



GRANDE ÉCOLE
D'ACTUARIAT
ET DE GESTION
DES RISQUES

INSTITUT DE SCIENCE FINANCIÈRE ET D'ASSURANCES

PROJET DATA SCIENCE

RAPPORT

GROUPE 9

ABDEL WEDOUD ISMAIL
GIMIÉ FLORIAN
TRUONG TINA

I. Introduction	4
1. Contexte	4
2. Présentation du sujet	4
3. Procédé de l'étude	5
II. Analyse univariée	6
1. La variable sexe	6
2. La variable carType	7
3. La variable carCategory	7
4. La variable occupation	8
5. La variable age	9
6. La variable carGroup	9
7. La variable bonus	10
8. La variable carValue	11
9. La variable material	12
10. La variable Region	13
11. La variable subRegion	13
12. La variable cityDensity	14
13. La variable exposure	14
14. La variable claimNumber	16
15. La variable claimValue	16
III. Classification	18
1. Préparation de la classification	18
a. Nettoyage de la base de données	18
b. Feature engineering et traitements supplémentaires	18
c. Plus de données pour un meilleur modèle ?	19
d. Imbalanced data	20
e. Traitement de la variable exposure	22
f. L'intelligence artificielle pour assister au machine learning ?	22
2. Métriques d'évaluation	23
a. Evaluation de la performance du classificateur	23
b. Evaluation des classificateurs qui produisent des prédictions probabilistes	24
3. Modèle du Random Forest	25
a. Commentaires modèle CART	25
b. Résultats	26
4. Modèle XGBoost	28
5. D'autres modèles possibles	29
6. Conclusion et Shapley	30
a. Shapley summary plot	30

b. Shapley force plot	31
c. Conclusion de la partie classification	31
IV. Régression	32
1. Préparation de la Régression	32
a. Première alternative	32
b. Deuxième alternative	32
3. Modèle Random Forest pour la régression	34
a) Features importance	34
b) Paramètres de notre modèle	35
d) Résultat: deuxième alternative	35
e) Bucketing ?	38
i) Idée d'implémentation	38
ii) Une analyse intéressante	39
f) Shapley	40
g) Rendu final	40
V. Simulation du marché pour mesurer l'efficacité des algorithmes retenus face à la concurrence	41
1. Algorithme qui nous donne le résultat comptable	41
a. Situation de concurrence	41
2. Commentaires et conclusion	42
VI. Références	42

I. Introduction

1. Contexte

En tarification, l'actuaire doit considérer les données historiques mises à sa disposition (caractéristiques de l'assuré, primes, sinistralité, ...) et être en mesure, grâce à son expertise, de proposer un prix d'équilibre du contrat. Ce prix, dit prime pure, doit refléter le plus fidèlement possible les coûts engendrés par les sinistres futurs que l'assureur sera amené à régler l'année suivante. Il est ensuite en partie *chargé* par les frais d'acquisition et de gestion pour finalement donner la prime commerciale.

Outre les applications de la théorie de la crédibilité en assurance non vie, les méthodes d'apprentissage statistique apportent de nouvelles compétences à l'actuaire en tarification. Celui-ci peut ainsi sonder les données, essayer de comprendre quels profils de risque impactent le bilan de l'assureur, et discuter des solutions avec les équipes de conception des produits. En effet, cet apprentissage du portefeuille de l'assureur doit pouvoir donner lieu à une concrétisation d'un point de vue commercial, telle qu'un lancement de nouveaux produits d'assurance. Enfin, il est important que l'actuaire puisse expliquer les résultats obtenus. Certains algorithmes d'apprentissage statistique tels que le *boosting* sont connus pour être difficiles à interpréter (d'où leur appellation *black box*). Un travail d'extraction d'un modèle plus simple est parfois nécessaire pour que l'actuaire puisse présenter ses résultats à ses supérieurs.

2. Présentation du sujet

Nous avons à disposition une base de données d'entraînement de 60 000 assurés en automobile qui ont été observés l'année dernière pendant au moins 3 mois. A l'aide de l'apprentissage statistique, une prime devra être proposée à chacun des 20 000 assurés d'une base test.

Afin de mesurer la performance de nos algorithmes, un algorithme choisira 6 groupes aléatoirement et chaque assuré sera affecté au groupe qui leur propose le prix le plus bas. A l'issue d'une année, on regardera ensuite le résultat comptable de chaque groupe. Ce processus est itéré plusieurs fois.

Il faut alors prédire le meilleur prix pour l'assuré sans sous-tarifier (le risque est que l'on fasse des pertes) ni sur-tarifier par rapport aux autres (le risque est que les assurés choisissent d'autres contrats). En effet, il s'agit de rendre l'activité la plus pérenne possible en évitant ces risques de sous et sur tarification.

3. Procédé de l'étude

Pour sa richesse en *packages* et sa documentation, nous utilisons le langage **R** pour la partie descriptive de l'analyse. Pour chaque covariable, nous avons confronté les deux bases *train* et *test* afin d'identifier et comprendre les deux populations étudiées, repérer les incohérences, et apporter les modifications nécessaires. Pour ce faire, nous avons réalisé une étude univariée préliminaire sur chaque covariable.

