78 ;416.89[	0	[70.45 ;572.3[	0
ClusterCARROSSERIE_VEH	1	0		0
ClusterCSP_CP	7	0	6	0
DISTRIBUTEUR_NOM	5	0	5	0
FORMULE_UNI	RC+BDG+VI+DTA	0		0
NB_SIN_ANNE	>=3	0	>=3	0
NB_SIN_NON_RESP	>=3	0	>=3	0
NB_SIN_TOTAL	>=3	0	>=3	0
PARK	Z.INCONNU	0		0
SEXE_CP	M	0	M	0
SITUATION_FAMILLE	Veuf	0	NC	0
USAGE	NC	0	NC	0
AAC		0	Non	0
ACQU_MODE		0	Crédit	0
AGGR_RESIL_FAUSSE_DECLA		0	OUI	0
AGGR_RESIL_NON_PAIEMENT		0	OUI	0
AGGR_RET_PERMIS		0	OUI	0
AGGR_SUSPENSION_PC		0	OUI	0
ClusterGROUPE_SRA		0	4	0

Tableau 11.9 – Comparaison des profils peu sensibles à la résiliation entre les périmètres standard et malussé

Périmètre :	Standard		Malussé	
Nom Var	Modalité	Score	Modalité	Score
ClasseAGE_CP_OBS	[18.0 ;28.0[	1.88	[18.0 ;29.0[	1.58
ClasseAncienneté_Contrat	[2.0 ;3.0[	6.09	[3.0 ;4.0[	8.18
ClasseCRM_AUTO	>=0.93	1.51	[0.50 ;0.64[	1.67
ClasseEvol_Prime_TTC	[331.72 ;563.86[	10.1	[314.26 ;644.53[	17.74
ClasseEvol_Prime_TTC_%age	[-4.34 ;12.04[	9.89	[0.75 ;12.81[	5.12
ClassePRIME_TTC	>=2712.18	12.8	>=3791.91	7.39
ClusterCARROSSERIE_VEH	7	6.13		0
ClusterCSP_CP	5	14.33	5	26.68
DISTRIBUTEUR_NOM	4	8.44	3	6.44
FORMULE_UNI	RC+BDG+VI	0.21		0
NB_SIN_ANNE	0.0	3.39	0.0	2.08
NB_SIN_NON_RESP	0.0	1.51	0.0	2.23
NB_SIN_TOTAL	0.0	2.64	0.0	2.56
PARK	VoiePublique	9.27		0
SEXE_CP	F	0.19	F	0.16
SITUATION_FAMILLE	NC	1.95	Pacsé	1.56
USAGE	Privé	9.66	Promenade / trajets travail	6.01
AAC		0	Oui	0.23
ACQU_MODE		0	Comptant/crédit	1.14
AGGR_RESIL_FAUSSE_DECLA		0	NC	4.82
AGGR_RESIL_NON_PAIEMENT		0	NON	0.34
AGGR_RET_PERMIS		0	NON	1.94
AGGR_SUSPENSION_PC		0	NON	0.45
ClusterGROUPE_SRA		0	0	1.69

TABLEAU 11.10 – Comparaison des profils les plus sensibles à la résiliation entre les périmètres standard et malussé

De manière plus générale, on peut noter que :

- Les jeunes semblent plus prompts à la résiliation que les personnes plus âgées, en effet pour les deux périmètres, on constate que la classe d'âge regroupant les plus jeunes assurés représente des risques importants de résiliation tandis que la classe correspondant aux risques faibles est dans les deux cas la classe regroupant les personnes les plus âgées ;
- De façon naturelle, les contrats de moins d'un an ne présentent pas de grands risques de résiliation. Comme nous avions pu l'observer sur les graphiques présentant les variables de la base de données, nous constatons que les catégories les plus sensibles à la résiliation sont les contrats âgés de deux ans pour le périmètre standard et de trois ans pour le périmètre malussé. Cela met en exergue une forte rotation de nos assurés ;
- Comme nous l'avons déjà évoqué, les assurés malussés dont le Coefficient de Réduction/Majoration (CRM) est élevé ont un risque faible de résilier, ceci s'explique par le fait que ces assurés auront sans doute du mal à trouver une assurance par ailleurs au vu de leur profil très risqué en termes de sinistralité. Sur ce même périmètre, nous observons que les assurés dont le coefficient est minimal ont une forte propension à résilier, ce résultat est attendu dans la mesure où des assurés malussés avec un faible CRM correspondent probablement à des assurés ayant des aggravations plutôt anciennes et qui peuvent obtenir un contrat standard. Pour le périmètre standard, les résultats sont différents, en effet, nous observons que les assurés ayant un CRM élevé ont davantage tendance à résilier que les autres assurés. On peut relier ce résultat à celui obtenu pour la variable relative à l'ancienneté du contrat, ainsi qu'à l'âge du conducteur, car les jeunes assurés ont un CRM proche de 1 ainsi les résultats obtenus semblent cohérents pour ces trois variables ;
- En ce qui concerne, les évolutions de prime, nous observons des résultats qui semblent logiques, à savoir que les évolutions (en pourcentage de prime ou en valeur absolue) les plus basses correspondent aux profils à faible risque pour les deux périmètres. Aussi, nous observons que les profils les plus risqués correspondent à des augmentations de prime proches de 0 % et donc pas aux augmentations maximales. On peut expliquer cela par le fait que les augmentations maximales correspondent probablement à des changements de contrats et donc que les assurés ne résilient pas à la suite de ces hausses de tarif. De plus, nous avions remarqué de grands pics de résiliation (et d'exposition) autour du 0 pour ces variables dans la section 10.2 ;

- Pour les deux périmètres, nous constatons que les assurés les moins risqués par rapport à la résiliation sont ceux dont la prime est la plus faible, et que les plus risqués sont ceux dont la prime est la plus forte ;
- Il semblerait que les assurés « *Tous-Risques* » présentent un profil moins risqué par rapport à la résiliation, alors que les assurés les plus sensibles semblent être ceux ayant choisi la formule RC+BDG+VI, à savoir une formule intermédiaire entre le *Tiers-Payant* et le *Tous-Risques*
- Les assurés ayant eu des sinistres récents semblent moins sensibles à la résiliation, ce qui paraît cohérent dans la mesure où des derniers auraient possiblement des difficultés à trouver un autre assureur. Les profils les plus risqués correspondent quant à eux aux assurés n'ayant pas eu de sinistres récents ;
- Pour certaines variables comme Usage, Park, ou situation famille les profils "extrêmes" correspondent à l'absence d'information, cela est problématique car nous ne pouvons considérer que ces valeurs sont pertinentes. On note tout de même, concernant ces variables, que les assurés du périmètre standard qui n'ont pas de parking privatif ont une plus forte tendance à résilier, les assurés malussés pacsés semblent plus risqués vis-à-vis de la résiliation alors que pour le périmètre standard les veufs sont moins enclins à résilier. Enfin, concernant la variable Usage, le profil risqué correspond pour le périmètre standard à la modalité "Privé" et à la modalité "Promenade/trajets/travail" pour le périmètre malussé ;
- Le fait d'avoir un conducteur en conduite accompagnée augmente le risque de résiliation pour le périmètre malussé (la variable n'est pas significative pour le périmètre standard) ;
- Les assurés malussés ayant acheté leur véhicule à crédit sont moins sensibles à la résiliation que les autres ;
- Concernant les critères d'aggravation, ceux qui n'en ont pas sont plus sensibles à la résiliation, ce résultat est logique dans la mesure où il est plus compliqué de se faire assurer lorsque nous avons déjà été résilié par un assureur par exemple.

Chapitre 12

La modélisation par Forêts Aléatoires

Pour les modèles de Forêts Aléatoires, nous reprenons les bases sans partitionnement, en effet l'un des intérêts des algorithmes de *Machine Learning* basés sur des arbres de décisions est qu'ils vont scinder une variable à chaque création de nœud. Un partitionnement créé en amont briderait donc les capacités prédictives de ces modèles. De plus, il n'est pas nécessaire de prendre une base rééchantillonnée car, il est possible soit d'équilibrer les classes directement lors de la création des modèles de *forêts aléatoires* soit de pondérer le poids des individus, notamment selon la classe, pour pénaliser ou rendre plus importante les observations lors de la création des modèles.

12.1 Outils

Pour la modélisation par forêts aléatoires, nous avons utilisé le langage Python et notamment la fonction *RandomForestClassifier*[1] de la bibliothèque *scikit-learn*[2].

Cette fonction a de nombreux hyper-paramètres que l'on peut régler, nous allons néanmoins nous concentrer sur les suivants :

- n_estimators : Nombre d'arbres créés pour construire la forêt aléatoire ;
- criterion : Le critère de mesure de la qualité de la division du nœud ;
- max_depth : Profondeur maximale de l'arbre ;
- min_samples_leaf : Nombre minimal d'individus dans une feuille ;
- max_features : Nombre de variables utilisées à la division des nœuds ;
- class_weight : Poids associés aux classes de la variable réponse.

Pour régler les hyper-paramètres nous avons utilisé la fonction *GridSearchCV*[3], issue aussi de la bibliothèque *scikit-learn*. Cette fonction permet d'ajuster plusieurs modèles selon un dictionnaire de paramètres que l'on définit puis d'effectuer une validation croisée des modèles ainsi créés.
Les principaux paramètres qui nous intéressent pour cette fonction sont les suivants :

- estimator : Le modèle que l'on choisit d'ajuster, ici ce sera RandomforestClassifier ;
- param_grid : Dictionnaire des paramètres que l'on veut tester ;
- scoring : Métrique(s) utilisée(s) pour évaluer le modèle ;
- refit : Ce paramètre permet de choisir, si l'on souhaite réajuster le *meilleur* modèle sur l'ensemble du jeu de données après avoir évalué les modèles. Pour définir le meilleur modèle, il est nécessaire, dans le cas où l'on a choisi différentes métriques dans *scoring*, de définir soit celle qui est la plus importante, soit une stratégie de choix si l'on veut par exemple utiliser plusieurs métriques ;
- cv : La stratégie de cross-validation, nous utiliserons une validation croisée à 5 blocs ;
- return_train_score : Ce paramètre permet d'obtenir les scores calculés pendant la validation croisée.

12.2 Périmètre standard

Dans un premier temps, concentrons-nous sur le périmètre standard. La première chose à faire est de régler les hyper-paramètres pour obtenir les meilleurs scores pour le modèle final.

1. https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html
2. https://scikit-learn.org/stable/index.html
3. https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.GridSearchCV.html

12.2.1 Réglage des paramètres du modèle

Nous allons commencer par choisir le nombre d'arbres créés (n_estimators) ainsi que le nombre de variables utilisées lors de la division des nœuds (max_features) à l'aide du score Out-Of-Bag (OOB) des modèles. Pour choisir ces paramètres, nous avons créé le graphique 12.1

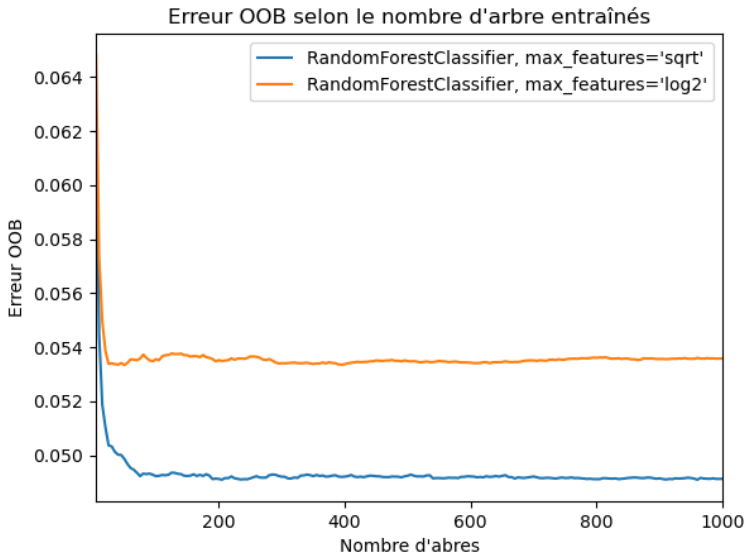


FIGURE 12.1 – Comparaison des stratégies pour max_features en fonction du nombre d'arbres

Soit k le nombre de variables disponibles dans la base d'entraînement, les différentes stratégies représentées sur le graphique 12.1 correspondent à :

- sqrt : $max\_features = \sqrt{k}$, on utilise donc la racine carrée du nombre de variables, ce choix est la valeur conseillée par Breiman dans le cas d'une classification ;
- log2 : $max\_features = log_2(k)$, on considérera donc le log en base 2 du nombre de variables ;
- None : $max\_features = k$, on essayera toutes les variables à chaque division.

Dans la mesure où le choix d'utiliser l'ensemble des variables ne nous semblait pas pertinent, et pour accélérer le temps de calcul l'option max_features = None n'a pas été testée. Ainsi, nous observons que le choix des paramètres $n\_estimators = 600$ et $max\_features = "sqrt"$ semblent pertinents. En effet, on observe une stabilisation de l'erreur OOB après le seuil d'à peu près 600 arbres. Aussi, il semblerait que cette erreur soit minimisée lorsque l'on considère $\sqrt{k}$ variables lors de la création des nœuds par rapport à $log_2(k)$.

Ensuite nous avons utilisé la fonction GridSearchCV pour comparer les paramètres criterion, max_depth, min_samples_leaf et class_weight. Nous avons défini une fonction permettant de définir la stratégie de sélection du *meilleur* modèle. L'objectif de cette démarche est de pouvoir adapter cette fonction aux métriques qui nous intéressent le plus, nous avons donc choisi de mettre en avant le modèle qui permettait un bon compromis entre la justesse pondérée et le score F_2. Le processus de sélection est le suivant :

1. On calcule les scores BA et F_2 pour tous les modèles ;
2. On conserve les 30 % des modèles les plus performants selon le score BA ;
3. Soit min_{stdBA} le plus faible écart-type parmi les modèles restants, et σ_{stdBA} l'écart-type des écarts-types de chaque modèle restant. On élimine les modèles dont l'écart-type est éloigné de plus de σ_{stdBA} de min_{stdBA}.
4. On choisit le modèle qui maximise le score F_2 parmi les modèles restants.

Pour le paramètre min_samples_leaf, nous avons testé différentes valeurs : 1 ou différents taux t de façon à obtenir des feuilles peuplées d'au minimum $t \times n$ individus (avec n le nombre de d'individus dans l'échantillon de modélisation). Toutefois les résultats étant moins bons pour les modèles entraînés avec une autre valeur que 1, nous avons décidé de régler ce paramètre à 1 pour le choix des autres paramètres.

Sur les graphiques disponibles sur la figure 12.2, nous pouvons observer que les scores BA et F_2 ne sont pas maximisés pour les mêmes paramètres, d'où l'intérêt de définir une stratégie de choix réalisant un compromis entre ces deux métriques. Nous observons que le score F_2 est croissant en fonction de la profondeur des arbres mais cela n'est pas le cas du score de justesse pondérée. La stratégie de choix du modèle établi que le *meilleur* compromis est de choisir les paramètres suivants :

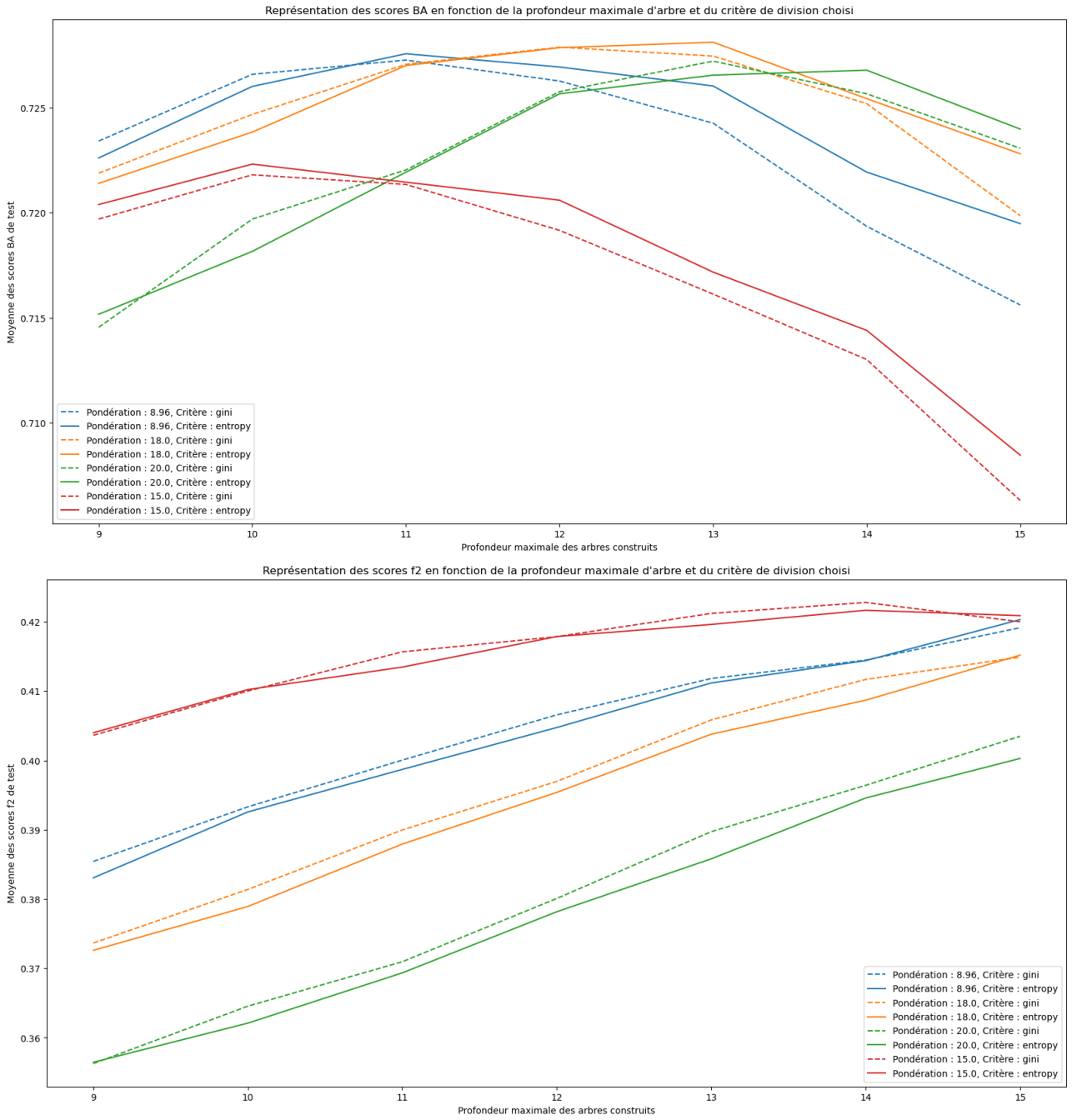


FIGURE 12.2 – Comparaison des scores BA et F_2 en fonction des hyper-paramètres du modèle

- n_estimators = 600 (choisi précédemment) ;
- max_features = 'sqrt' (choisi précédemment) ;
- min_samples_leaf = 1 (choisi précédemment) ;
- criterion = 'entropy' (choisi entre 'gini' ou 'entropy') ;
- max_depth = 15 (choisi parmi 9, 10, 11, 12, 13, 14, 15) ;
- class_weight = 0 : 1, 1 :18 (choisi parmi 0 : 0.53, 1 : 8.96, 0 :1, 1 : 18, 0 :1, 1 : 20, 0 :1, 1 : 15).