L'apprentissage statistique a été implémenté à l'aide du langage de programmation **Python**.

Dans cette étude, nous avons pour chaque individu les montants cumulés de sinistres :

$$Y = \sum_{k=1}^N Y_k$$

où N est une variable aléatoire désignant le nombre de sinistres, et Y_k le montant du k -ième sinistre.

Nous ne pouvons donc pas faire l'hypothèse habituelle sur la prime pure :

$$\pi = E(N)E(Y_k)$$

Une alternative consiste à calculer la probabilité qu'un individu X donné ait un sinistre (classification) et estimer le coût moyen espéré pour les individus ayant eu au moins un sinistre (régression) :

$$\pi_x = P(Y>0 | X=x)E(Y | Y>0, X=x)$$

Après traitement de la base de données *train*, nous avons choisi de mettre en œuvre divers algorithmes, à savoir les modèles *Random Forest* et *XGBoost*. En effet, tout en conservant un schéma qui reste suffisamment interprétable, le *Random Forest* est souvent la contrepartie du modèle *CART* auquel l'on reproche parfois une forte variance du résultat. Quant à l'*XGBoost*, nous perdons en interprétabilité.

Les métriques employées pour la validation des modèles sont définies dans les parties respectives.

II. Analyse univariée

Comme évoqué précédemment, la base de données *train* est composée de 60 000 individus, décrits selon 14 variables explicatives. La réponse est le montant agrégé des sinistres par individu pendant la durée d'exposition *claimValue*.

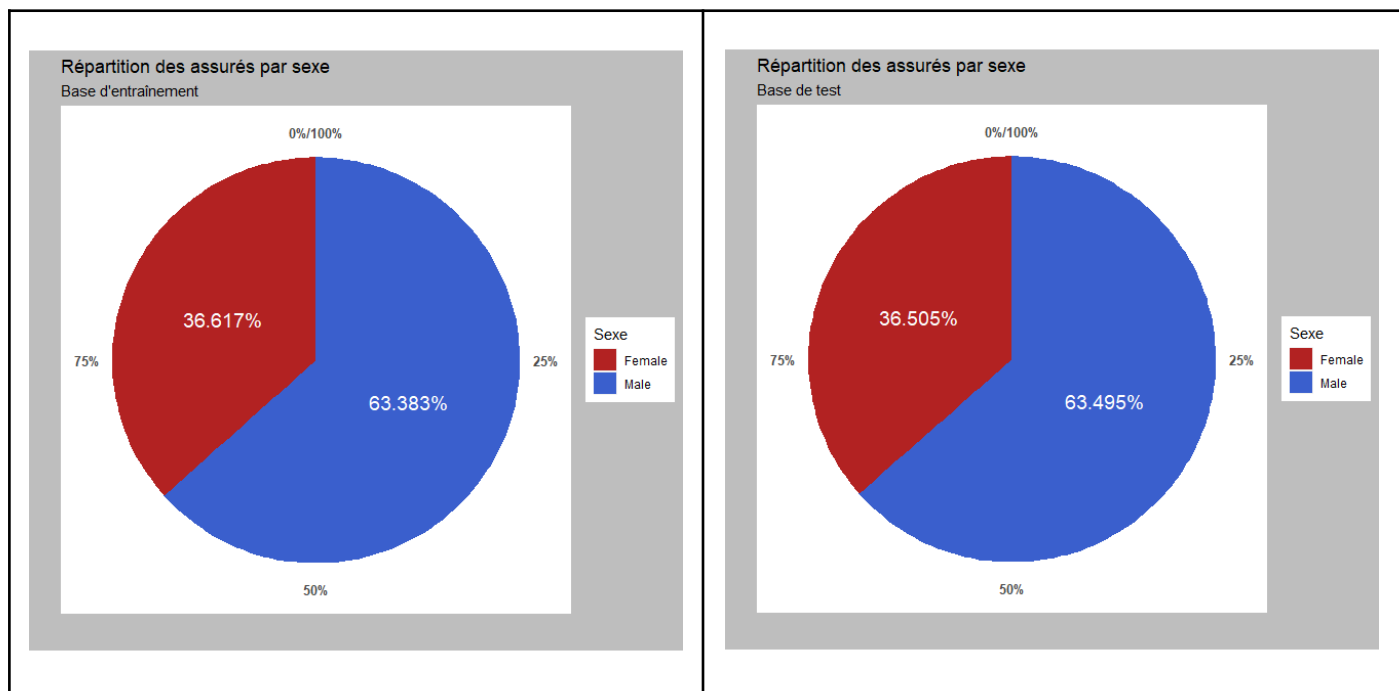
Les individus sont numérotés grâce à la variable *id*. En effet, il est important, en pratique, que les assurés se voient attribuer un identifiant unique qui permet à l'assureur, et plus précisément aux directions de la gestion et de l'inventaire, de retrouver rapidement le dossier du sinistré.

Dans le cadre de notre étude, cette variable n'intervient pas dans la tarification, mais on peut quand même s'assurer qu'il n'y a pas de doublons, ce qui est le cas ici.

Enfin, après vérification, la base de données ne contient pas de valeurs manquantes.

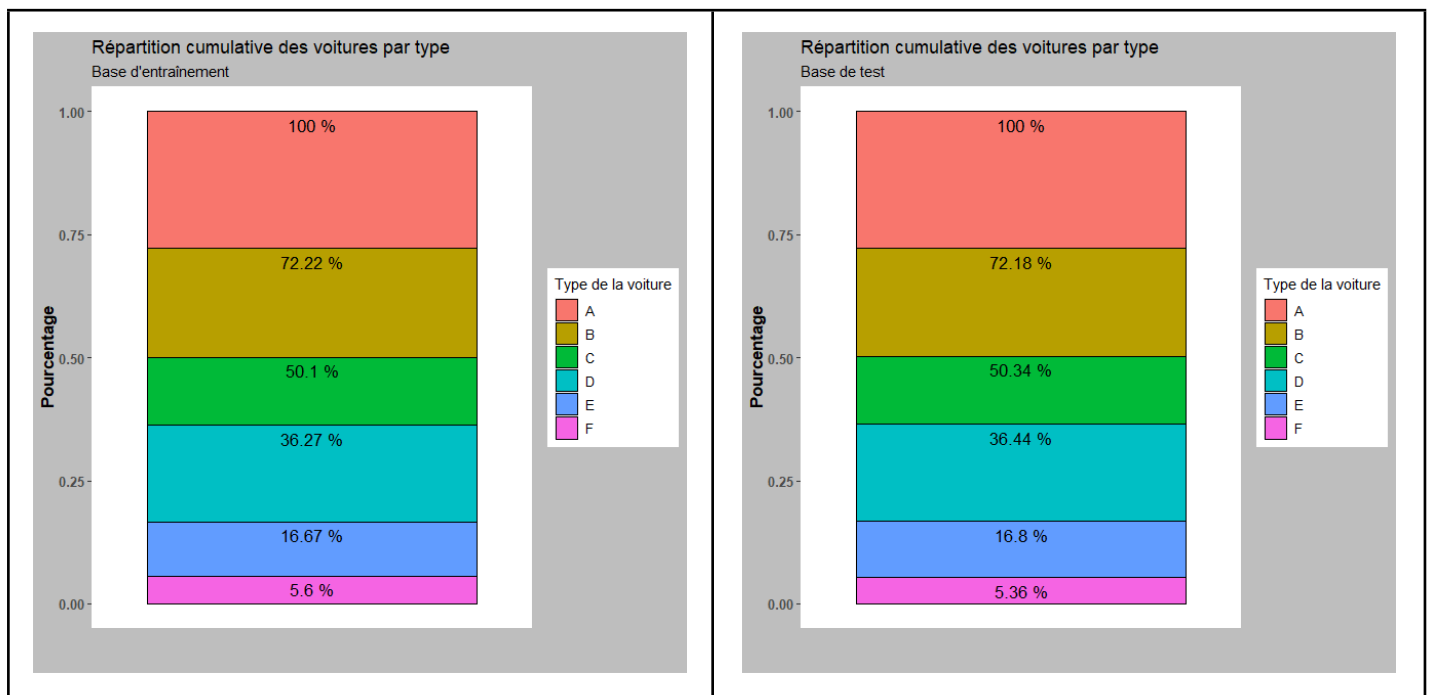
1. La variable sexe

La variable *gender* est une variable qualitative binaire. Les bases *train* et *test* ont une répartition homme/femme similaire. La proportion d'hommes est de 63%, celle des femmes de 37%.