Lors du réglage de ces paramètres, nous n'avons pas observé de sur-apprentissage des modèles construits. En effet, les scores obtenus lors de la validation croisée et ceux obtenus avec la base de validation du modèle sont proches. De plus, la stratégie de choix de la combinaison des paramètres prend en compte l'écart-type des scores des modèles et tente de conserver une faible variance entre les échantillons de test de la validation-croisée [4].

Nous avons résumé les résultats de la validation-croisée pour ce modèle dans le tableau 12.1.

4. Un modèle en sur-apprentissage n'aurait pas de grand pouvoir généralisation, l'écart-type des scores de chaque sous-échantillons serait alors important.

Score	Split0	Split1	Split2	Split3	Split4	Moyenne	Validation
F2	0.4171	0.4143	0.4127	0.4184	0.4132	0.4151	0.4094
BA	0.7232	0.7216	0.7226	0.7272	0.7223	0.7234	0.7228

TABLEAU 12.1 – Score de la validation croisée par split pour les paramètres sélectionnés

Les résultats obtenus sur la base de validation en terme de justesse pondérée et de score F_2 sont présentés dans le tableau 12.2. On y observe que le gain est plus important en terme de score F_2 qu'en terme de justesse pondérée. On peut toutefois, conclure que le modèle Random Forest obtient de meilleurs résultats en terme de prédiction.

	Score F_2	BA
GLM	0.36	0.70
Random Forest	0.41	0.72

TABLEAU 12.2 – Comparaison des scores entre Random Forest et GLM

Nous pouvons désormais nous intéresser à l'importance attribuée aux variables par ce modèle pour les comparer avec les résultats de l'étude analogue menée avec le GLM.

12.2.2 Importance des variables

Nous allons donc maintenant étudier l'importance des variables dans la prédiction de la forêt aléatoire. Rappelons les résultats principaux de l'analyse de l'importance des variables du modèle GLM standard développés en 11.2.5 :

- La variable *Ancienneté_ Contrat* est la plus impactante sur la qualité du modèle ;
- Les variables liées aux primes et à leurs évolutions ont une importance certaine ;
- La variable *Usage* est la deuxième variable la plus importante.

Il est important de rappeler que les variables qualitatives ont dû être "binarisée" pour construire la forêt aléatoire, ainsi on ne peut s'attendre à obtenir des importances de variables tout à fait égales à celles observées pour le GLM.

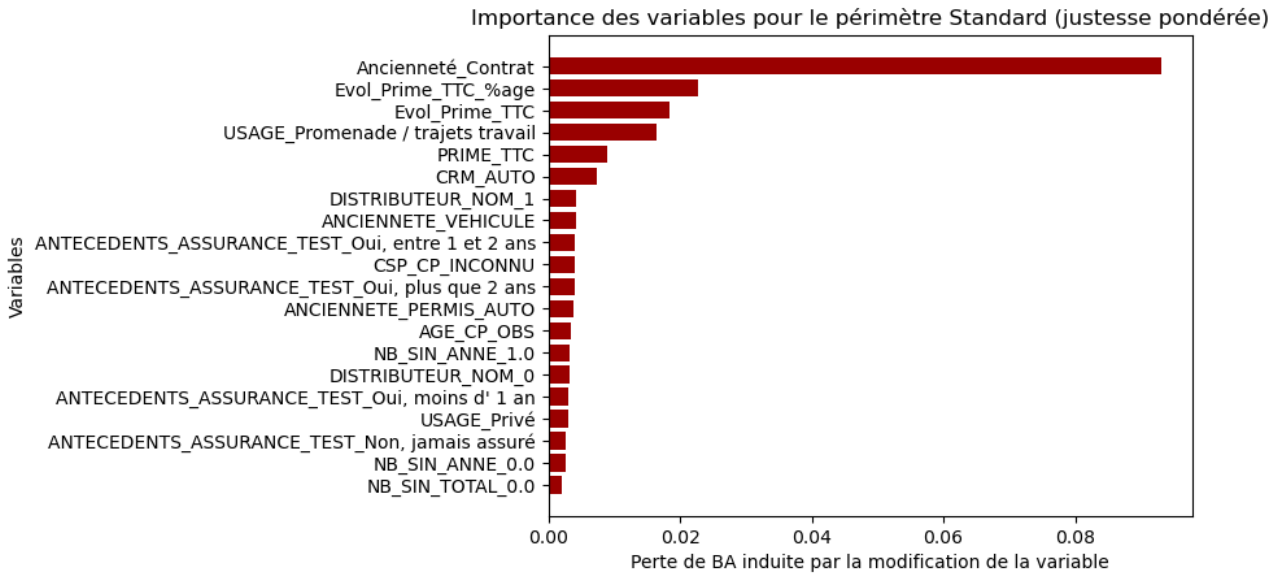


FIGURE 12.3 – Importance des variables selon pour le périmètre standard

Il est important de rappeler, ici, qu'il est nécessaire de *binariser* nos variables qualitatives pour créer nos modèles de forêts aléatoires, ainsi nous obtenons l'importance de certaines modalités, et non l'importance globale de la variable avec la méthode de la PFI.

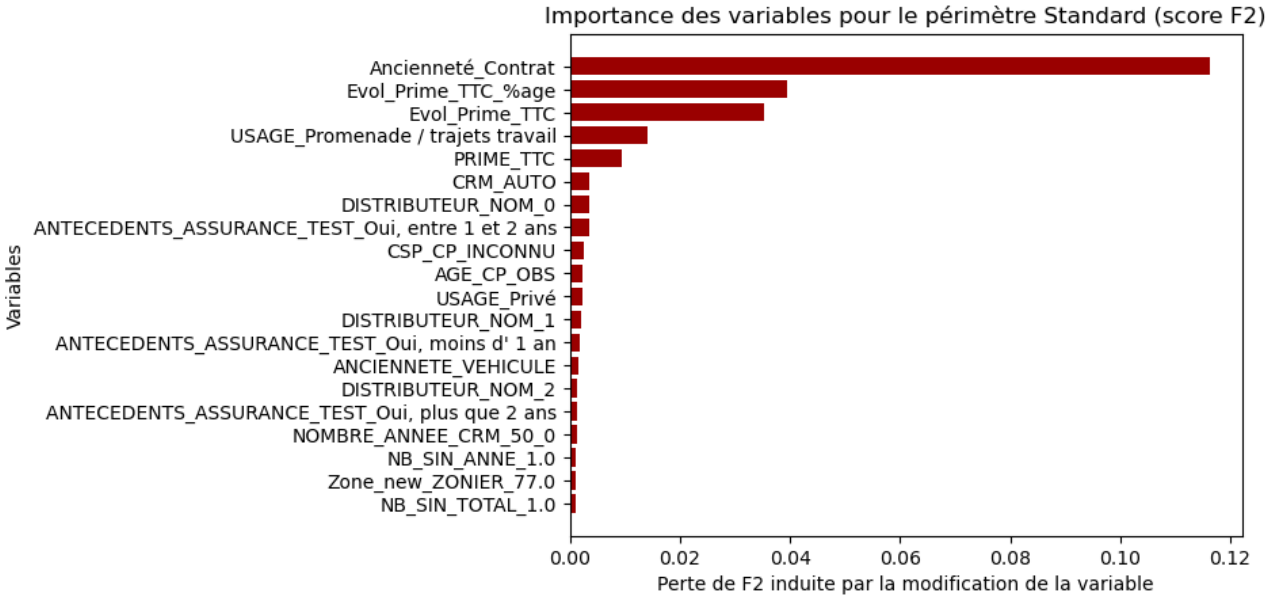


FIGURE 12.4 – Importance des variables selon le score F_2 pour le périmètre standard

Sur les graphiques 12.3 et 12.4, nous pouvons observer les vingt variables les plus importantes par score pour le modèle *Random Forest* entraîné sur le périmètre standard. Nous remarquons que les variables n'ont pas la même importance selon le choix du score. Néanmoins nous observons tout de même des similarités entre les deux graphiques. Nous pouvons noter que la variable la plus importante, quelques soit le score est la variable *Ancienneté_ Contrat*, de plus les variables liées aux primes semblent aussi très impactantes dans la modélisation de la résiliation par forêt aléatoire. Aussi, nous pouvons observer que différentes modalités de la variable *USAGE* apparaissent dans chacun des graphiques 12.3 et 12.4. Nous pouvons donc conclure que les variables les plus importantes de notre modèle GLM le sont toujours dans la modélisation par forêts aléatoires. Ces résultats permettent d'ores et déjà d'établir des pistes de réflexion quant aux variables auxquelles il faudra s'intéresser lors de prochaines revalorisations.

Nous observons également que des variables qui avaient été éliminées lors de la réduction de la dimension du modèle GLM apparaissent ici. C'est, par exemple, le cas de la variable *ANTECEDENTS_ ASSURANCE_ TEST* dont les modalités apparaissent dans les graphiques d'importance, quelques soit le score.

Pour conclure sur la contribution des variables dans la qualité du modèle de forêt aléatoire, nous pouvons établir que lors de prochaines revalorisations, il serait intéressant de paramétrer les revalorisations selon les variables les plus importantes obtenues avec cette méthode. Ainsi, nous notons qu'en plus de l'ancienneté contrat de l'usage fait du véhicule et des variables liées aux primes, il est intéressant de moduler la revalorisation selon le CRM de l'assuré ou l'âge de l'assuré par exemple. Cela demande cependant d'avantage de travail pour savoir quels sont les profils les plus sensibles à la résiliation selon ces variables. Nous observons aussi que le comportement de résiliation semble impacté entre autres par le partenaire auquel est rattaché l'assuré, ainsi, une étude plus poussée pourrait permettre d'aider nos partenaires les moins performants à améliorer la rétention de leur portefeuille des profils rentables.

12.3 Périmètre malussé

Nous allons maintenant nous intéresser au périmètre malussé. Les bases de données malussé et standard étant de dimensions différentes, il est nécessaire de faire un nouveau paramétrage du modèle de forêt aléatoire.

12.3.1 Réglage des paramètres du modèle

Rappelons les paramètres que nous allons optimiser pour obtenir le meilleur algorithme possible :

- n_estimators ;
- criterion ;
- max_depth ;
- min_samples_leaf ;
- max_features ;

- class_weight.

Nous allons donc régler dans un premier temps les paramètres n_estimators et max_features à l'aide de l'estimation de l'erreur Out-Of-Bag. Ensuite, nous réaliserons une validation-croisée pour obtenir la *meilleure* combinaison possible entre les différents paramètres restant à régler selon la même stratégie de choix que pour le périmètre standard.

Nous pouvons nous aider du graphique 12.5 pour déterminer un choix idéal du nombre d'estimateurs et du nombre de variables utilisées pour la division des nœuds. On peut noter que la stratégie visant à prendre $\sqrt{k}$ pour k variables initiales semble plus adaptée à notre problème. Nous pouvons aussi constater sur le graphique une baisse de l'erreur Out-Of-Bag autour de 750 arbres construits dans la forêt. Nous établirons donc pour la suite que les paramètres optimaux pour les variables n_estimators et max_features sont :

- $n\_estimators = 750$;
- $max\_features = "sqrt"$

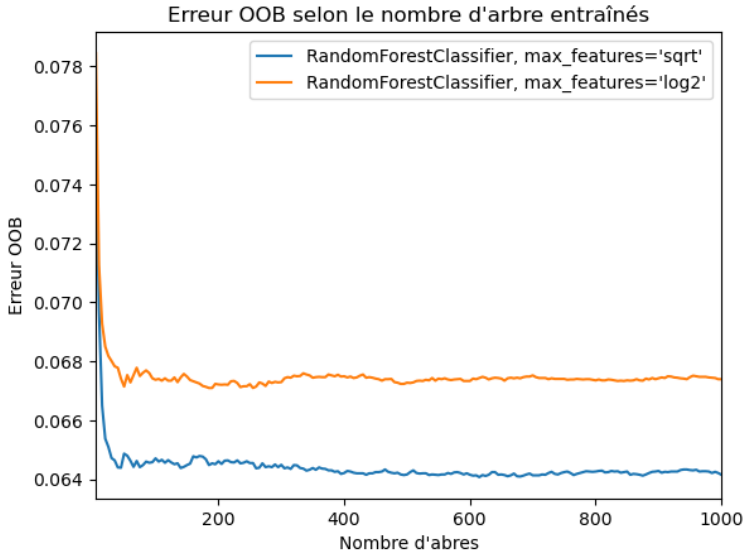


FIGURE 12.5 – Comparaison des stratégies pour max_features en fonction du nombre d'arbres

Par la suite, nous avons utilisé la fonction GridSearchCV pour définir des paramètres pertinents pour notre forêt aléatoire. Nous pouvons résumer les résultats obtenus à l'aide des graphiques 12.6. Nous pouvons y observer les scores F_2 et BA en fonction des différents paramètres testé : la profondeur maximale des arbres, le critère de division des nœuds et la pondération des classes de la variable réponse. Nous pouvons observer sur ce graphique que la justesse pondérée chute lorsque la profondeur de l'arbre dépasse 13 quelques soit la pondération choisie, il semblerait donc qu'une profondeur maximale de 12 ou 13 selon la pondération choisie soit optimale. Sur ce périmètre aussi le score F_2 est croissant de la profondeur jusqu'à un certain seuil (dépendant de la pondération choisie) à partir duquel il va décroître.

On obtient, à la suite, de cette validation-croisée les réglages suivants :

- n_estimators = 750 (choisi précédemment) ;
- max_features = 'sqrt' (choisi précédemment) ;
- min_samples_leaf = 1 (choisi précédemment) ;
- criterion = 'gini' (choisi entre 'gini' ou 'entropy') ;
- max_depth = 12 (choisi parmi 8, 9, 10, 11, 12, 13, 14, 15) ;
- class_weight = 0 : 1, 1 :16 (choisi parmi 0 : 0.53, 1 : 7.46, 0 :1, 1 : 16, 0 :1, 1 : 18, 0 :1, 1 : 20).

Les résultats diffèrent du modèle entraîné pour le périmètre standard, en effet, la profondeur maximale est de 12 contre 15 dans le périmètre standard et la pondération est de 1 pour la classe 0 et 16 pour la classe 1 contre 1 pour la classe 0 et 18 pour la classe 1 pour le périmètre standard. Ce dernier paramètre s'explique par le fait que la proportion des 0 et des 1 pour la variable réponse ne sont pas les mêmes dans les deux bases d'études.

Nous observons une plus grande variance dans les résultats des différents scores calculés lors de la validation-croisée, présentés dans le tableau 12.3, que pour le périmètre standard, l'écart type reste néanmoins très faible (0.005 pour le modèle finalement choisi), et les scores moyens établis lors de la validation-croisée sont très

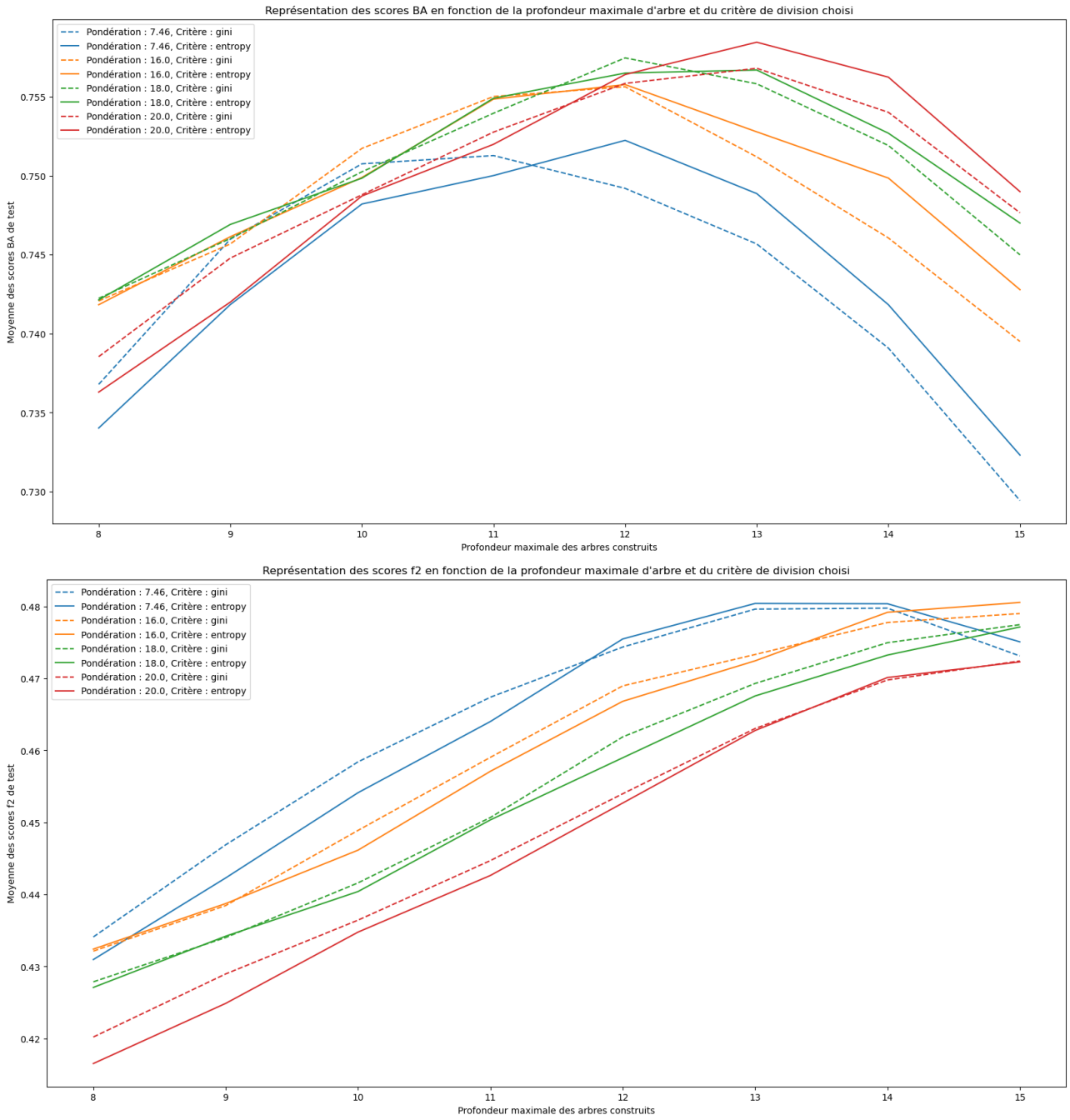


FIGURE 12.6 – Comparaison des scores BA et F_2 en fonction des hyper-paramètres du modèle

Score	Split0	Split1	Split2	Split3	Split4	Moyenne	Validation
F2	0.4765	0.4749	0.4785	0.4768	0.4748	0.4763	0.4751
BA	0.7608	0.76	0.7498	0.7503	0.7606	0.7563	0.7553

TABLEAU 12.3 – Score de la validation croisée par split pour les paramètres sélectionnés

proches des scores évalués sur la base de validation. Nous en concluons donc que le sur-apprentissage du modèle reste assez faible et que notre modèle garde un pouvoir de généralisation important.