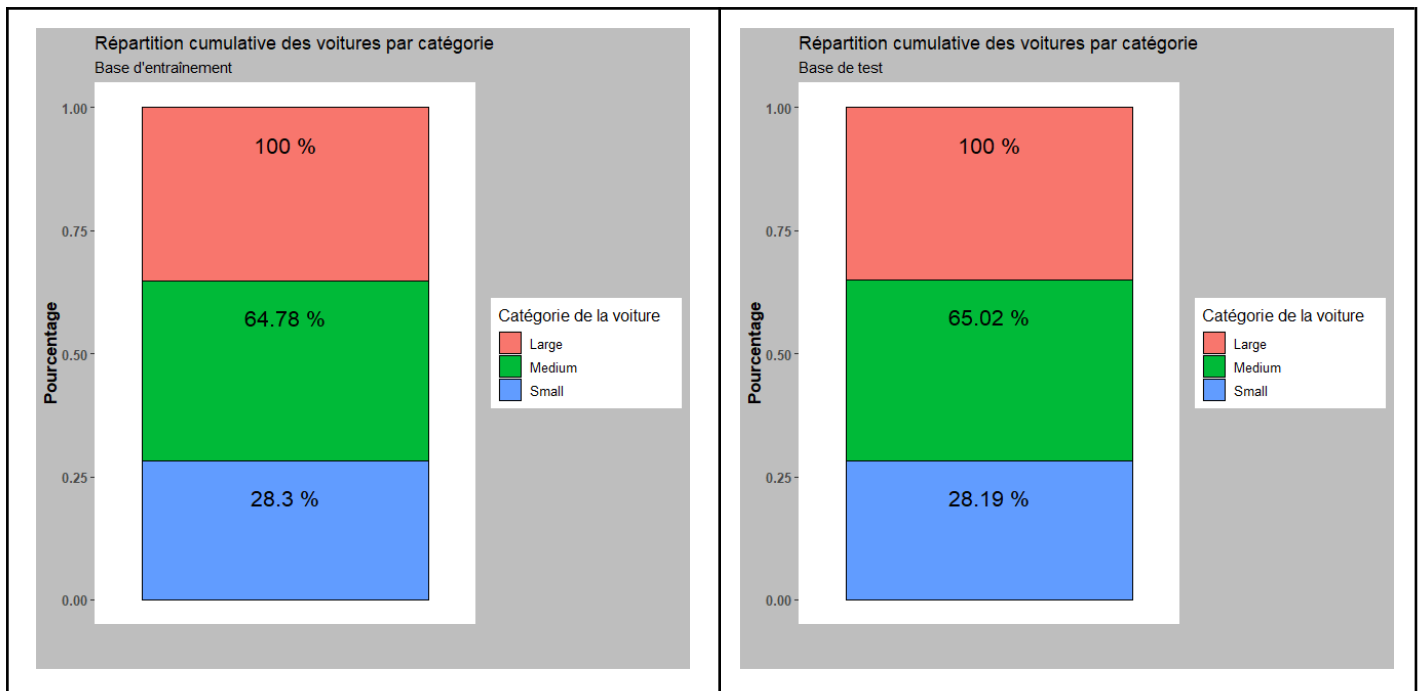
2. La variable carType

La variable *carType* est une variable catégorielle. Elle représente les segments dans lesquels sont classées les voitures. Par exemple, le segment A contient les mini-citadines et le segment F les voitures de luxe. Outre une répartition similaire entre la base *train* et *test*, nous pouvons constater que les voitures citadines et familiales (A, B et D) totalisent près de trois quarts des voitures du portefeuille, ce qui est assez représentatif de ce que l'on pourrait observer dans la réalité.



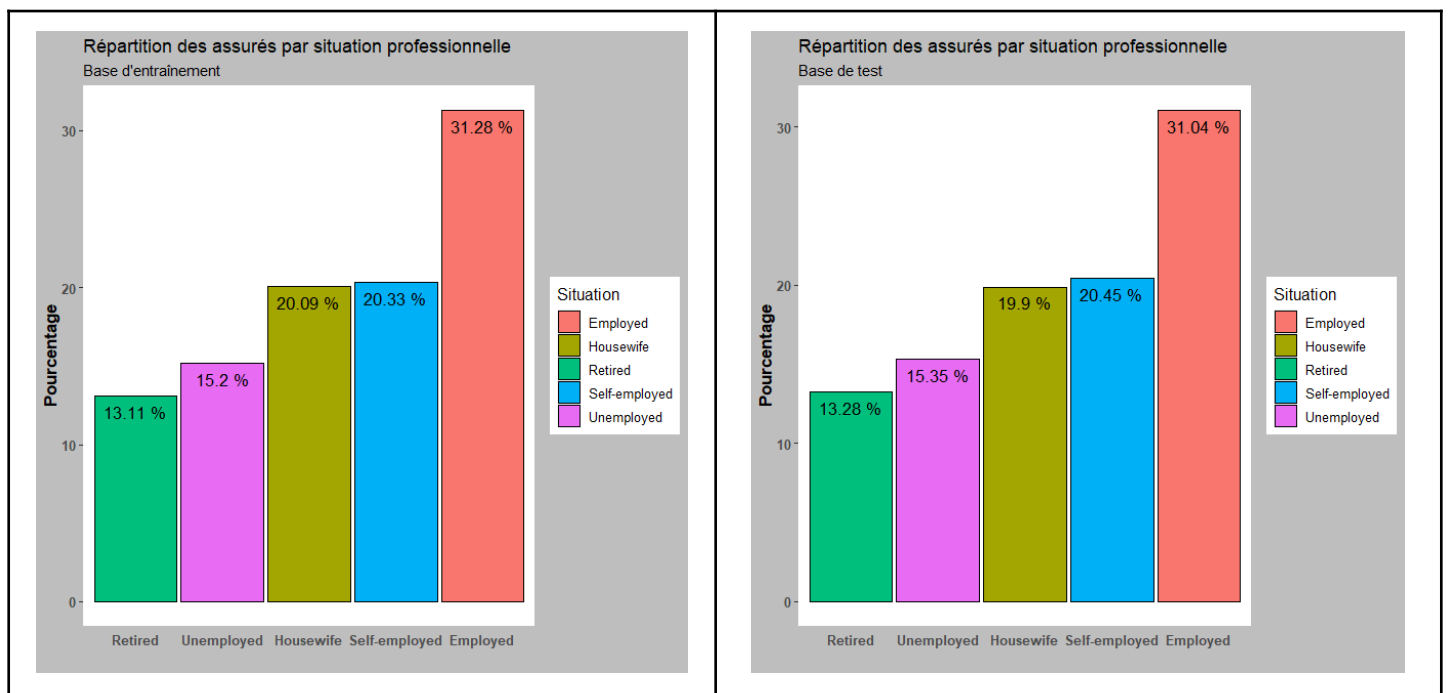
3. La variable carCategory

La variable *carCategory* est une variable qualitative ternaire. Ses modalités regroupent des voitures dont le volume est compris dans un intervalle défini par la classification. A titre d'exemple, dans la classification américaine, les voitures de catégorie "Large" sont celles dont le volume est supérieur à 120 pieds cubes.



4. La variable occupation

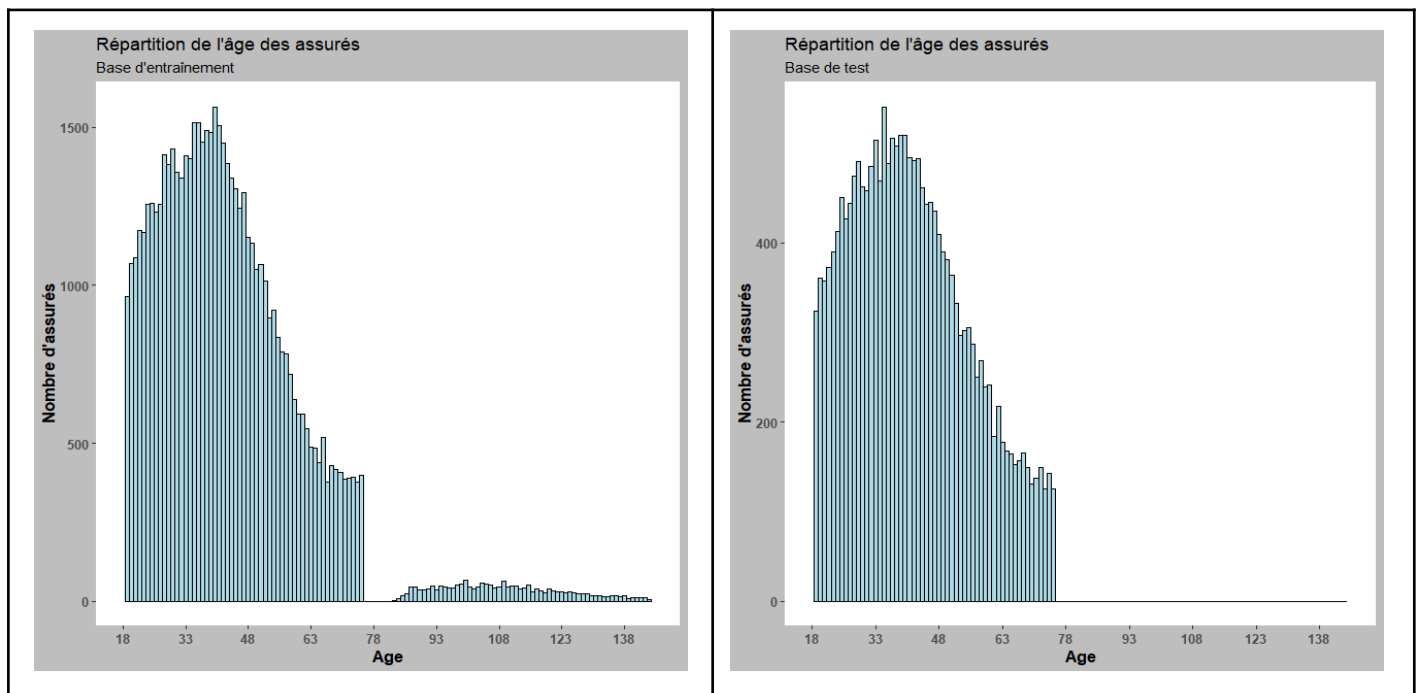
La variable *occupation* donne la situation professionnelle des individus. Ainsi, près de la moitié des assurés travaillent. Nous pouvons nous attendre à ce que ces assurés prennent plus souvent la voiture et soient donc plus exposés au risque.



5. La variable age

La variable *age* est une variable quantitative indiquant l'âge des assurés. Nous pouvons observer une concentration des âges entre 18 et 50 ans avec un pic à 40 ans. Une décroissance constante et rapide se fait dès lors jusqu'à l'atteinte d'un plateau des âges "avancés" entre 65 et 75 ans.