Nous allons maintenant comparer les résultats de la modélisation par forêts aléatoires aux résultats obtenus lors de la modélisation par GLM, à l'aide du tableau 12.4. Rappelons que ces résultats ont été évalués sur la même base de données et que les modèles ont aussi le même support d'entraînement.

On observe donc une très légère amélioration des résultats entre la modélisation GLM et la modélisation RF pour ce périmètre.

	Score F_2	BA
GLM	0.46	0.75
Random Forest	0.48	0.76

TABLEAU 12.4 – Comparaison des scores entre Random Forest et GLM

Nous pouvons, maintenant, nous intéresser à l'importance des variables induites par le modèle de forêt aléatoire obtenu.

12.3.2 Importance des variables

Nous allons, ici, étudier l'importance des variables obtenues grâce à la méthode *PFI*, pour le modèle de forêt aléatoire construit pour le périmètre malussé. Nous pourrons ensuite comparer ces résultats à ceux obtenus pour la modélisation GLM. Dans un premier temps, nous pouvons rappeler les conclusions tirées du graphique 11.17 représentant l'importance des variables dans le modèle GLM :

- La variable *Ancienneté_ Contrat* est la variable dont une modification perturbe le plus le modèle ;
- Certaines variables liées aux critères d'aggravation sont importantes ;
- Les variables liées aux primes sont classées $5^{\text{ème}}$, $6^{\text{ème}}$ et $8^{\text{ème}}$ et ont donc une certaine importance.

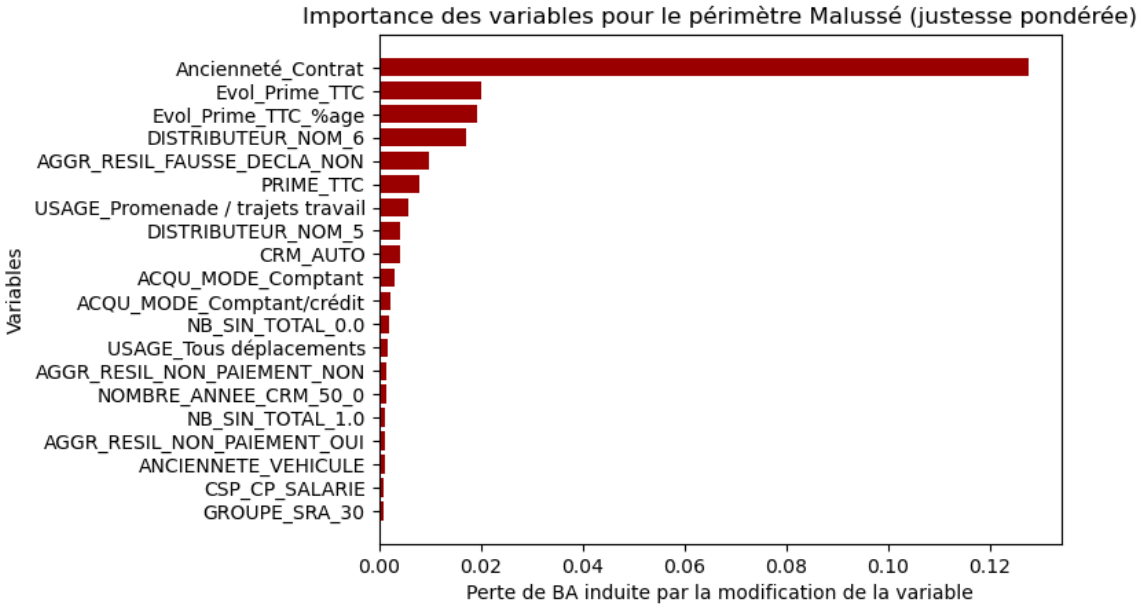


FIGURE 12.7 – Importance des variables selon la justesse pondérée pour le périmètre malussé

Nous observons sur les graphiques 12.7 et 12.8 que la variable la plus importante selon cette méthode reste *Ancienneté_ Contrat*. Les variables liées aux pirmes conservent une grande importance dans ce modèle. Nous observons aussi que certaines variables liées aux aggravations sont représentées parmi les vingt variables les plus importantes du modèle, nous constatons, notamment, la présence des variables *AGGR_ RESIL_ FAUSSE_ DECLA*, *AGGR_ SUSPENSION_ PC* ou encore *AGGR_ RESIL_ NON_ PAIEMENT*. Ainsi, nous concluons que les variables liées aux primes et aux aggravations sont des variables déterminantes pour modéliser la résiliation des assurés.

Il est important de remarquer que selon le périmètre, ce ne sont pas les mêmes partenaires qui obtiennent une grande importance d'après la méthode PFI, ainsi réalisé un travail avec chaque partenaire pour améliorer la rétention est important et permettra d'améliorer le chiffre d'affaires de toutes les entités concernées.

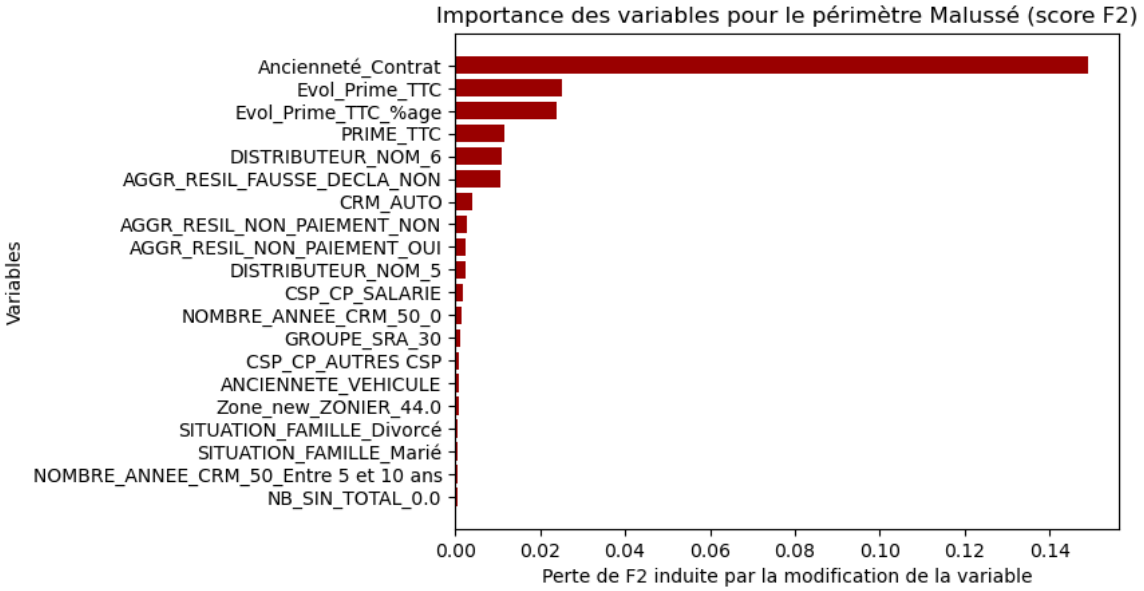


FIGURE 12.8 – Importance des variables selon le score F_2 pour le périmètre malussé

12.4 Résumé de la modélisation par forêts aléatoires

Pour résumer, la modélisation par forêts aléatoires obtient de meilleurs résultats que les GLM que nous avons entraînés. Cependant, nous n'obtenons pas de coefficients pour chaque variable et nous perdons, donc, la capacité de créer une échelle de scoring comme nous l'avions fait à la suite de la régression logistique.

L'étude de l'importance des variables a montré que les variables les plus importantes du modèle de régression logistique sont aussi importantes dans la modélisation par forêts aléatoires. Ainsi, cela nous permet d'obtenir des premières pistes de réflexion pour les futures revalorisations car nous connaissons désormais quelles sont les variables qui semblent avoir le plus grand impact sur l'acte de résiliation pour les assurés. La méthode *PFI* sur le modèle de forêts aléatoires a aussi mis en exergue certaines variables qui semblaient moins importantes pour le GLM, ainsi nous pouvons aussi réfléchir à moduler la revalorisation des contrats selon le coefficient de réduction-majoration (CRM) dans l'optique de conserver les profils les plus rentables en portefeuille, alors que cette variable, qui avait été découpée en classe, ne semblait pas très impactante dans la modélisation par GLM.

Chapitre 13

La modélisation CatBoost

Dans ce chapitre, nous allons utiliser une méthode de boosting afin de modéliser la résiliation sur les périmètres standard et malussé. Pour cela, nous allons utiliser les bases d'entraînements initiales (sans rééchantillonnage ni partitionnement des variables). Utiliser une base rééchantillonnée n'est pas pertinent car les algorithmes de *Machine Learning* autorisent des pondérations des individus, selon leurs classes par exemple. Le partitionnement des variables serait lui contre productif car les algorithmes de *Machine Learning* basés sur des arbres de décisions séparent "en deux" les variables à chaque nœud, créer des classes a priori briderait donc les résultats du modèle.

13.1 Outils

Pour la méthode de boosting que nous avons considérée, nous avons utilisé la bibliothèque *CatBoost*[1] de Python, en particulier, nous avons utilisé la fonction *CatBoostClassifier* dont nous pouvons détailler les principaux paramètres pour obtenir le meilleur modèle possible :

- iterations : Nombre d'itérations du Boosting ;
- scale_pos_weight : Pondération de la classe 1 dans le cas d'une classification binaire ;
- learning_rate : Contrôle le taux d'apprentissage du modèle en modifiant la contribution de chaque arbre lorsque ce dernier est ajouté à l'approximation courante ;
- max_depth : Profondeur maximale des arbres ;
- colsample_bylevel : Proportion de variables utilisées pour la séléction de chaque division ;
- bootstrap_type : Paramètre influant sur le choix de la division des nœuds de l'arbre ;
- min_child_samples : Nombre minimum d'individus dans une feuille ;
- subsample : Proportion des individus de la base initiale utilisés dans la création des arbres.

Pour établir les paramètres optimaux, nous avons utilisé la fonction GridSearchCV que nous avons déjà définie précédemment. Pour rappel, la stratégie de sélection du meilleur modèle est la suivante :

1. On calcule les scores BA et F_2 pour tous les modèles ;
2. On conserve les 30% des modèles les plus performants selon le score BA ;
3. Soit min_{stdBA} le plus faible écart-type parmi les modèles restants, et σ_{stdBA} l'écart-type des écarts-types de chaque modèle restant. On élimine les modèles dont l'écart-type est éloigné de plus de σ_{stdBA} de min_{stdBA}.
4. Enfin, on choisit le modèle qui maximise le score F_2 parmi les modèles restants.

13.2 Périmètre standard

Nous allons dans un premier temps nous intéresser au périmètre standard. Pour commencer, il est nécessaire de définir les paramètres qui nous permettront d'obtenir le meilleur modèle possible. Dans la mesure où le temps d'entraînement des modèles est relativement long, nous allons procéder par étapes en établissant des paramètres pertinents à chaque étape. Nous allons d'abord définir les paramètres iterations, learning_rate et max_depth, puis nous définirons le paramètre scale_pos_weight. Ensuite nous regarderons de nouveau max_depth en parallèle de min_child_samples car ces paramètres sont fortement liés. Enfin, nous établirons les paramètres colsample_bylevel bootstrap_type et subsample.

1. https://catboost.ai/

Nous avons défini plusieurs paramètres pour accélérer les temps de calcul. Ainsi, l'un des atouts de l'algorithme CatBoost selon ses auteurs est sa grande rapidité lorsqu'il est entraîné en utilisant le GPU, c'est-à-dire en utilisant la puissance de la carte graphique et non uniquement le processeur comme on peut classiquement faire. Cet effet a pu être observé lors de différents essais d'ajustement de modèles avec le CPU ou le GPU où le GPU était plus performant, l'écart de performance augmentant en parallèle du nombre d'arbres créés notamment.

Pour l'ajustement des paramètres iterations, learning_rate et max_depth, nous avons ajusté des modèles selon différentes combinaisons de ces paramètres. Nous avons aussi établi le paramètre auto_class_weights="Balanced" pour tous les modèles, ce dernier permet de pondérer les classes de la variable réponse de façon à ce qu'elles aient le même poids dans la modélisation. Les résultats obtenus pour le paramétrage de iterations, learning_rate et max_depth sont résumés dans la figure 13.1.

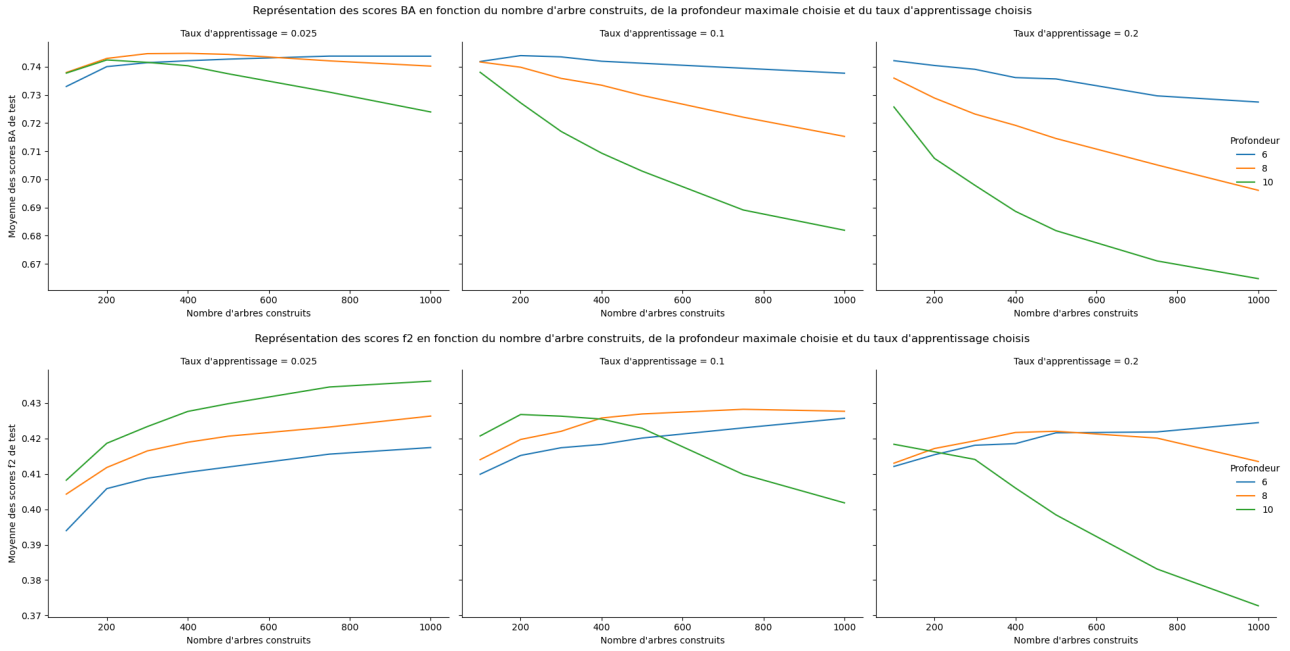


FIGURE 13.1 – Résumé des scores de justesse pondérée et F_2 pour le paramétrage d'iterations, learning_rate et max_depth

On observe sur les graphiques 13.1 que le taux d'apprentissage qui semble le plus adapté à notre problème est 0.025, en effet les scores semblent inférieurs pour les autres taux, de plus on peut noter que, dans certains cas, les scores décroissent lorsque le nombre d'arbres augmente cela peut indiquer un sur-apprentissage du modèle qui connaît trop bien le modèle et perd donc toute capacité de généralisation. Ainsi, le choix d'un grand nombre d'arbres ne semble pas pertinent. De la même façon, ces graphiques montrent que faire le choix de la profondeur maximale n'est pas nécessairement le meilleur choix.

Ainsi les paramètres retenus d'après la stratégie de sélection des paramètres sont les suivants :

- iterations : 300, choisi parmi [100, 200, 300, 400, 500, 750, 1000] ;
- learning_rate : 0.025, choisi parmi [0.025, 0.1, 0.2] ;
- max_depth : 8, choisi parmi [6, 8, 10].

Nous n'observons pas de grands écarts-types pour les scores calculés sur les échantillons de test durant la validation-croisée, de plus, les scores calculés sur la base de validation sont très proches du score moyen issu de la validation-croisée.

Nous allons maintenant régler le paramètre scale_pos_weight, pour cela nous allons prendre les paramètres obtenus précédemment et ajuster des modèles avec différents poids pour la classe minoritaire, qui nous intéresse dans notre problème. Nous obtenons ainsi le graphique 13.2, celui-ci représente les résultats des scores obtenus lors de l'exécution de la fonction GridSearchCV ainsi que l'écart-type des scores représenté sous la forme de barre d'erreur. Nous constatons que le choix le plus pertinent semble être de choisir une pondération entre 11 et 17, en effet, pour une pondération de 11, nous obtenons le meilleur score F_2 et pour une pondération de 17, nous obtenons la meilleure justesse pondérée, de plus, les écarts-types sont faibles pour ces valeurs.

Pour conclure, le choix établi par la stratégie de sélection est de prendre un poids de 15 pour la classe 1.

La prochaine étape de ces réglages est d'ajuster les paramètres max_depth et min_child_samples en parallèle. Nous pouvons comparer les résultats des ajustements sur le graphique 13.3. Nous constatons que, malgré

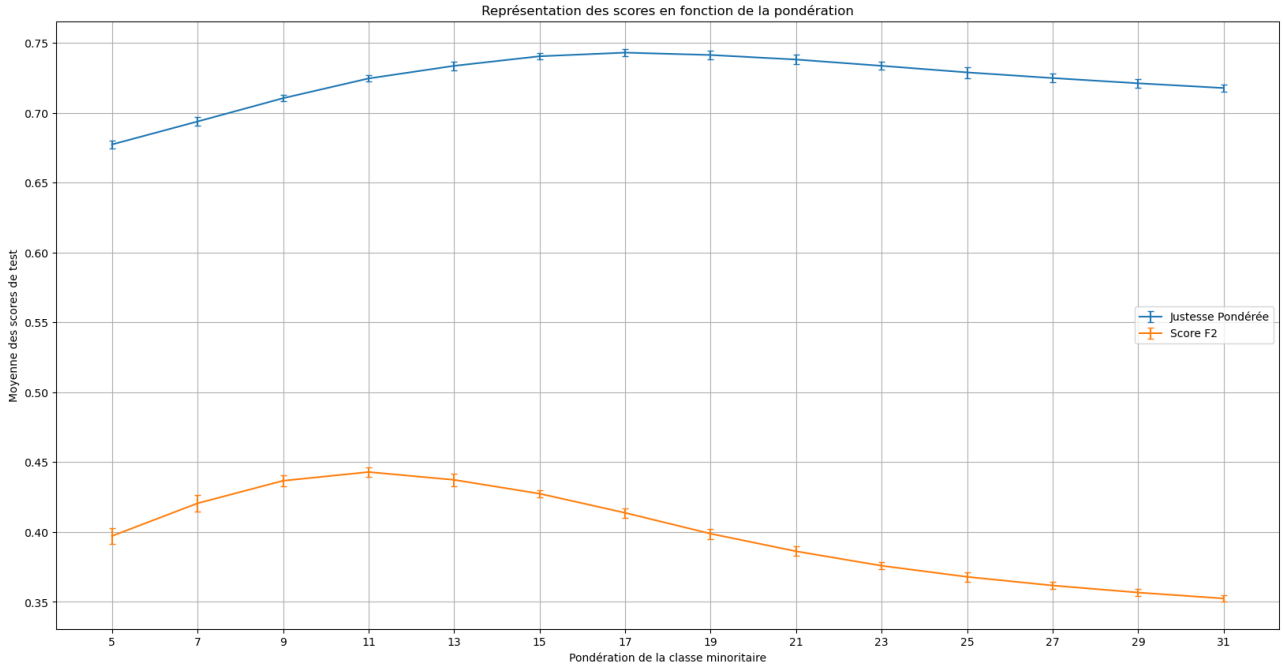


FIGURE 13.2 – Graphique des scores F_2 et de justesse pondérée selon la pondération

une grande étendue des valeurs, le choix de min_child_samples n'a pas d'impact sur la qualité du modèle, ainsi cette variable sera laissée à sa valeur par défaut (1). Nous constatons néanmoins que le meilleur compromis entre les scores est atteint pour $max\_depth = 9$.