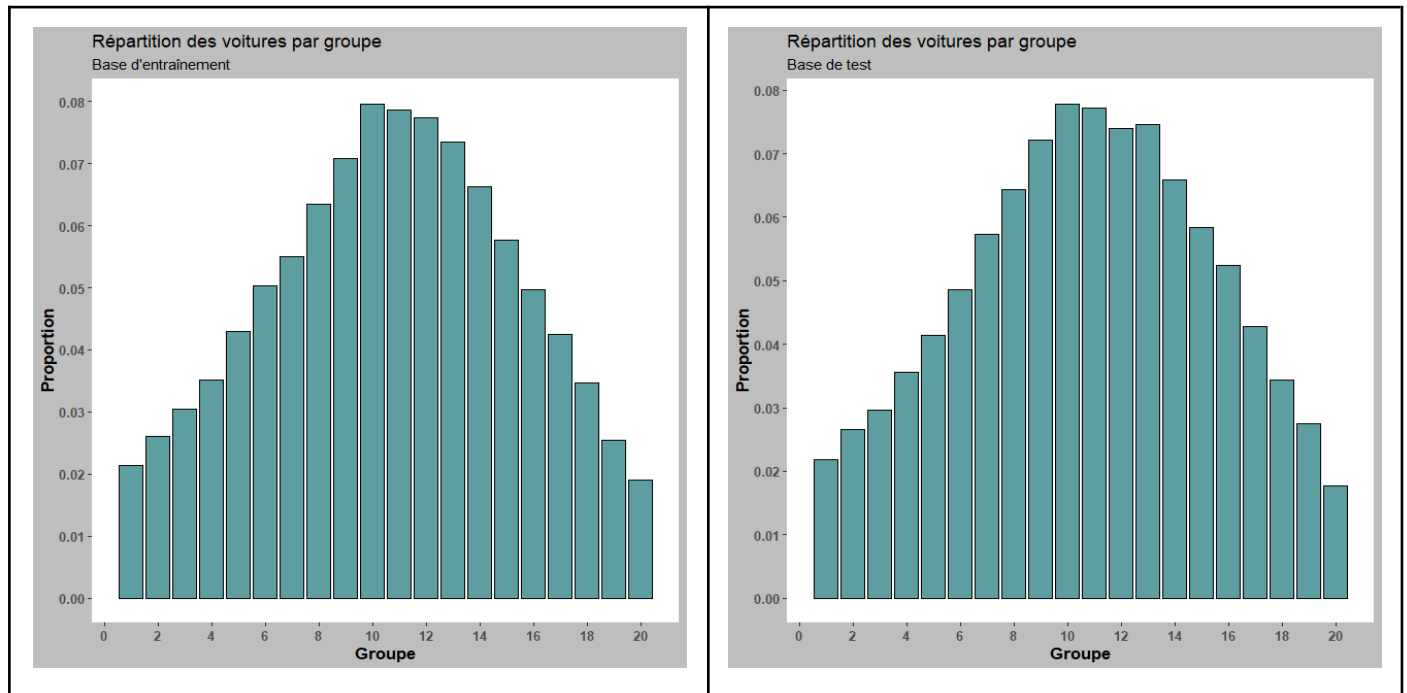
Par ailleurs, des assurés inscrits comme étant âgés de plus de 83 ans (soit 2000 assurés) sont présents dans la base d'entraînement. Il est important de s'interroger sur la validité de ces âges avant de mettre en place l'apprentissage statistique. En effet, il serait peu judicieux d'apprendre d'un assuré âgé de 132 ans. Hormis quelques cas particuliers, l'assureur a probablement oublié de retirer ces assurés dont l'âge est incrémenté automatiquement chaque année, ou s'est trompé lors de la saisie. Dans tous les cas, un traitement doit être fait.



6. La variable carGroup

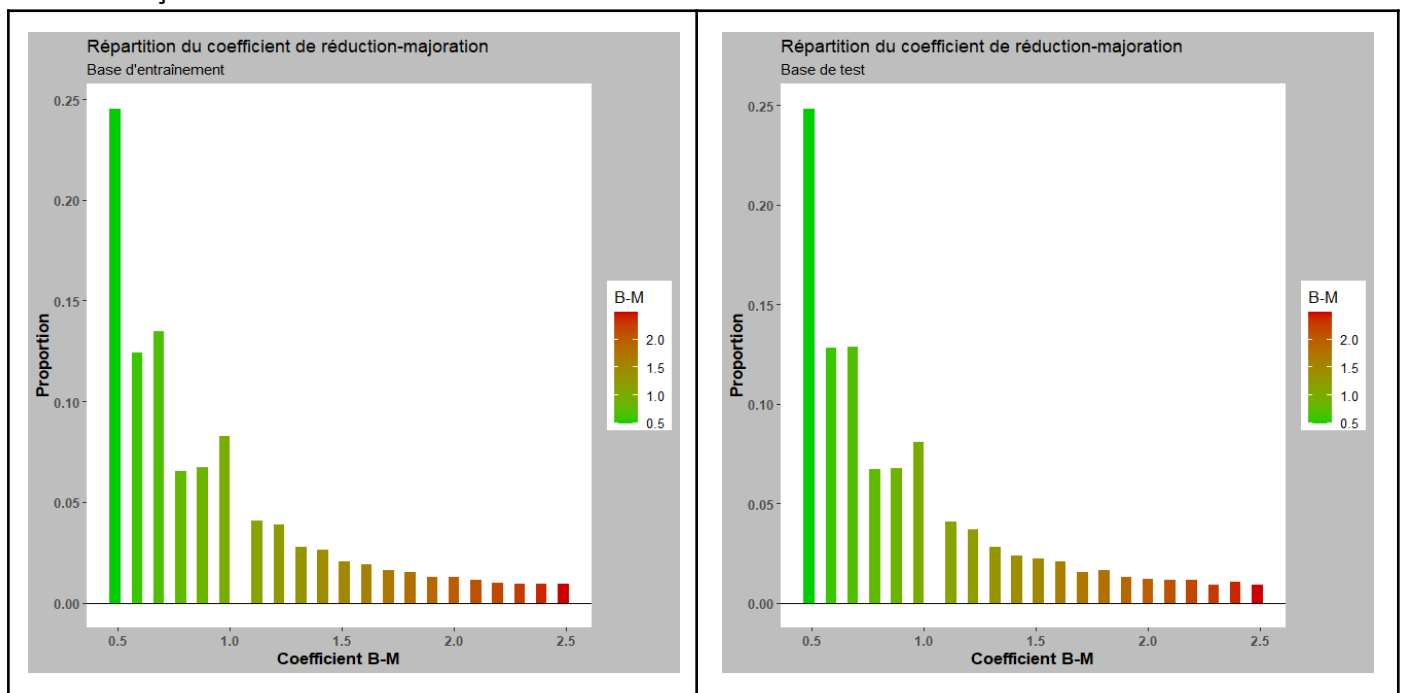
Toujours dans l'objectif de classer les véhicules, la variable *carGroup* est constituée de 20 catégories allant de 1 à 20. En réalité, nous pouvons retrouver 50 groupes dans lesquels des voitures aux caractéristiques similaires (valeur de la voiture, performance, sécurité, ...) sont intégrées. Le groupe 1 correspond aux voitures bon marché, tandis que le groupe 50 recense les voitures de luxe.