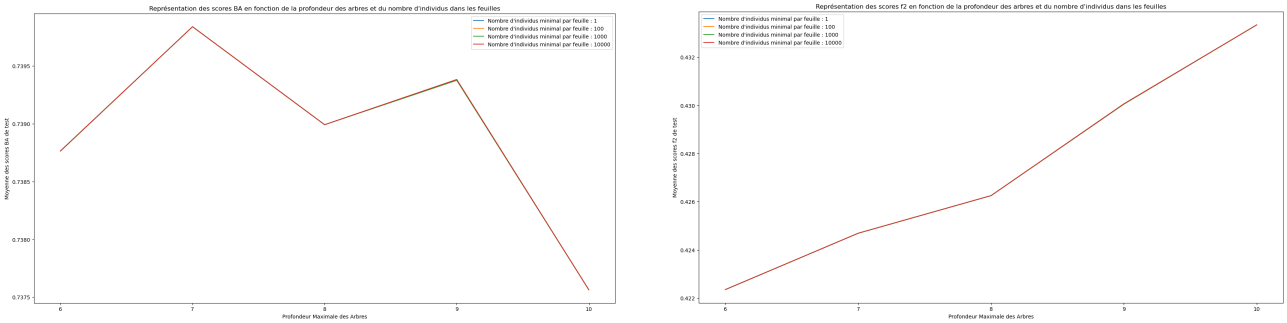


FIGURE 13.3 – Graphique des scores F_2 et de justesse pondérée en fonction de la profondeur et du nombre d'individus par feuille

Nous pouvons donc résumer les paramètres obtenus jusqu'ici :

- iterations = 300 ;
- learning_rate = 0.025 ;
- scale_pos_weight = 15 ;
- max_depth = 9 choisi parmi [6, 7, 8, 9, 10] ;
- min_child_samples = 1 choisi parmi [1, 100, 1000, 10000] ;

Nous allons maintenant nous intéresser aux paramètres colsample_bylevel et subsample.

Nous pouvons noter que le score F_2 et la justesse pondérée sont maximisés pour les mêmes paramètres, ainsi le choix des paramètres est plus simple. Nous utiliserons donc la stratégie MVS (pour Minimum Variance Sampling), en considérant 80% de la base d'entraînement et 60% des colonnes lors de la création des arbres.

Ainsi, nous obtenons les paramètres suivants pour le modèle CatBoost pour le périmètre standard :

- iterations = 300 ;
- learning_rate = 0.025 ;
- scale_pos_weight = 15 ;
- max_depth = 9 ;

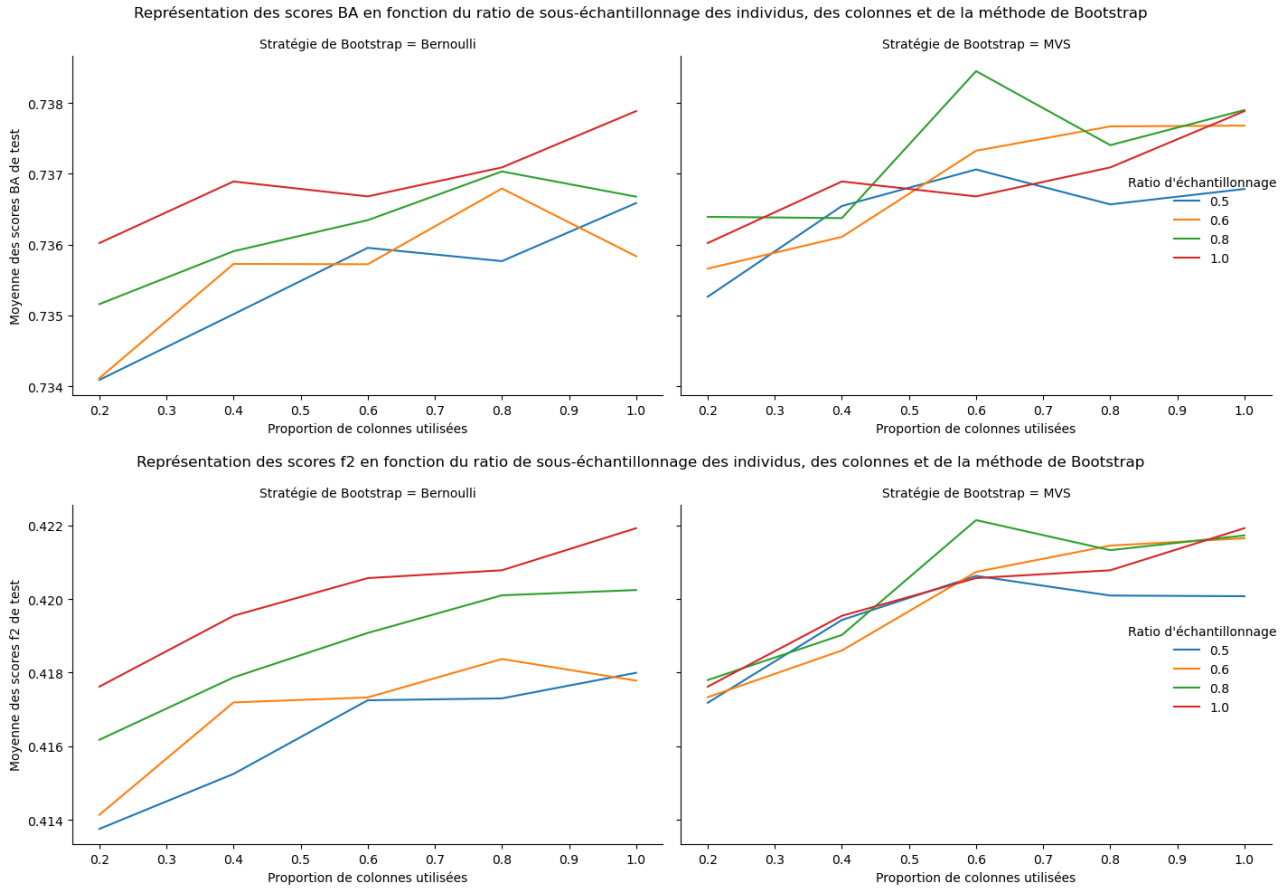


FIGURE 13.4 – Résumé des scores de justesse pondérée et F_2 pour le paramétrage de colsample_bylevel et subsample

- min_child_samples = 1 ;
- colsample_bylevel = 0.6 choisi parmi [0.2, 0.4, 0.6, 0.8, 1] ;
- bootstrap_type = MVS choisi parmi ['Bernoulli', 'MVS'] ;
- subsample = 0.8 choisi parmi [0.5, 0.6, 0.8 ,1].

Nous avons résumé les résultats de la validation-croisée pour ce modèle dans le tableau 13.1.

Score	Split0	Split1	Split2	Split3	Split4	Moyenne	Validation
F2	0.4194	0.4203	0.4220	0.4211	0.4204	0.4206	0.4163
BA	0.7359	0.7361	0.7380	0.7379	0.7369	0.7370	0.7358

TABLEAU 13.1 – Score de la validation croisée par split pour les paramètres sélectionnés

Ainsi, nous obtenons les résultats suivants :

Les résultats présentés en 13.2 montrent que les résultats sont légèrement meilleurs pour le modèle CatBoost que pour les autres modèles créés précédemment. Ainsi, les résultats montrent une amélioration de nos scores et les modèles prédisent de mieux en mieux la résiliation. Pour le périmètre standard, le modèle qui semble optimal pour la modélisation de la résiliation est donc le modèle CatBoost.

	Score F_2	BA
GLM	0.36	0.70
Random Forest	0.41	0.72
CatBoost	0.42	0.74

TABLEAU 13.2 – Comparaison des scores entre CatBoost, Random Forest et GLM

13.2.1 Importance des variables

Nous pouvons nous intéresser à l'importance des variables selon le modèle Catboost. Pour cela, nous utiliserons la méthode du Permutation Importance Importance Plot, comme pour les modèles GLM et Random Forest.

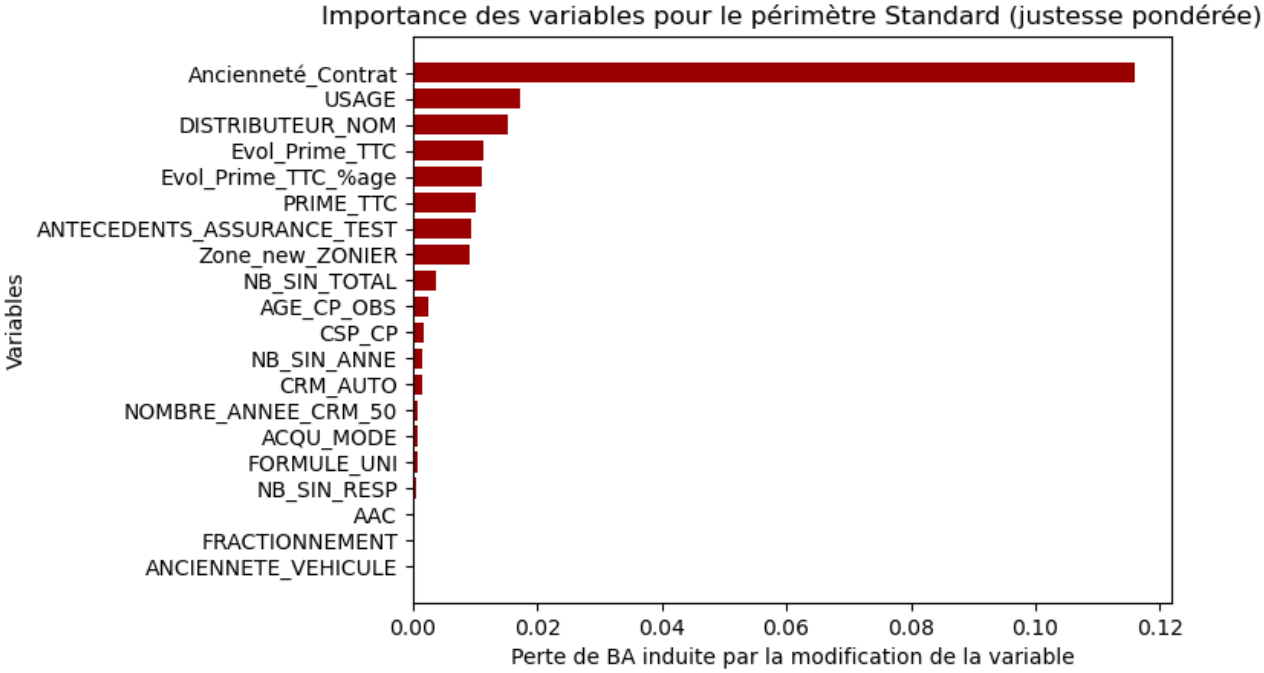


FIGURE 13.5 – Importance des variables selon la justesse pondérée pour le périmètre standard

Sur les graphiques 13.5 et 13.6, nous obtenons des résultats qui sont similaires et donc cohérents avec les résultats que nous avons pu observer pour les autres modèles. Ainsi, la variable la plus importante, pour les deux scores, reste la variable *Ancienneté_Contrat*. Ensuite, les variables liées aux primes sont assez importantes dans le modèle tout comme les variables *USAGE* ou *DISTRIBUTEUR_NOM* qui apparaissaient déjà dans les autres modèles.

Notons tout de même que la variable relative aux antécédents d'assurance semble assez importante d'après le modèle Catboost, comme certaines de ses modalités pour le modèle de forêts aléatoire. Aussi, le zonier semble important dans cette modélisation.

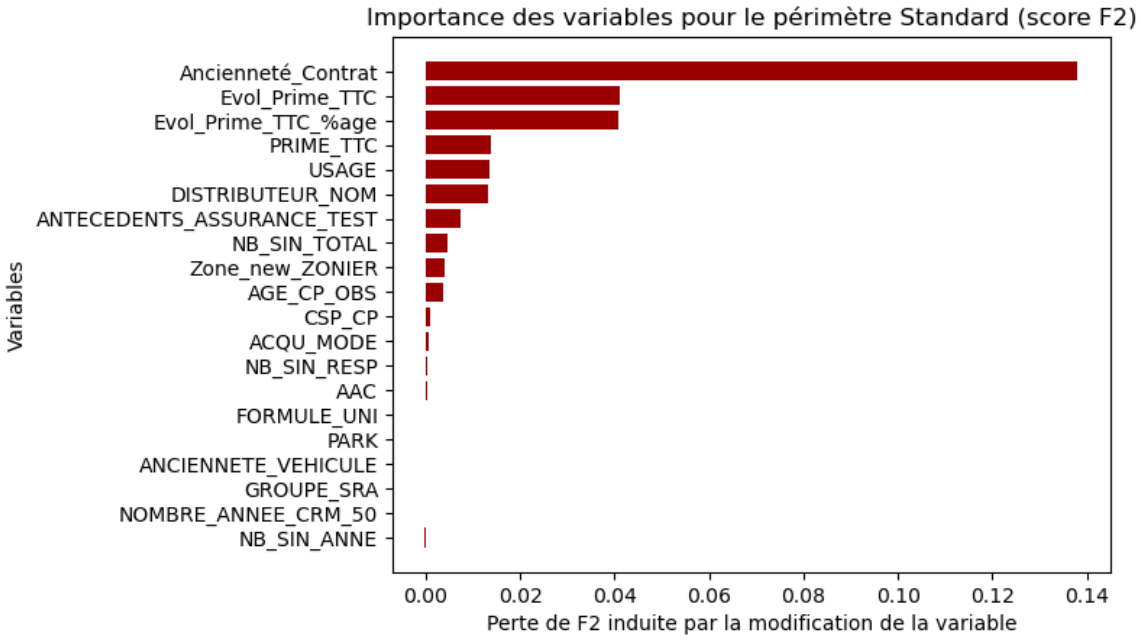


FIGURE 13.6 – Importance des variables selon le score F_2 pour le périmètre standard

13.3 Périmètre malussé

Pour la modélisation du périmètre malussé, nous allons utiliser la même méthode de réglage des paramètres que précédemment. Les graphiques représentant le score F_2 et la justesse pondérée permettant l'ajustement des hyper-paramètres sont disponibles en annexe C. Nous obtenons à la suite de cette étape les paramètres suivants :

- iterations = 200 ;
- learning_rate = 0.025 ;
- scale_pos_weight = 17 ;
- max_depth = 10 ;
- min_child_samples = 10000 ;
- colsample_bylevel = 0.6 ;
- bootstrap_type = MVS ;
- subsample = 0.8.

Nous avons résumé les résultats de la validation-croisée pour ce modèle dans le tableau 13.3.

Score	Split0	Split1	Split2	Split3	Split4	Moyenne	Validation
F2	0.4844	0.4908	0.4872	0.4904	0.4931	0.4892	0.4858
BA	0.7743	0.7795	0.7740	0.7770	0.7820	0.7774	0.7671

TABLEAU 13.3 – Score de la validation croisée par split pour les paramètres sélectionnés

Nous obtenons donc les scores suivants :

	Score F_2	BA
GLM	0.46	0.75
Random Forest	0.48	0.76
CatBoost	0.49	0.77

TABLEAU 13.4 – Comparaison des scores entre Catboost, Random Forest et GLM

Pour ce périmètre aussi, nous observons une amélioration progressive de nos résultats. La résiliation est donc mieux prédite par le modèle Catboost que par les autres modèles. Toutefois, il est important de noter que le réglage des paramètres est beaucoup plus long pour ce modèle que pour les autres, en effet le temps de calcul est plus important et le nombre de paramètre plus important. Ajuster un modèle CatBoost sur une base de données de 300 000 lignes en considérant 1000 itérations peut en effet prendre plusieurs heures, contre quelques dizaines de minutes pour un modèle de forêts aléatoires, par exemple.

13.3.1 Importance des variables

Nous allons maintenant nous intéresser à l'importance des variables pour ce modèle. En comparant les résultats avec ceux des modèles précédents nous pourrons observer d'éventuelles différences entre les modèles vis-à-vis des variables les plus importantes. Rappelons d'abord que les variables de grande importance selon les autres modèles sont les variables liées à la prime, la variable *Ancienneté_ Contrat* ainsi que certaines des variables liées au critère d'aggravation du contrat.

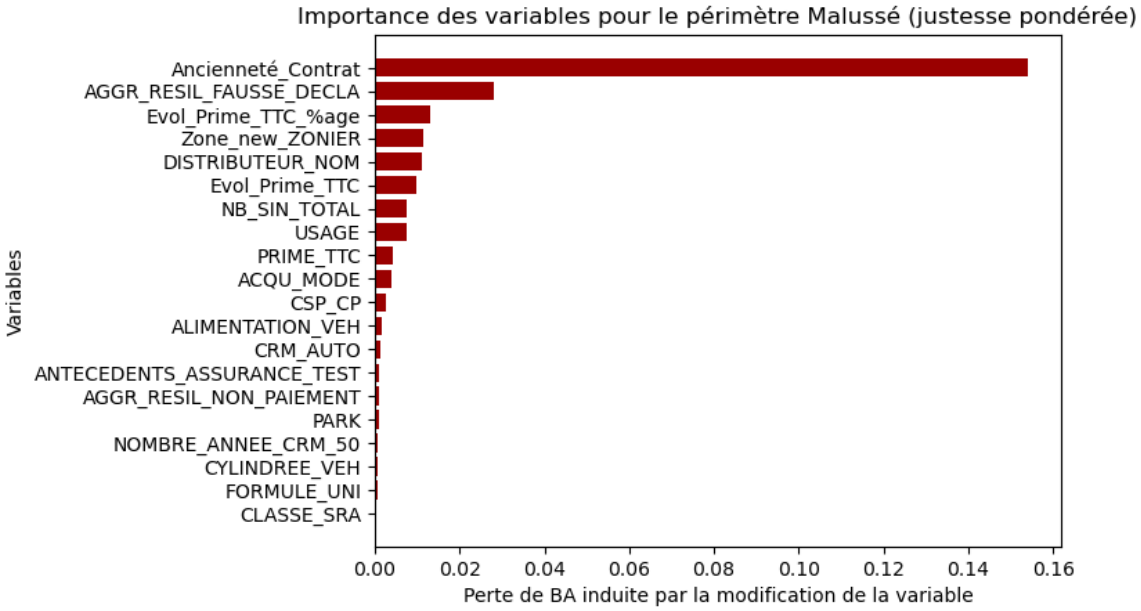


FIGURE 13.7 – Importance des variables selon la justesse pondérée pour le périmètre malussé

Nous observons sur les graphiques 13.7 et 13.8 que la variable *Ancienneté_ Contrat* est, encore une fois, la plus importante pour ce modèle. Les variables liées aux primes restent elles aussi assez importantes. La variable relative au zonier est particulièrement importante dans ce modèle, ce qui n'était pas autant le cas dans les modèles précédents. Nous notons aussi que la variable concernant le nombre de sinistres de l'assuré durant son contrat est importante dans ce modèle.

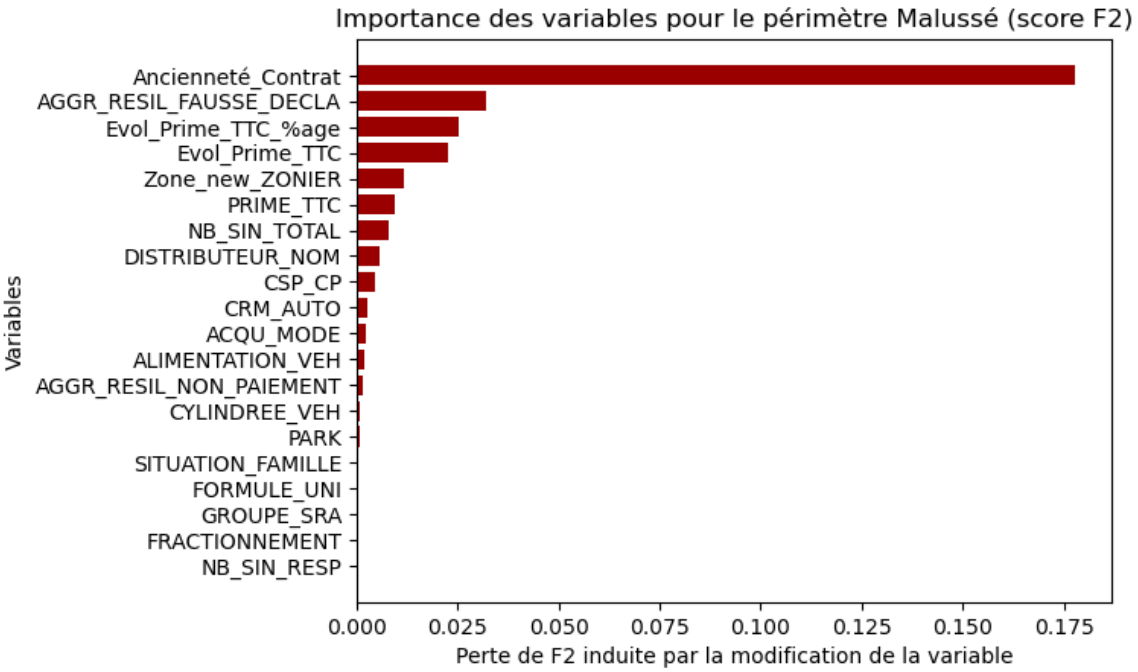


FIGURE 13.8 – Importance des variables selon le score F_2 pour le périmètre malussé

13.4 Comparaison entre les périmètres

Pour résumer la modélisation par CatBoost, nous pouvons noter que nous avons obtenu de meilleurs résultats qu'avec les précédentes méthodes. Les variables les plus importantes obtenues par PFI, sont plus ou moins les mêmes que celles des modèles précédents. Nous retrouvons ainsi, la variable *ancienneté contrat*, les variables liées à la prime ou encore à la variable relative à l'utilisation du véhicule. Nous pouvons en tirer la conclusion que ces variables sont particulièrement intéressantes à prendre en compte lors de la revalorisation des contrats. Toutefois, nous pourrons aussi considérer les variables zonier et antécédents assurance (surtout pour le modèle standard pour cette dernière) dans les prochaines revalorisations car elles sont d'une importance certaines d'après ce modèle.

Chapitre 14

Conclusion sur la modélisation

Par la suite, nous allons nous concentrer sur le *meilleur* modèle obtenu pour chaque périmètre et essayer de mieux le comprendre, à l'aide notamment des valeurs de SHAP. Par la suite, nous utiliserons notre modèle pour tenter de prédire le résultat technique attendu après revalorisation, en fonction des résiliations des assurés.

Pour comparer les résultats, nous pouvons rappeler les tableaux de comparaisons des résultats et représenter les courbes ROC, qui nous permettent de calculer les AUC obtenues pour ces modèles.

Périmètre :	Standard			Malussé		
Mesure :	F_2	BA	AUC	F_2	BA	AUC
GLM	0.36	0.70	0.78	0.46	0.75	0.83
RandomForest	0.41	0.72	0.83	0.48	0.76	0.85
CatBoost	0.42	0.74	0.84	0.49	0.77	0.87

Tableau 14.1 – Comparaison des scores pour les différents modèles

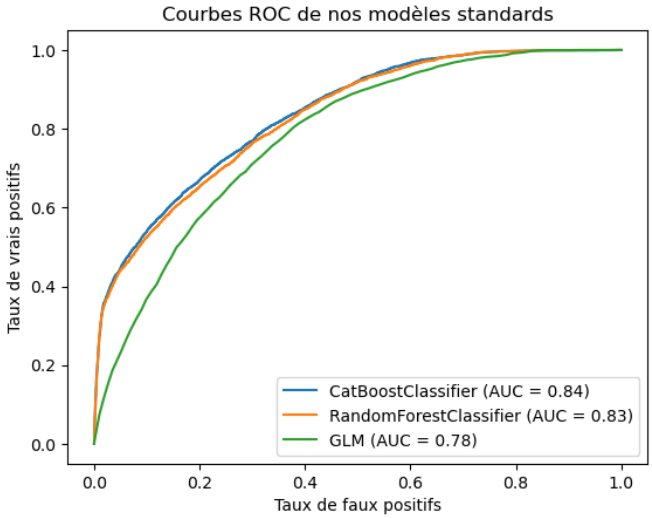


Figure 14.1 – Périmètre standard

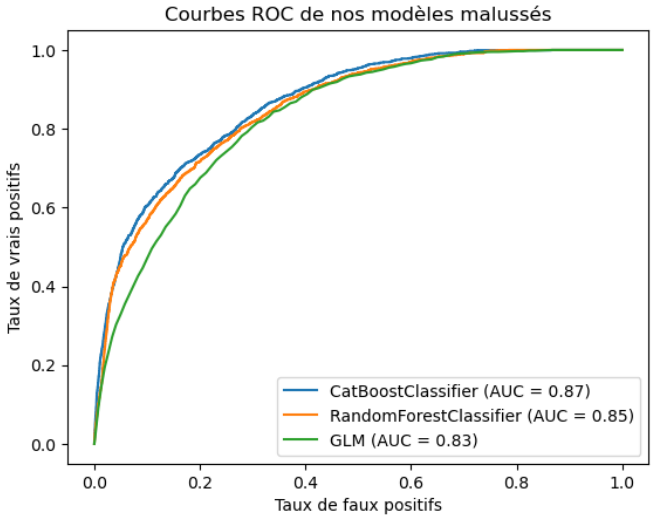


Figure 14.2 – Périmètre malussé

Nous observons donc sur le tableau 14.1 et sur la figure 14.2 que le meilleur modèle est le modèle CatBoost, que ce soit pour le périmètre standard ou pour le périmètre malussé. Le gain en performance du CatBoost se fait au prix du temps de calcul, bien plus important que pour les autres modèles et de la « transparence » du modèle, dans la mesure où ce modèle est un modèle black-box. Nous pouvons néanmoins utiliser certaines méthodes pour obtenir des informations sur les relations déduites par le modèle et qui nous permettrons ainsi de mieux calibrer les politiques de revalorisation.

14.1 Valeurs de SHAP

Pour mettre en évidence les relations apprises par le modèle, nous allons utiliser la méthode des valeurs de SHAP qui permet de quantifier la contribution des variables dans la prédiction du modèle en reposant sur une analyse locale et non globale. Ainsi, cette méthode permet, pour un individu donné, d'expliquer quelle variable a influé sur la sortie du modèle, dans quelle mesure. Nous pouvons par la suite agréger ces résultats pour obtenir une idée de l'impact global des variables dans notre modèle.

Nous allons ici, nous concentrer sur les variables les plus importantes de nos modèles, d'après la méthode de la permutation des variables ainsi que par la méthode des valeurs de SHAP.

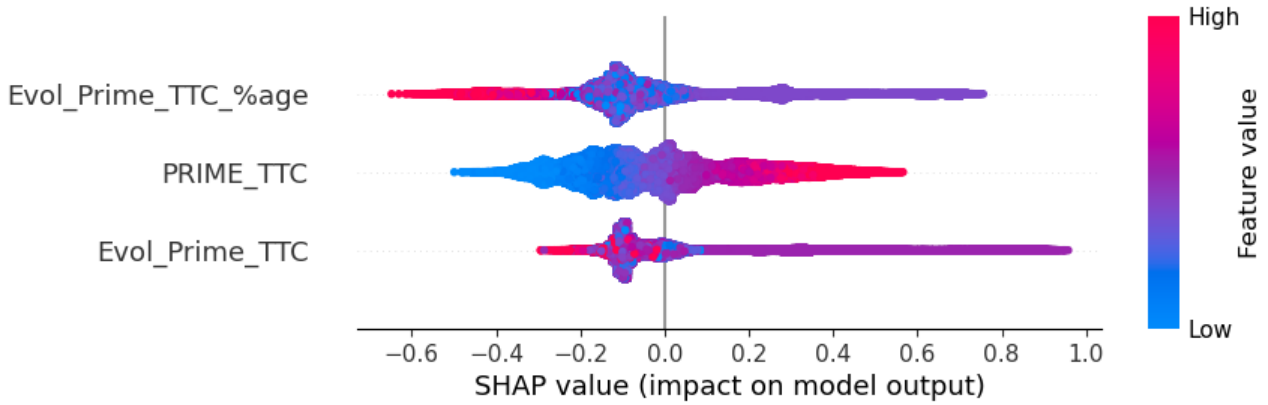


FIGURE 14.3 – Valeurs de SHAP des variables Primes pour le périmètre standard

Le graphique 14.3, montre les valeurs de SHAP des différentes variables de Primes. Les résultats obtenus ne sont pas ceux auxquels nous aurions pu nous attendre dans la mesure où l'on observe que les assurés dont la prime a beaucoup augmenté, que ce soit en valeur absolue ou en pourcentage ont une tendance plus faible à la résiliation. Cela s'explique par le fait qu'une partie de ces augmentations a sans doute lieu suite au changement des conditions du contrat, ainsi un assuré qui changerait de voiture pour en prendre une de gamme équivalente mais plus récente verrait mécaniquement sa prime augmenter sans forcément changer de contrat d'assurance. Cela expliquerait aussi le fait que l'on observe que les augmentations *moyennes* de prime ont un impact très important sur la prédiction du modèle, puisqu'elles contribuent positivement à la sortie du modèle.

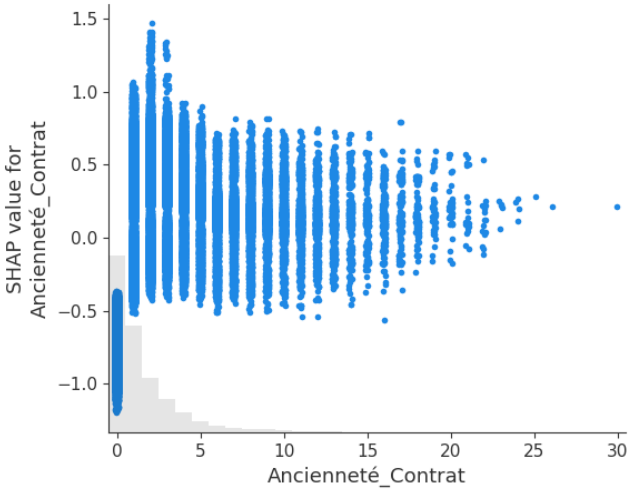


FIGURE 14.4 – Valeurs de SHAP de la variable Ancienneté Contrat pour le périmètre standard

Le graphique 14.4 présente les valeurs de SHAP pour la variable Ancienneté Contrat dont l'importance a été mise en évidence par tous nos modèles. Nous observons principalement que la modalité qui fait le plus décroître la prédiction du modèle est la modalité 0, cela est cohérent car il n'est pas possible de résilier son assurance auto avant le premier anniversaire du contrat. Nous pouvons aussi noter que les assurés sous contrat depuis deux ou trois ans semblent particulièrement sensibles à la résiliation, nous obtenons en effet les plus grandes valeurs de SHAP pour ces modalités. De plus, nous observons une plus faible tendance à la résiliation pour les assurés sous

contrat plus anciens avec des valeurs de SHAP dont la moyenne semble légèrement au-dessus de 0. Il faudra donc faire particulièrement attention à cette donnée lors des prochaines revalorisations, en revalorisant par exemple moins fortement les assurés dont le contrat est "jeune". Les assurés dont le contrat est plus ancien sont sans doute davantage attachés à l'assureur ou au courtier et ont donc moins tendance à résilier. Nous pouvons donc établir que lors de prochaines revalorisations, il sera possible d'adapter les revalorisations selon l'ancienneté en portefeuille des assurés. Il est notamment possible de mettre en place une augmentation plus faible pour les profils rentables dont le contrat a de moins de 3 ans dans l'optique de les conserver en portefeuille. Aussi comme nous avons pu constater que les contrats plus anciens semblent moins sensibles à la résiliation on peut envisager une stratégie de revalorisation plus agressive pour ces derniers.

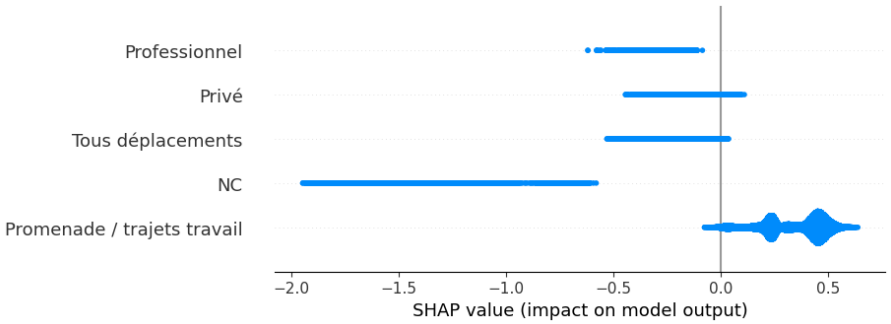


FIGURE 14.5 – Valeurs de SHAP de la variable Usage pour le périmètre standard

Le graphique 14.5 représente les valeurs de SHAP pour chaque modalité de la variable Usage. Cette variable s'est vue accorder une grande importance d'après la méthode SHAP, nous avons donc décidé de la représenter pour observer son impact sur les prédictions du modèle. Ce graphique met en évidence que les assurés qui utilisent leurs voitures pour des promenades ou pour aller au travail semblent être les plus enclins à la résiliation, a contrario ceux pour qui l'information n'est pas connue ont une très faible tendance à la résiliation. Il est plus complexe de prendre en compte cette variable lors de la revalorisation dans la mesure où la modalité qui a le plus d'impact sur la sortie du modèle est "NC" et correspond à un défaut d'information.

Nous allons maintenant, nous intéresser aux valeurs de SHAP du modèle malussé.

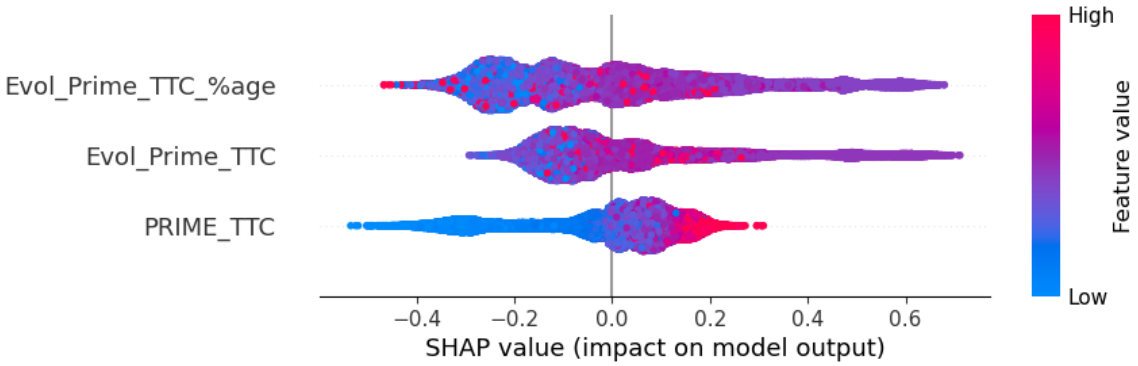


FIGURE 14.6 – Valeurs de SHAP des variables Primes pour le périmètre malussé

Sur le graphique 14.6, nous pouvons voir les valeurs de SHAP des variables liées aux primes. Nous observons une meilleure répartition des valeurs de SHAP pour les plus fortes évolutions de primes, même si le modèle reste biaisé par le grand nombre d'évolutions de primes nulles. Les valeurs de SHAP de la prime semblent quant à elles montrer un lien croissant entre la prime et la probabilité de résiliation.

Nous observons, sur le graphique 14.7, les valeurs de SHAP relatives à la variable *Ancienneté Contrat*. Nous constatons, comme pour le périmètre standard, que les valeurs semblent atteindre leur maxima pour la modalité 2 ou 3 ans. Cette valeur décroît néanmoins après ces modalités, ce qui confirme une forme de fidélisation des clients une fois qu'ils sont restés plus de 3 ans en portefeuille.

Enfin, le graphique 14.8 montre sensiblement les mêmes résultats que pour le modèle standard.