Nous pouvons observer une répartition “en cloche” pour les bases d’entraînement et de test avec un cumul pour les groupes moyens qui représentent certainement des voitures “classiques”.



7. La variable bonus

La variable *bonus* est le coefficient de réduction-majoration du système bonus-malus français.

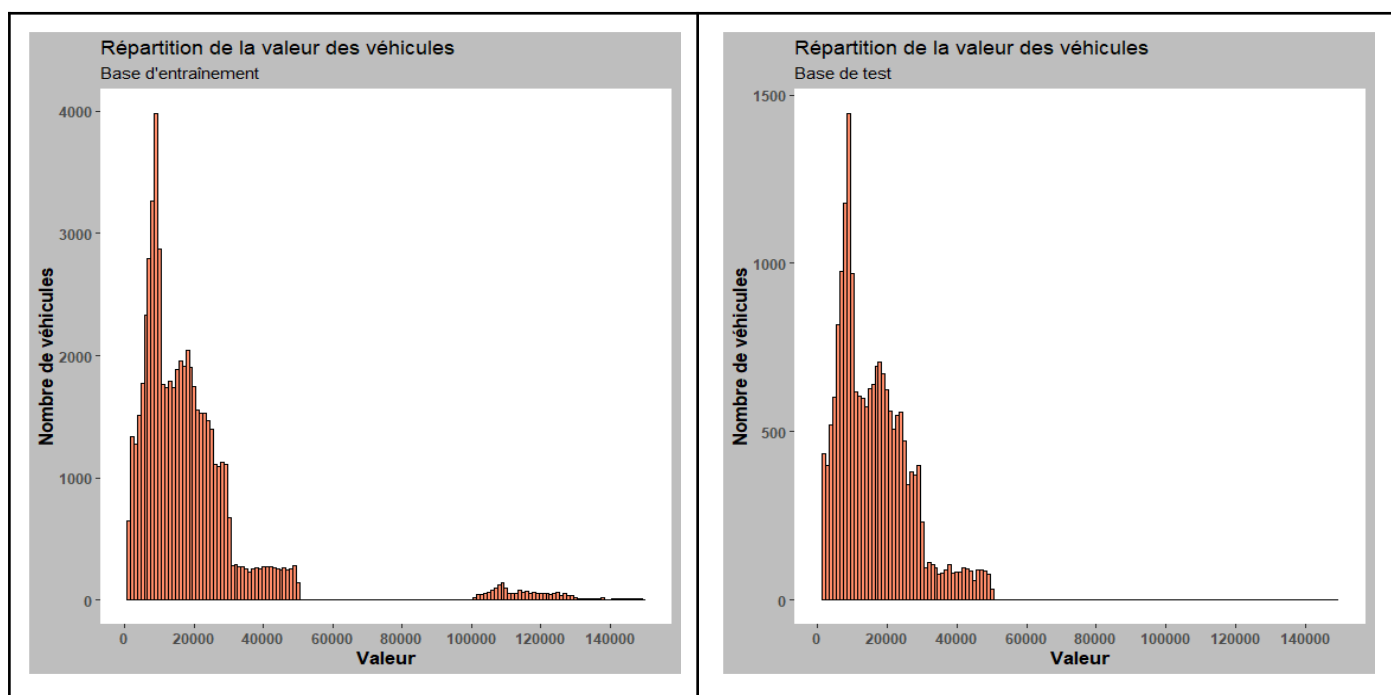


Nous pouvons voir ici qu'il est compris entre 0,5 et 2,5. Selon la réglementation, le malus est plafonné à 3,5 fois la prime, et le bonus ne peut excéder la moitié de la prime. Un sinistre responsable majore la prime pour l'année suivante de 1,25, tandis qu'une année sans sinistre responsable se traduit par une minoration de celle-ci de 0,95. Un nouvel assuré démarre avec un coefficient égal à 1.