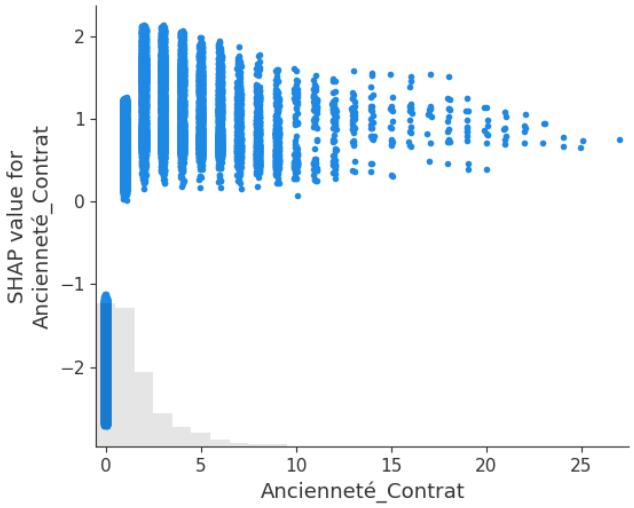


FIGURE 14.7 – Valeurs de SHAP de la variable Ancienneté Contrat pour le périmètre malussé

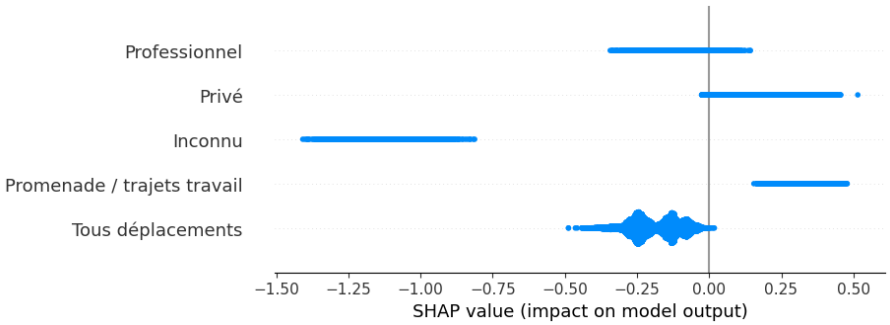


FIGURE 14.8 – Valeurs de SHAP de la variable Usage pour le périmètre malussé

14.2 Mise en application du modèle

Dans cette section, nous allons nous servir de notre modèle pour simuler le comportement des assurés. L'objectif de cette application sera de proposer une revalorisation qui permette de conserver en portefeuille les assurés souhaitant initialement résilier afin d'améliorer le chiffre d'affaires de l'Équité.

Pour réaliser cette simulation, nous avons d'abord choisi de nous restreindre à une police mère [1] au cours de l'année 2022 pour chaque périmètre. Nous avons ensuite déterminé deux seuils de probabilité par police mère, un seuil prudent qui sera relativement plus élevé et un seuil optimiste qui sera relativement plus faible. Ces seuils permettent de classifier en résiliation, ou non, les individus de la base de données selon la prédiction obtenue par le modèle. Après avoir déterminé les seuils de probabilité, nous avons identifié les assurés que nous souhaitons conserver en portefeuille dans l'optique de proposer une augmentation à des profils plutôt rentables. Ainsi, nous avons décidé de proposer un nouveau tarif aux assurés n'ayant pas eu de sinistres sur les 36 derniers mois pour le périmètre standard et sur les 12 derniers mois pour le périmètre malussé. Pour déterminer le tarif le plus intéressant à proposer, c'est-à-dire le tarif maximum permettant de conserver l'assuré en portefeuille, nous avons, pour chaque assuré, modélisé sa probabilité de résiliation selon différentes revalorisations de la prime. Nous pouvons ainsi faire une contre-proposition aux assurés à l'aide des nouveaux taux obtenus.

1. Ces polices mères ont été choisies de façon à représenter correctement le portefeuille total, c'est-à-dire que l'on n'observe pas de différences majeures pour les distributions des variables.

Périmètre standard	Vision prudente	Vision optimiste
Primes 2021	691 900	691 900
Primes théoriques après la revalorisation 2022	688 400	688 400
Primes réelles après la revalorisation2022 (1)	442 600	442 600
Primes espérées après application du modèle (2)	451 300	457 900
Pourcentage de gain ((2-1)/1)	1,96%	3,45%

Tableau 14.2 – Résultats du modèle obtenus sur le périmètre standard

Nous observons sur le tableau 14.2 que nous pouvons espérer près de 2 % de gain en ayant une approche prudente et plus de 3 % en étant plus confiant dans la prédiction du modèle. Ces résultats, s'ils se généralisaient sur l'ensemble du portefeuille permettraient un gain de chiffre d'affaires compris entre 3 et 5 Millions d'€ pour l'année 2022.

Périmètre malussé	Vision prudente	Vision optimiste
Primes 2021	93 600	93 600
Primes théoriques après la revalorisation 2022	90 800	90 800
Primes réelles après la revalorisation 2022 (1)	52 500	52 500
Primes espérées après application du modèle (2)	54 500	56 000
Pourcentage de gain ((2-1)/1)	3,76%	6,74%

Tableau 14.3 – Résultats du modèle obtenus sur le périmètre malussé

Le tableau 14.3 présente les résultats obtenus pour la police mère malussé. Nous y constatons des gains plus importants que pour le périmètre standard. En effet, nous pouvons espérer $3,76$ % de gain avec un seuil de probabilité plutôt prudent et plus de $6,5$ % pour le seuil optimiste. Ces gains correspondraient, s'ils étaient généralisés à l'ensemble du portefeuille malussé à un gain compris entre 2,5M et 5M d'€ pour l'année 2022.

Conclusion

Nous avons donc mis en place, tout au long de ce mémoire, différents modèles permettant de modéliser la résiliation. Pour ce faire, nous avons dû retravailler largement la base de données afin d'ajouter des variables, d'en retravailler d'autres, et de supprimer celles dont les informations n'étaient pas suffisamment renseignées.

Pour le modèle GLM, nous avons également dû segmenter nos variables quantitatives. Face à la trop faible part d'assurés résiliés dans notre base de données, nous avons dû rééchantillonner notre base de données afin d'obtenir des classes équilibrées. Ainsi, nous avons pu réaliser une modélisation de la résiliation. À l'aide des coefficients du GLM, nous avons créé une échelle de scoring des profils selon leur sensibilité à la résiliation. Cela nous a permis d'établir des profils types de l'assuré le plus prompt à la résiliation et celui qui risque le moins de résilier son contrat.

Ensuite, nous avons testé un modèle de forêts aléatoires, ce dernier a obtenu des résultats légèrement meilleurs que le GLM, cependant il est moins facile à interpréter. Il n'est pas, par exemple, possible d'obtenir *facilement* des coefficients pour ce modèle.

Enfin, nous avons entraîné des modèles de boosting, et particulièrement des Catboost. Ces derniers ont obtenu les meilleurs résultats, même si l'amélioration n'est pas très importante, et nous avons pu réaliser une projection du chiffre d'affaires de l'Équité basée sur ces derniers modèles.

Nous avons toutefois rencontré un certain nombre d'écueils au cours de ce mémoire :

- Le déséquilibre des classes qui a nécessité de rééchantillonner la base de données ;
- Les évolutions de primes étaient très concentrées autour de 0, cela a fortement influencé les résultats de nos modèles et a rendu difficile l'interprétation des résultats de notre étude de la sensibilité au prix ;
- Les métriques choisies (à savoir l'exactitude pondérée et le score F_2), présentent des avantages et nous ont permis de sélectionner nos modèles, cependant, les scores étaient relativement proches pour les différents modèles, et l'étude de la sensibilité prix a montré quelques failles. Peut-être qu'un autre choix de métrique aurait permis d'obtenir de meilleurs résultats finaux ;
- Le temps d'entraînement des modèles CatBoost peut dépasser plusieurs heures lors de la recherche de paramètres, ce problème est cependant à relativiser dans la mesure où il est possible d'enregistrer le modèle et de le réappliquer sans avoir à le réentraîner sur l'ensemble des données.

Le choix des paramètres des modèles de machine learning, à l'aide d'une validation-croisée a néanmoins permis d'éviter le sur-apprentissage et nos modèles semblent avoir un bon pouvoir de généralisation. Ainsi, nous pourrons les utiliser sur d'autres partenaires afin de modéliser la résiliation pour ces derniers.

Nous pouvons, aussi, rappeler que les trois modèles ont mis en avant les variables liées aux primes, ce qui justifie notre volonté initiale de modéliser la résiliation par rapport à ces variables. La variable relative à l'ancienneté du contrat a aussi eu un poids très important dans tous nos modèles, cela semble être dû au fait qu'il ne soit pas possible de résilier dans la première année de contrat (pour les motifs que nous avons choisis) et qu'une part relativement importante de notre base de données représentait de jeunes contrats. Les profils de risques faibles et fort que nous avons pu établir grâce aux GLM ainsi que les différents graphiques d'importance des variables permettront sans doute de détailler plus finement les prochaines revalorisations auprès des partenaires. Nous avons ainsi pu établir tout au long de ce mémoire une liste de variables qui semblent importantes à prendre en compte dans la résiliation. Définir les prochaines revalorisations avec les partenaires en fonction de cette liste de variables permettrait possiblement d'augmenter la rétention en portefeuille de profils rentables. Il faudrait néanmoins étudier plus avant ces variables pour déterminer quels sont les assurés les plus sensibles à la résiliation.

Pour prolonger cette étude, nous pourrions utiliser d'autres méthodes de modélisation comme les SVM ou les réseaux de neurones. Nous pourrions aussi ajuster un XGBoost, ce qui permettrait de comparer les résultats avec le modèle CatBoost car ces deux modèles sont basés sur la méthode du boosting. Il serait aussi intéressant de confronter ces résultats à d'autres données par exemple d'un autre partenaire. Nous pourrions ainsi, mettre en application ce mémoire de façon concrète et vérifier s'il a de bonnes capacités de généralisation.

Annexes

Annexe A

Graphiques du coude pour différentes variables

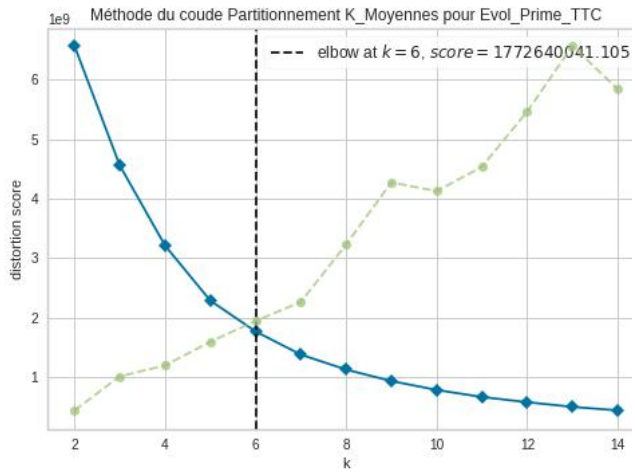


FIGURE A.1 – Graphique du coude pour Evol_Prime_TTC

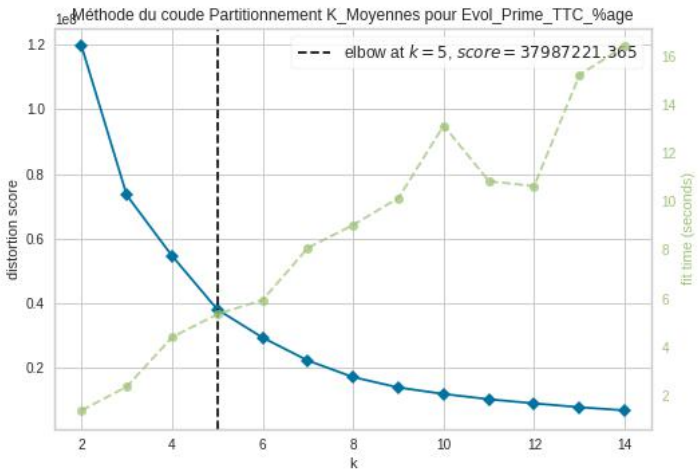


FIGURE A.2 – Graphique du coude pour Evol_Prime_TTC_%age

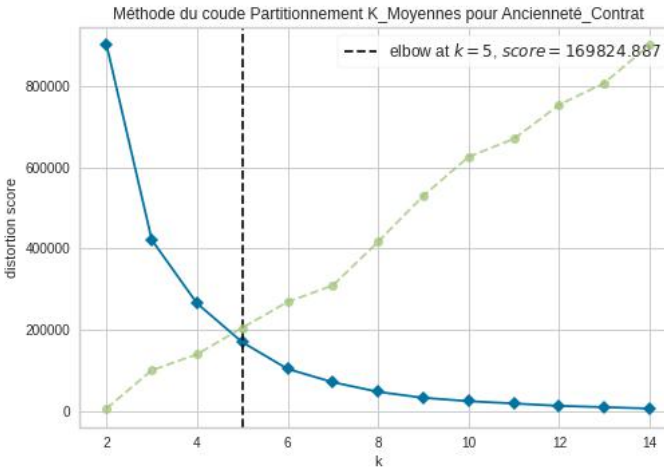


FIGURE A.3 – Graphique du coude pour Ancienneté_Contrat

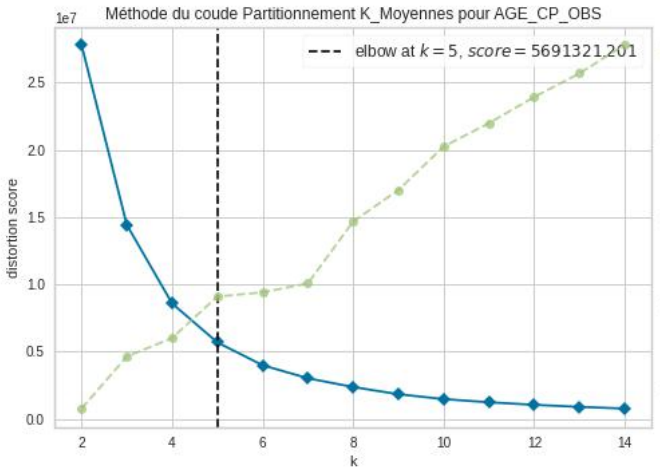


FIGURE A.4 – Graphique du coude pour AGE_CP_OBS

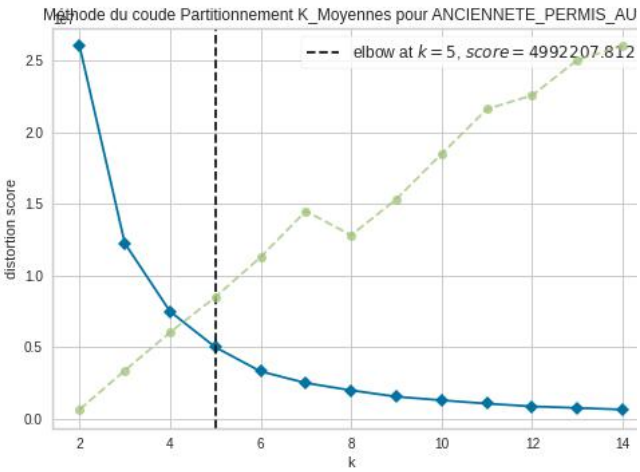


FIGURE A.5 – Graphique du coude pour ANCIENNETE_PERMIS_AUTO

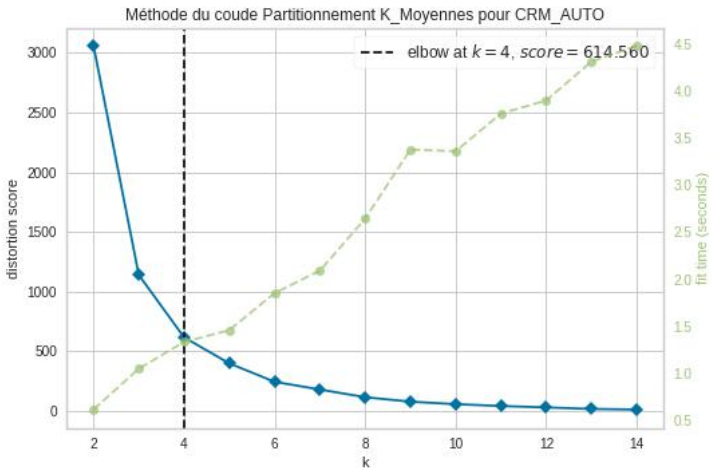


FIGURE A.6 – Graphique du coude pour CRM_AUTO

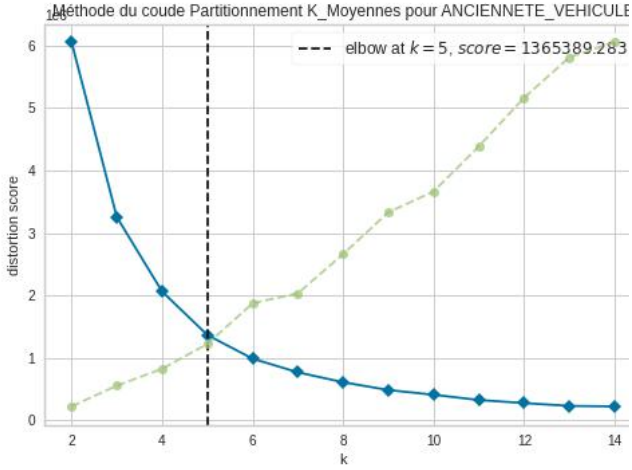


FIGURE A.7 – Graphique du coude pour ANCIENNETE_VEHICULE

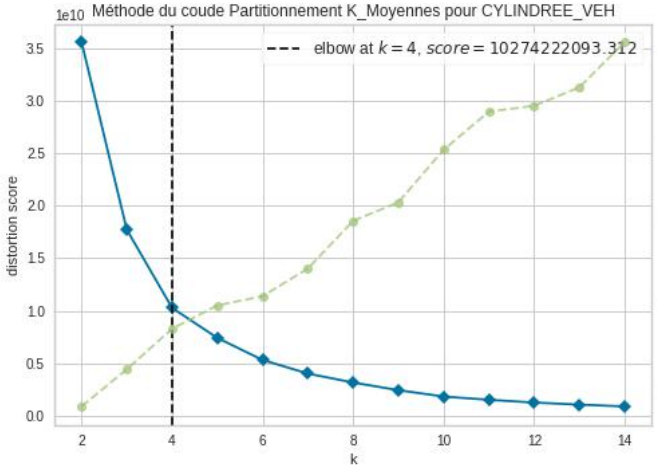


FIGURE A.8 – Graphique du coude pour CYLINDREE_VEH

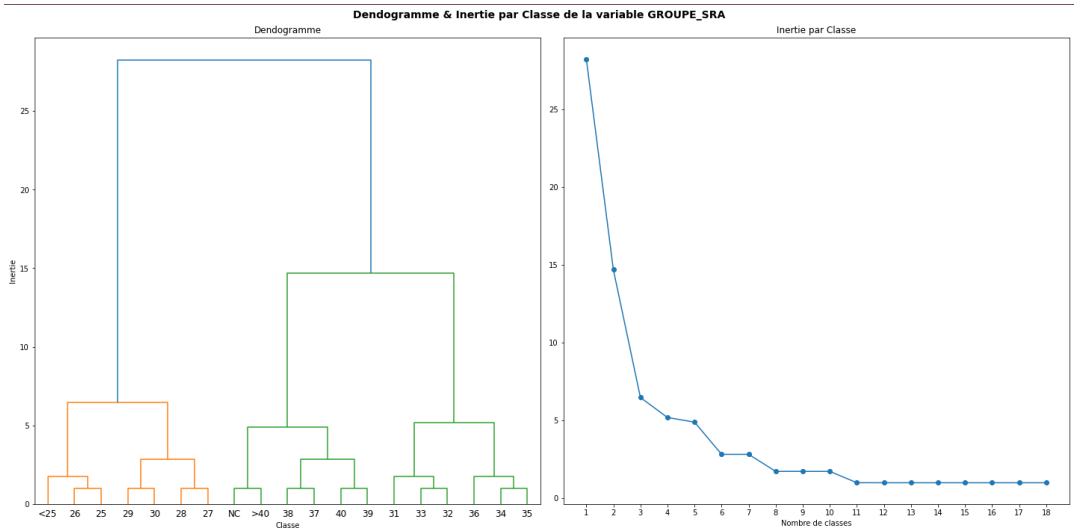


FIGURE A.9 – Classification Ascendante Hiérarchique pour la variable GROUPE_SRA

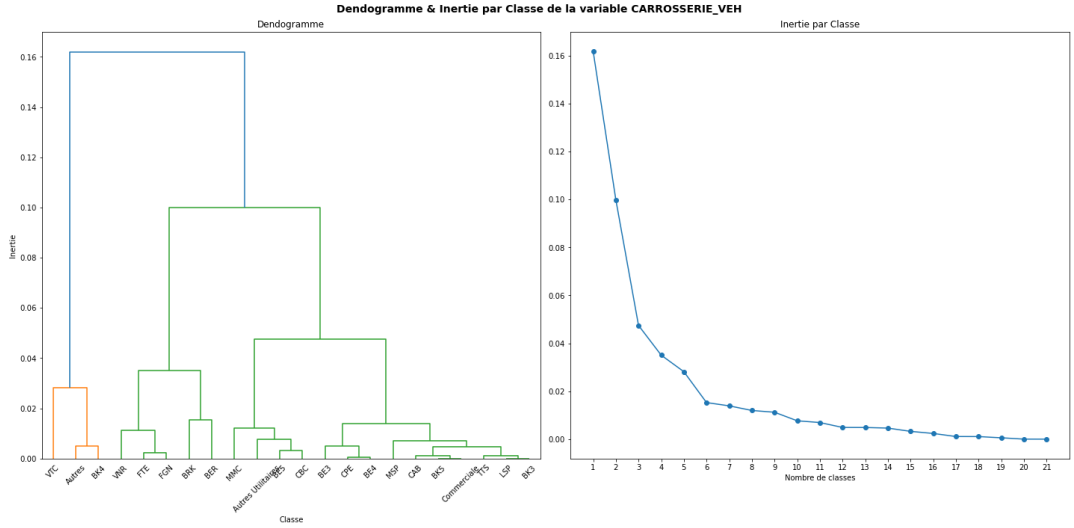


FIGURE A.10 – Classification Ascendante Hiérarchique pour la variable CARROSSERIE_VEH

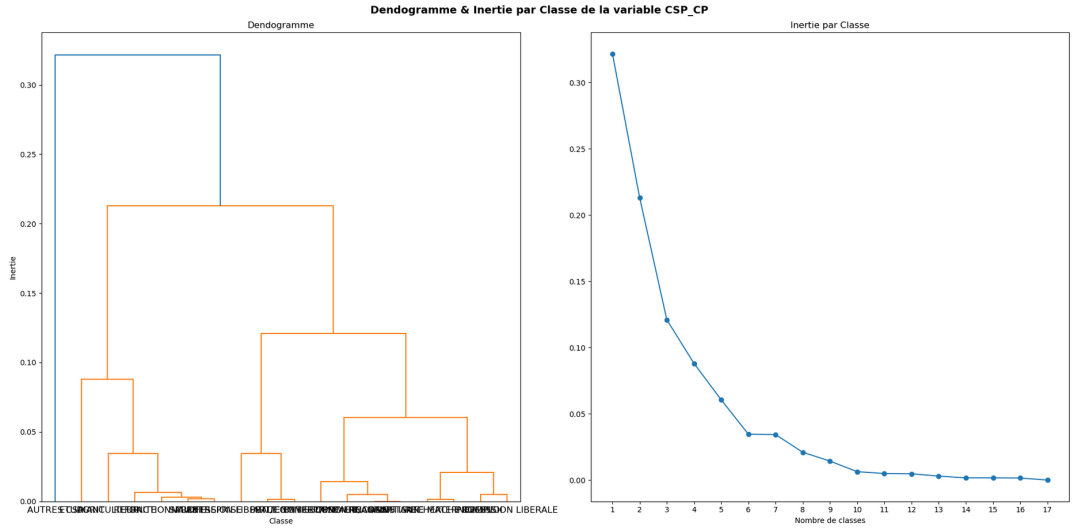


FIGURE A.11 – Classification Ascendante Hiérarchique pour la variable CSP_CP

Annexe B

Informations complémentaires sur la régression logistique

		SMOTE		ROSE		OVUN	
Ratio réél	Seuil	Score F2	Balanced acc	Score F2	Balanced acc	Score F2	Balanced acc
0.06	0.3	0.04977772	0.51871211	0.0453607	0.51671991	0.04762203	0.51785167
0.1	0.3	0.18569031	0.56997783	0.17331992	0.56690197	0.17281748	0.56651007
0.15	0.3	0.26060403	0.6065678	0.30634387	0.63595808	0.3099051	0.63832122
0.2	0.3	0.2859957	0.62399591	0.35952756	0.68831729	0.35850906	0.68717654
0.25	0.3	0.30453031	0.64048094	0.36189545	0.70665955	0.36367896	0.70751545
0.3	0.3	0.31040342	0.64807977	0.36062334	0.71700545	0.36065835	0.71697494
0.35	0.3	0.31447054	0.65452886	0.35148815	0.71370809	0.35112272	0.71336158
0.4	0.3	0.31644764	0.65912712	0.34266808	0.70751203	0.34200385	0.70674312
0.45	0.3	0.31609402	0.66125673	0.3354339	0.70153808	0.33497101	0.70056258
0.5	0.3	0.31458423	0.66185333	0.32787769	0.69380373	0.32806488	0.69402211
0.06	0.5	0.00160195	0.50054354	0.00128148	0.5004017	0.00192246	0.50068692
0.1	0.5	0.04194053	0.51528596	0.02568493	0.50964649	0.02723243	0.51015988
0.15	0.5	0.11780932	0.54203073	0.07750276	0.52883577	0.08214744	0.53062485
0.2	0.5	0.17016449	0.56097795	0.15880244	0.56086897	0.15993409	0.561307
0.25	0.5	0.21057265	0.57828954	0.23887809	0.59646935	0.24358655	0.59903524
0.3	0.5	0.24191867	0.5943655	0.32051018	0.64594291	0.31734925	0.64384483
0.35	0.5	0.26229633	0.60669731	0.35518849	0.68003954	0.35301626	0.67863755
0.4	0.5	0.28095005	0.62022889	0.36448997	0.70063005	0.36384011	0.70025376
0.45	0.5	0.29301058	0.63054552	0.36188959	0.70969917	0.36195644	0.70934259
0.5	0.5	0.29898373	0.63686212	0.35824632	0.71486542	0.35909111	0.71592971
0.06	0.7	#DIV/0!	0.49999246	#DIV/0!	0.49998493	#DIV/0!	0.49999246
0.1	0.7	0.00352248	0.50129829	0.00064107	0.50018878	0.00064107	0.50018878
0.15	0.7	0.02180646	0.50761696	0.00703415	0.50261166	0.00703505	0.50262673
0.2	0.7	0.05272192	0.51790806	0.02286295	0.50858973	0.02286731	0.50861234
0.25	0.7	0.09586069	0.5333561	0.04824561	0.51811581	0.04944298	0.51852352
0.3	0.7	0.12574417	0.5434251	0.08964386	0.53354623	0.08759657	0.5327084
0.35	0.7	0.15771882	0.55521735	0.13570303	0.55152176	0.13159434	0.54974794
0.4	0.7	0.1841208	0.56544952	0.19623699	0.57657598	0.19187053	0.57454518
0.45	0.7	0.21202132	0.57814426	0.26064313	0.60794896	0.25580613	0.6052761
0.5	0.7	0.23320075	0.58879513	0.3135334	0.64122509	0.31028919	0.63907407

FIGURE B.1 – Tableau résumant le score F_2 et la justesse pondérée selon la méthode, le seuil de rééchantillonnage et le seuil de probabilité sur le périmètre Standard

		SMOTE		ROSE		OVUN	
Ratio réél	Seuil	Score F2	Balanced acc	Score F2	Balanced acc	Score F2	Balanced acc
0.06	0.3	0.23157082	0.5933769	0.24770759	0.6007207	0.22765381	0.59162413
0.1	0.3	0.31428239	0.6319678	0.3260616	0.63828906	0.32341832	0.63699784
0.15	0.3	0.37042667	0.66421112	0.41249499	0.69058583	0.40720632	0.68786727
0.2	0.3	0.38665448	0.67712253	0.45778835	0.73156278	0.45823369	0.73141424
0.25	0.3	0.39824131	0.68808685	0.46135033	0.74688096	0.46011132	0.74570823
0.3	0.3	0.40440013	0.69573372	0.4605524	0.75825897	0.45728677	0.75486011
0.35	0.3	0.40369869	0.69771374	0.44530494	0.75280072	0.44608434	0.75406667
0.4	0.3	0.40395359	0.70044451	0.43250181	0.74740649	0.42920354	0.74459388
0.45	0.3	0.40477497	0.70353408	0.41417192	0.73268966	0.41886457	0.7380176
0.5	0.3	0.40255633	0.7037661	0.40228747	0.72215187	0.4013106	0.72128491
0.06	0.5	0.07855239	0.52990077	0.09663515	0.53696325	0.07241282	0.52746126
0.1	0.5	0.15509601	0.56016738	0.16668898	0.56530245	0.16925341	0.56644121
0.15	0.5	0.23094255	0.59157026	0.24622414	0.59983323	0.25343588	0.60308273
0.2	0.5	0.28435469	0.61586633	0.32358835	0.63707449	0.31827881	0.63434102
0.25	0.5	0.31324518	0.63025023	0.37286855	0.66458049	0.37311816	0.6646619
0.3	0.5	0.33207316	0.6406979	0.41398479	0.69269409	0.41946799	0.69623838
0.35	0.5	0.34286815	0.64728515	0.44625465	0.7200095	0.44693967	0.72088744
0.4	0.5	0.35953101	0.65862029	0.46156295	0.7401951	0.45896371	0.73954462
0.45	0.5	0.37377504	0.66949388	0.46131702	0.74982366	0.45889721	0.74770814
0.5	0.5	0.38207897	0.67692986	0.4551495	0.75257866	0.45761143	0.75540141
0.06	0.7	0.00506806	0.50196615	0.00723066	0.5027082	0.00434405	0.50165479
0.1	0.7	0.04345348	0.51636849	0.02723624	0.51012471	0.03219805	0.51204818
0.15	0.7	0.09832433	0.53727854	0.10120616	0.53869469	0.09764543	0.53698852
0.2	0.7	0.14043065	0.55298869	0.16923077	0.56641987	0.16310914	0.56386103
0.25	0.7	0.18642325	0.57147497	0.21262718	0.58507374	0.21430433	0.58579445
0.3	0.7	0.22592957	0.58810966	0.25991964	0.60611493	0.26190774	0.60692492
0.35	0.7	0.26079773	0.60395093	0.3034491	0.62683128	0.30283041	0.62646854
0.4	0.7	0.27627729	0.61041658	0.34340348	0.64755861	0.35193903	0.65230484
0.45	0.7	0.28922046	0.61637097	0.38816134	0.67419538	0.38634463	0.67316724
0.5	0.7	0.31120761	0.62820639	0.41730843	0.69543936	0.41454266	0.69369052

FIGURE B.2 – Tableau résumant le score F_2 et la justesse pondérée selon la méthode, le seuil de rééchantillonnage et le seuil de probabilité sur le périmètre Malussé

Échelles de score

Modalité	Score
ClasseAGE_CP_OBS[18.0 ;28.0[	1.88
ClasseAGE_CP_OBS[28.0 ;35.0[	1.15
ClasseAGE_CP_OBS[35.0 ;41.0[	0.46
ClasseAGE_CP_OBS[41.0 ;46.0[	0.23
ClasseAGE_CP_OBS[46.0 ;51.0[	0.35
ClasseAGE_CP_OBS[51.0 ;57.0[	0.05
ClasseAGE_CP_OBS[57.0 ;67.0[	0.26
ClasseAGE_CP_OBS>=67	0
ClasseAncienneté_Contrat[0.0 ;1.0[	0
ClasseAncienneté_Contrat[1.0 ;2.0[	5.5
ClasseAncienneté_Contrat[2.0 ;3.0[	6.09
ClasseAncienneté_Contrat[3.0 ;5.0[	5.81
ClasseAncienneté_Contrat[5.0 ;8.0[	4.83
ClasseAncienneté_Contrat[8.0 ;12.0[	4.05
ClasseAncienneté_Contrat[12.0 ;16.0[	4.15
ClasseAncienneté_Contrat>=16	3.58
ClasseCRM_AUTO[0.50 ;0.56[	0.28
ClasseCRM_AUTO[0.56 ;0.66[	0.44

Modalité	Score
ClasseCRM_AUTO[0.66 ;0.75[	0.13
ClasseCRM_AUTO[0.75 ;0.83[	0
ClasseCRM_AUTO[0.83 ;0.93[	0.61
ClasseCRM_AUTO>=0.93	1.51
ClasseEvol_Prime_TTC<-664.91	0
ClasseEvol_Prime_TTC[-664.91 ;-220.06[	0.02
ClasseEvol_Prime_TTC[-220.06 ;54.26[	0.09
ClasseEvol_Prime_TTC[-54.26 ;44.06[	0.38
ClasseEvol_Prime_TTC[44.06 ;164.25[	2.31
ClasseEvol_Prime_TTC[164.25 ;331.72[	7.85
ClasseEvol_Prime_TTC[331.72 ;563.86[	10.1
ClasseEvol_Prime_TTC[563.86 ;915.65[	7.1
ClasseEvol_Prime_TTC[915.65 ;1548.51[	2.95
ClasseEvol_Prime_TTC>=1548.51	0.09
ClasseEvol_Prime_TTC_%age<-34.2	0
ClasseEvol_Prime_TTC_%age[-34.27 ;-15.89[	0.33
ClasseEvol_Prime_TTC_%age[-15.89 ;-4.34[	0.7
ClasseEvol_Prime_TTC_%age[-4.34 ;12.04[	9.89
ClasseEvol_Prime_TTC_%age[12.04 ;42.25[	7.67
ClasseEvol_Prime_TTC_%age[42.25 ;91.5[	7.21
ClasseEvol_Prime_TTC_%age>=91.5	4.73
ClassePRIME_TTC[68.78 ;416.89[	0
ClassePRIME_TTC[416.89 ;601.08[	1.01
ClassePRIME_TTC[601.08 ;794.64[	1.59
ClassePRIME_TTC[794.64 ;1026.26[	3.11
ClassePRIME_TTC[1026.26 ;1033.17[	4.27
ClassePRIME_TTC[1333.17 ;1800.48[	8.17
ClassePRIME_TTC[1800.48 ;2712.18[	10.81
ClassePRIME_TTC>=2712.18	12.8
ClusterCARROSSERIE_VEH0	3.01
ClusterCARROSSERIE_VEH1	0
ClusterCARROSSERIE_VEH2	2.82
ClusterCARROSSERIE_VEH3	5.45
ClusterCARROSSERIE_VEH4	2.51
ClusterCARROSSERIE_VEH5	2.43
ClusterCARROSSERIE_VEH6	2.91
ClusterCARROSSERIE_VEH7	6.13
ClusterCARROSSERIE_VEH8	0.74
ClusterCARROSSERIE_VEH9	3.19

Modalité	Score
ClusterCSP_CP0	8.52
ClusterCSP_CP1	9.18
ClusterCSP_CP2	8.86
ClusterCSP_CP3	8.95
ClusterCSP_CP4	8.03
ClusterCSP_CP5	14.33
ClusterCSP_CP6	9.85
ClusterCSP_CP7	0
ClusterCSP_CP8	11.89
DISTRIBUTEUR_NOM6	3.06
DISTRIBUTEUR_NOM5	0
DISTRIBUTEUR_NOM0	6.82
DISTRIBUTEUR_NOM3	7.86
DISTRIBUTEUR_NOM1	7.53
DISTRIBUTEUR_NOM2	7.05
DISTRIBUTEUR_NOM4	8.44
FORMULE_UNIRC	0.06
FORMULE_UNIRC+BDG+VI	0.21
FORMULE_UNIRC+BDG+VI+DTA	0
NB_SIN_ANNE0.0	3.39
NB_SIN_ANNE1.0	2.4
NB_SIN_ANNE2.0	1.4
NB_SIN_ANNE>=3	0
NB_SIN_NON_RESP0.0	1.51
NB_SIN_NON_RESP1.0	0.89
NB_SIN_NON_RESP2.0	0.78
NB_SIN_NON_RESP>=3	0
NB_SIN_TOTAL0.0	2.64
NB_SIN_TOTAL1.0	1.7
NB_SIN_TOTAL2.0	0.22
NB_SIN_TOTAL>=3	0
PARKGarage/Parking Clos	9.23
PARKParkingCollectif	9.1
PARKVoiePublique	9.27
PARKZ.INCONNU	0
SEXE_CPF	0.19
SEXE_CPM	0
SITUATION_FAMILLECélibataire	0.16
SITUATION_FAMILLEConcubin	0.36

Modalité	Score
SITUATION_FAMILLEDivorcé	0.48
SITUATION_FAMILLEMarié	0.47
SITUATION_FAMILLENC	1.95
SITUATION_FAMILLEPacsé	1.11
SITUATION_FAMILLEVeuf	0
USAGENC	0
USAGEPrivé	9.66
USAGEProfessionnel	6.79
USAGEPromenade/trajetstravail	8.48
USAGETousdéplacements	8.52

TABLEAU B.1 – Scores pour le périmètre Standard

Modalité	Score
ClasseAGE_CP_OBS[18.0 ;28.0[	1.88
ClasseAGE_CP_OBS[28.0 ;35.0[	1.15
ClasseAGE_CP_OBS[35.0 ;41.0[	0.46
ClasseAGE_CP_OBS[41.0 ;46.0[	0.23
ClasseAGE_CP_OBS[46.0 ;51.0[	0.35
ClasseAGE_CP_OBS[51.0 ;57.0[	0.05
ClasseAGE_CP_OBS[57.0 ;67.0[	0.26
ClasseAGE_CP_OBS>=67	0
ClasseAncienneté_Contrat[0.0 ;1.0[	0
ClasseAncienneté_Contrat[1.0 ;2.0[	5.5
ClasseAncienneté_Contrat[2.0 ;3.0[	6.09
ClasseAncienneté_Contrat[3.0 ;5.0[	5.81
ClasseAncienneté_Contrat[5.0 ;8.0[	4.83
ClasseAncienneté_Contrat[8.0 ;12.0[	4.05
ClasseAncienneté_Contrat[12.0 ;16.0[	4.15
ClasseAncienneté_Contrat>=16	3.58
ClasseCRM_AUTO[0.50 ;0.56[	0.28
ClasseCRM_AUTO[0.56 ;0.66[	0.44
ClasseCRM_AUTO[0.66 ;0.75[	0.13
ClasseCRM_AUTO[0.75 ;0.83[	0
ClasseCRM_AUTO[0.83 ;0.93[	0.61
ClasseCRM_AUTO>=0.93	1.51
ClasseEvol_Prime_TTC<-664.91	0
ClasseEvol_Prime_TTC[-664.91 ;-220.06[	0.02
ClasseEvol_Prime_TTC[-220.06 ;54.26[	0.09

Modalité	Score
ClasseEvol_Prime_TTC[-54.26 ;44.06[	0.38
ClasseEvol_Prime_TTC[44.06 ;164.25[	2.31
ClasseEvol_Prime_TTC[164.25 ;331.72[	7.85
ClasseEvol_Prime_TTC[331.72 ;563.86[	10.1
ClasseEvol_Prime_TTC[563.86 ;915.65[	7.1
ClasseEvol_Prime_TTC[915.65 ;1548.51[	2.95
ClasseEvol_Prime_TTC>=1548.51	0.09
ClasseEvol_Prime_TTC_%age<-34.2	0
ClasseEvol_Prime_TTC_%age[-34.27 ;-15.89[	0.33
ClasseEvol_Prime_TTC_%age[-15.89 ;-4.34[	0.7
ClasseEvol_Prime_TTC_%age[-4.34 ;12.04[	7.27
ClasseEvol_Prime_TTC_%age[12.04 ;42.25[	9.89
ClasseEvol_Prime_TTC_%age[42.25 ;91.5[	7.21
ClasseEvol_Prime_TTC_%age>=91.5	4.73
ClassePRIME_TTC[68.78 ;416.89[	0
ClassePRIME_TTC[416.89 ;601.08[	1.01
ClassePRIME_TTC[601.08 ;794.64[	1.59
ClassePRIME_TTC[794.64 ;1026.26[	3.11
ClassePRIME_TTC[1026.26 ;1033.17[	4.27
ClassePRIME_TTC[1333.17 ;1800.48[	8.17
ClassePRIME_TTC[1800.48 ;2712.18[	10.81
ClassePRIME_TTC>=2712.18	12.8
ClusterCARROSSERIE_VEH0	3.01
ClusterCARROSSERIE_VEH1	0
ClusterCARROSSERIE_VEH2	2.82
ClusterCARROSSERIE_VEH3	5.45
ClusterCARROSSERIE_VEH4	2.51
ClusterCARROSSERIE_VEH5	2.43
ClusterCARROSSERIE_VEH6	2.91
ClusterCARROSSERIE_VEH7	6.13
ClusterCARROSSERIE_VEH8	0.74
ClusterCARROSSERIE_VEH9	3.19
ClusterCSP_CP0	8.52
ClusterCSP_CP1	9.18
ClusterCSP_CP2	8.86
ClusterCSP_CP3	8.95
ClusterCSP_CP4	8.03
ClusterCSP_CP5	14.33
ClusterCSP_CP6	9.85

Modalité	Score
ClusterCSP_CP7	0
ClusterCSP_CP8	11.89
DISTRIBUTEUR_NOM6	3.06
DISTRIBUTEUR_NOM5	0
DISTRIBUTEUR_NOM0	6.82
DISTRIBUTEUR_NOM3	7.86
DISTRIBUTEUR_NOM1	7.53
DISTRIBUTEUR_NOM2	7.05
DISTRIBUTEUR_NOM4	8.44
FORMULE_UNIRC	0.06
FORMULE_UNIRC+BDG+VI	0.21
FORMULE_UNIRC+BDG+VI+DTA	0
NB_SIN_ANNE0.0	3.39
NB_SIN_ANNE1.0	2.4
NB_SIN_ANNE2.0	1.4
NB_SIN_ANNE>=3	0
NB_SIN_NON_RESP0.0	1.51
NB_SIN_NON_RESP1.0	0.89
NB_SIN_NON_RESP2.0	0.78
NB_SIN_NON_RESP>=3	0
NB_SIN_TOTAL0.0	2.64
NB_SIN_TOTAL1.0	1.7
NB_SIN_TOTAL2.0	0.22
NB_SIN_TOTAL>=3	0
PARKGarage/Parking Clos	9.23
PARKParkingCollectif	9.1
PARKVoiePublique	9.27
PARKZ.INCONNU	0
SEXE_CPF	0.19
SEXE_CPM	0
SITUATION_FAMILLECélibataire	0.16
SITUATION_FAMILLEConcubin	0.36
SITUATION_FAMILLEDivorcé	0.48
SITUATION_FAMILLEMarié	0.47
SITUATION_FAMILLENC	1.95
SITUATION_FAMILLEPacsé	1.11
SITUATION_FAMILLEVeuf	0
USAGENC	0
USAGEPrivé	9.66

Modalité	Score
USAGEProfessionnel	6.79
USAGEPromenade/trajetstravail	8.48
USAGETousdéplacements	8.52

Tableau B.2 – Scores pour le périmètre Malussé

Annexe C

Informations complémentaires sur la régression logistique

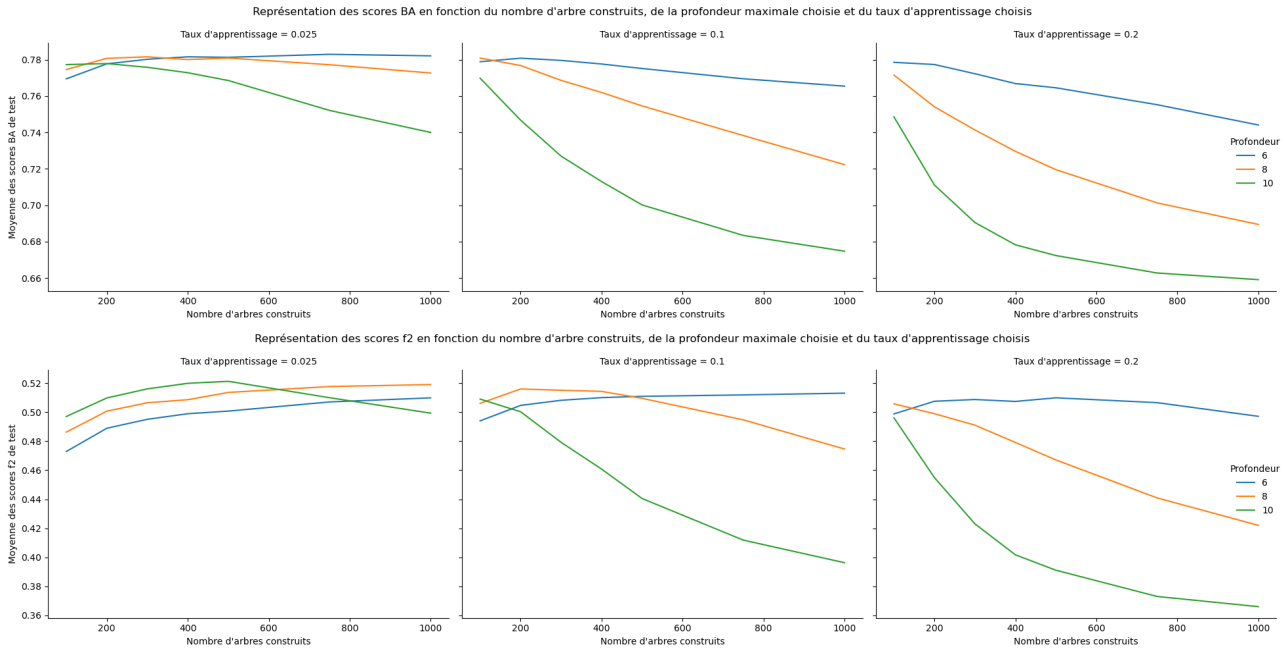


FIGURE C.1 – Résumé des scores de justesse pondérée et F_2 pour le paramétrage d'iterations, learning_rate et max_depth

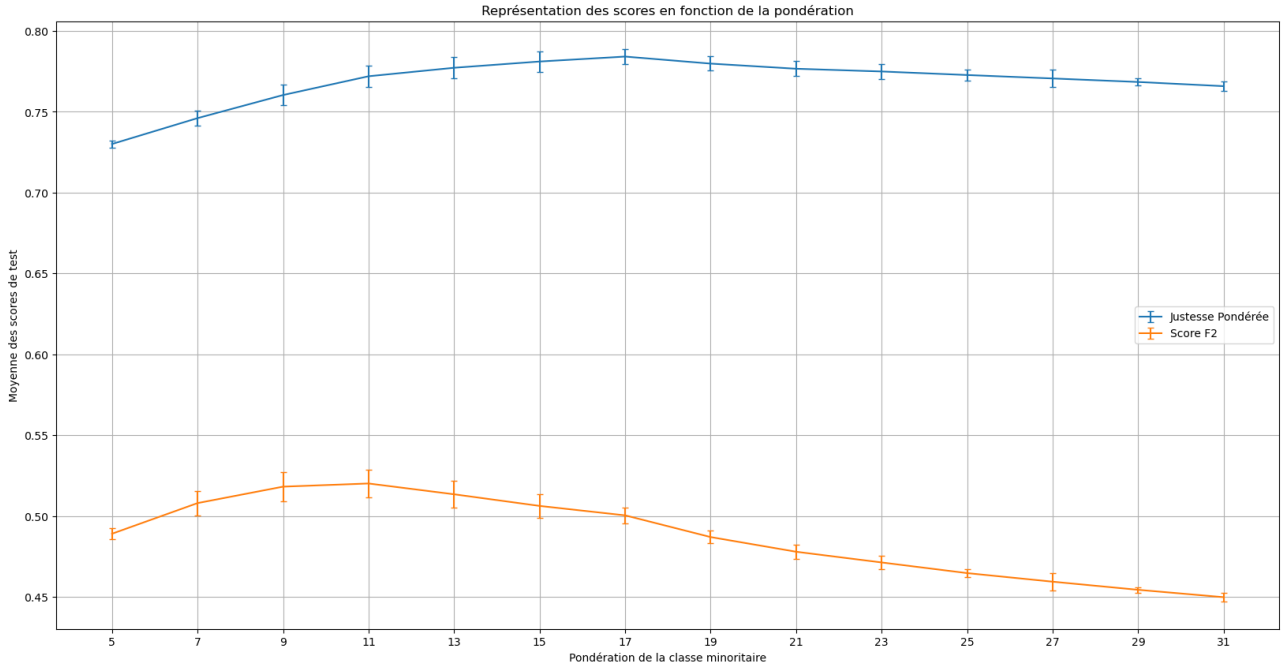


FIGURE C.2 – Graphique des scores F_2 et de justesse pondérée selon la pondération

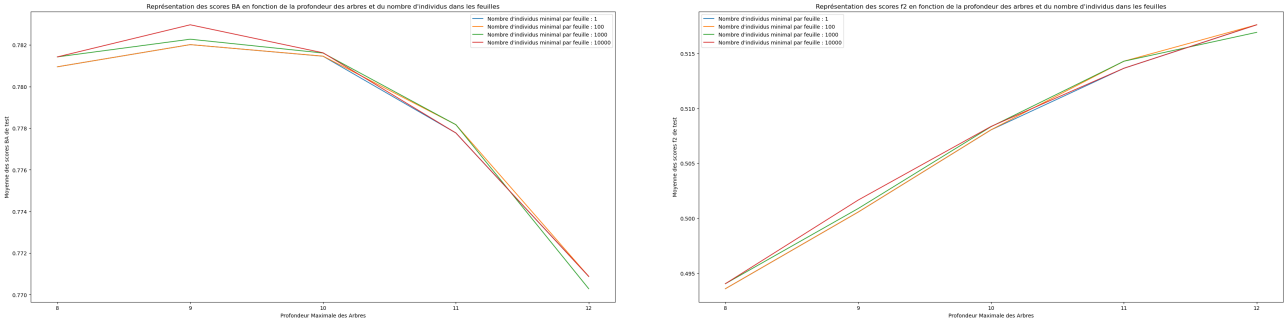


FIGURE C.3 – Graphique des scores F_2 et de justesse pondérée en fonction de la profondeur et du nombre d'individus par feuille

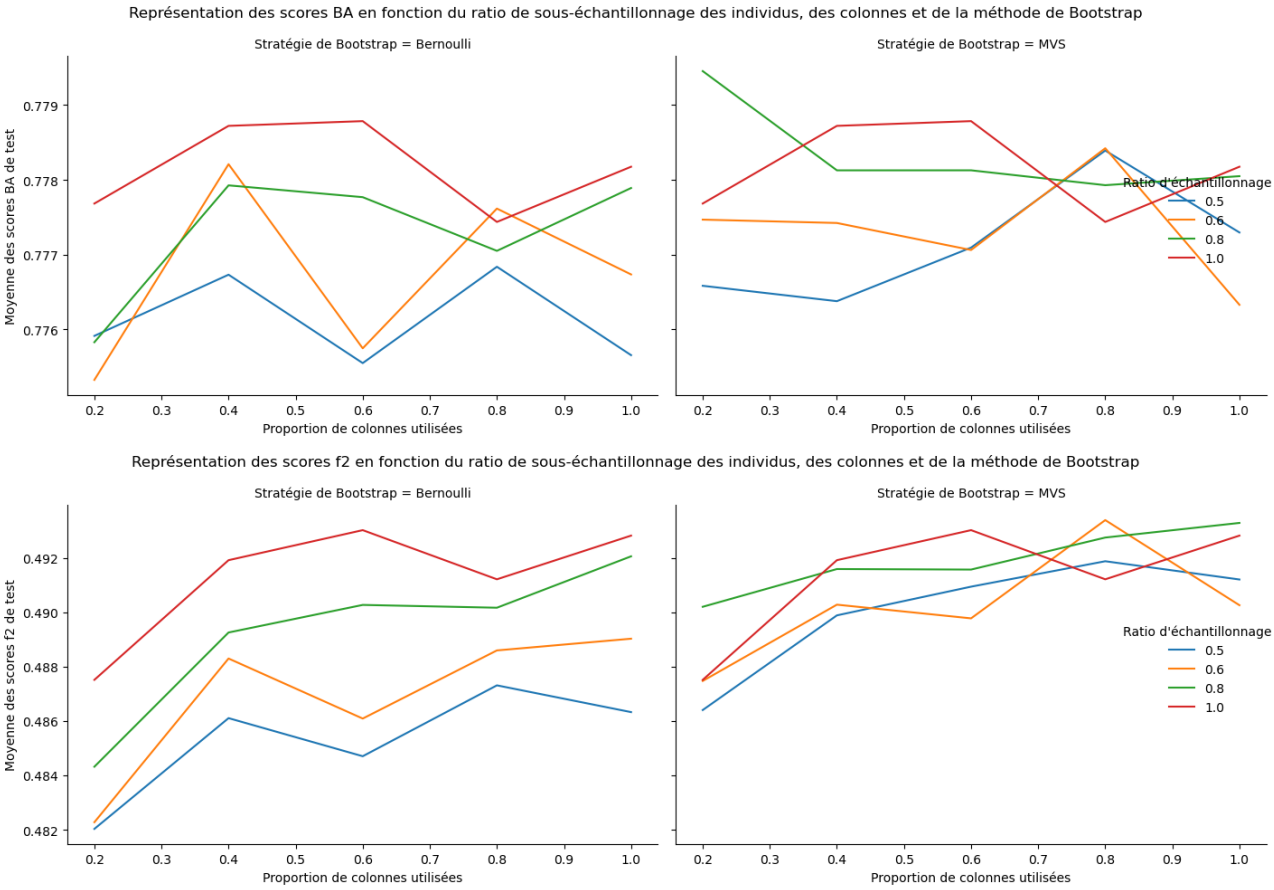


FIGURE C.4 – Résumé des scores de justesse pondérée et F_2 pour le paramétrage de colsample_bylevel et subsample

Liste des tableaux

Table des figures

Liste des Équations

Bibliographie

[1] France Assureur. Données clés 2021. https://www.franceassureurs.fr/wp-content/uploads/2022/09/donnees-cles-2021_juin22_webinteractif.pdf, 2022.

[2] France Assureur. Le marché de l'assurance automobile des particuliers en 2021, 2022.

[3] Przemyslaw Biecek and Tomasz Burzykowski. Explore, explain, and examine predictive models. with examples in r and python. https://ema.drwhy.ai/featureImportance.html, 2020.

[4] Claire Boyer. Machine learning. ISUP, 2021.

[5] Leo Breiman. Bagging predictors. Statistics Department, University of California, Berkeley, 1996. https://link.springer.com/article/10.1023/A:1018054314350.

[6] Leo Breiman. Random forest. Statistics Department, University of California, Berkeley, 2001. https://link.springer.com/content/pdf/10.1023/A:1010933404324.pdf.

[7] Leo Breiman. Manual on setting up, using, and understanding random forests v3. 1. Statistics Department, University of California, Berkeley, 2002. https://www.stat.berkeley.edu/~breiman/Using_random_forests_V3.1.pdf.

[8] Université de Californie. Apache spark. https://spark.apache.org/, 2014. [En ligne ; Lu le 28 août 2022].

[9] Anna Veronika Dorogush et al. Catboost : gradient boosting with categorical features support. Workshop on ML Systems at NIPS 2017, 2017. http://learningsys.org/nips17/assets/papers/paper_11.pdf.

[10] Caruana et al. Intelligible models for healthcare : Predicting pneumonia risk and hospital 30-day readmission. in KDD, 2015.

[11] Liudmila Prokhorenkova et al. Catboost : unbiased boosting with categorical features. Moscow Institute of Physics and Technology, Dolgoprudny, Russia, 2018. https://arxiv.org/pdf/1706.09516.pdf.

[12] W. James Murdoch et al. Interpretable machine learning : definitions, methods, and applications. arXiv :1901.04592, 2019.

[13] Wang et al. Smoking and the occurence of alzheimer's disease : Cross-sectional and longitudinal data in a population-based study. American journal of epidemiology, 1999.

[14] Giovanna Menardi et Nicolas Torelli. Training and assessing classification rules with unbalanced data, 2010. https://www.openstarts.units.it/bitstream/10077/4002/1/Menardi%20Torelli%20DEAMS%20WPS2.pdf.

[15] Yoav Freund et Robert E. Shapire. A decision-theoretic generalization of on-line learning and an application to boosting. Journal of Computer and System Sciences 55, 119-139, 1997. https://www.face-rec.org/algorithms/Boosting-Ensemble/decision-theoretic_generalization.pdf.

[16] A. Fisher. All models are wrong but many are useful : Variable importance for black-box, proprietary, or misspecified prediction models, using model class reliance. arXiv :1801.01489, 2018.

[17] Scott Fortmann-Roe. Understanding the bias-variance tradeoff. https://scott.fortmann-roe.com/docs/BiasVariance.html, 2012.

[18] Legifrance. Loi n° 2005-67 du 28 janvier 2005 tendant à conforter la confiance et la protection du consommateur. https://www.legifrance.gouv.fr/loda/id/JORFTEXT000000606011/, 2005. [En ligne ; Lu le 18 août 2022].

[19] Legifrance. Loi n° 2014-344 du 17 mars 2014 relative à la consommation. https://www.legifrance.gouv.fr/loda/id/JORFTEXT000028738036/, 2018. [En ligne ; Lu le 18 août 2022].

[20] Zachary C. Lipton. The mythos of model interpretability, 2016. .

[21] Christoph Molnar. *Interpretable Machine Learning*. 2 edition, 2022.

[22] Wiki Stat. Arbres binaires de décision. https://www.math.univ-toulouse.fr/~besse/Wikistat/pdf/st-m-app-cart.pdf, 2022.

[23] Miller Tim. Explanation in artificial intelligence : Insights from the social sciences. arXiv Preprint arXiv :1706.07269, 2017.

[24] Štrumbelj Erik and Igor Kononenko. An efficient explanation of individual classifications using game theory. Journal of Machine Learning Research 11 (March), 2010.

[25] Štrumbelj Erik and Igor Kononenko. Explaining prediction models and individual predictions with feature contributions. Knowledge and Information Systems 41, 2014.