Historiquement, en moyenne, les conducteurs ont un coefficient inférieur à 1. Cette moyenne peut varier d'une région à l'autre. Par exemple, le département des Bouches-du-Rhône est un département avec un CRM moyen relativement élevé.

Notre base de données semble refléter cette distribution avec une proportion d'assurés ayant un coefficient inférieur à 1 bien plus importante que celle avec un coefficient malussé.

8. La variable carValue



La valeur d'un véhicule est un élément qui entre en jeu dans la tarification en automobile et dans la classification des véhicules comme vu précédemment. La répartition est assez concentrée entre 10 000€ et 30 000€ avec une moyenne de 20 000€ pour la base d'entraînement. Compte tenu de la faible valeur de certains véhicules, certains ont certainement un kilométrage important.

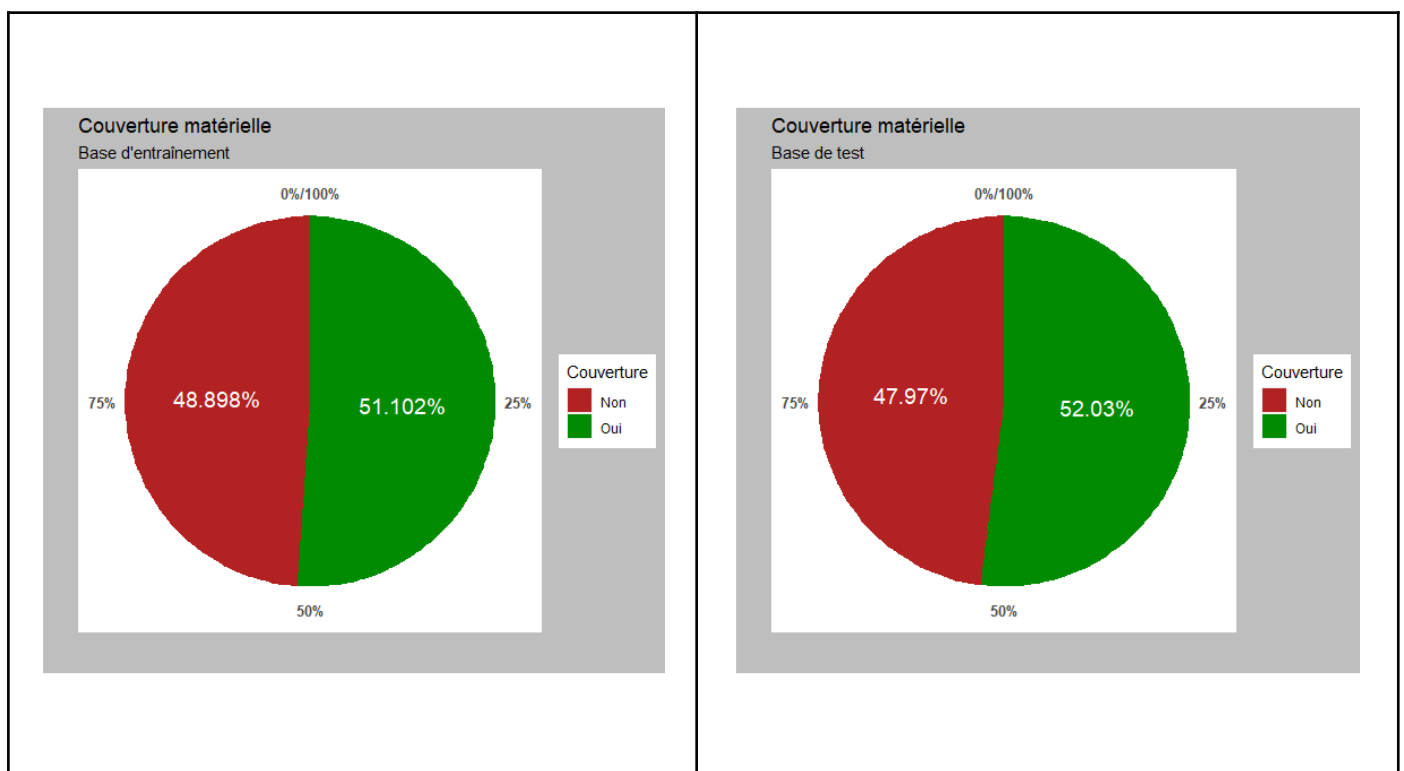
Après une absence notable de voitures dont la valeur est comprise entre 50 000€ et 100 000€, nous retrouvons pour la base d'entraînement précisément 2 000 véhicules dont la

valeur excède 100 000€. Afin d'éviter un apprentissage sur ces 2000 individus, nous ferons l'hypothèse que ces valeurs sont erronées et seront retraitées par la suite. En effet, il est difficilement concevable que, parmi 60 000 assurés, l'assureur définisse une politique interdisant de couvrir les véhicules valant entre 50 000€ et 100 000€ et acceptant ceux de valeur supérieure.

9. La variable material

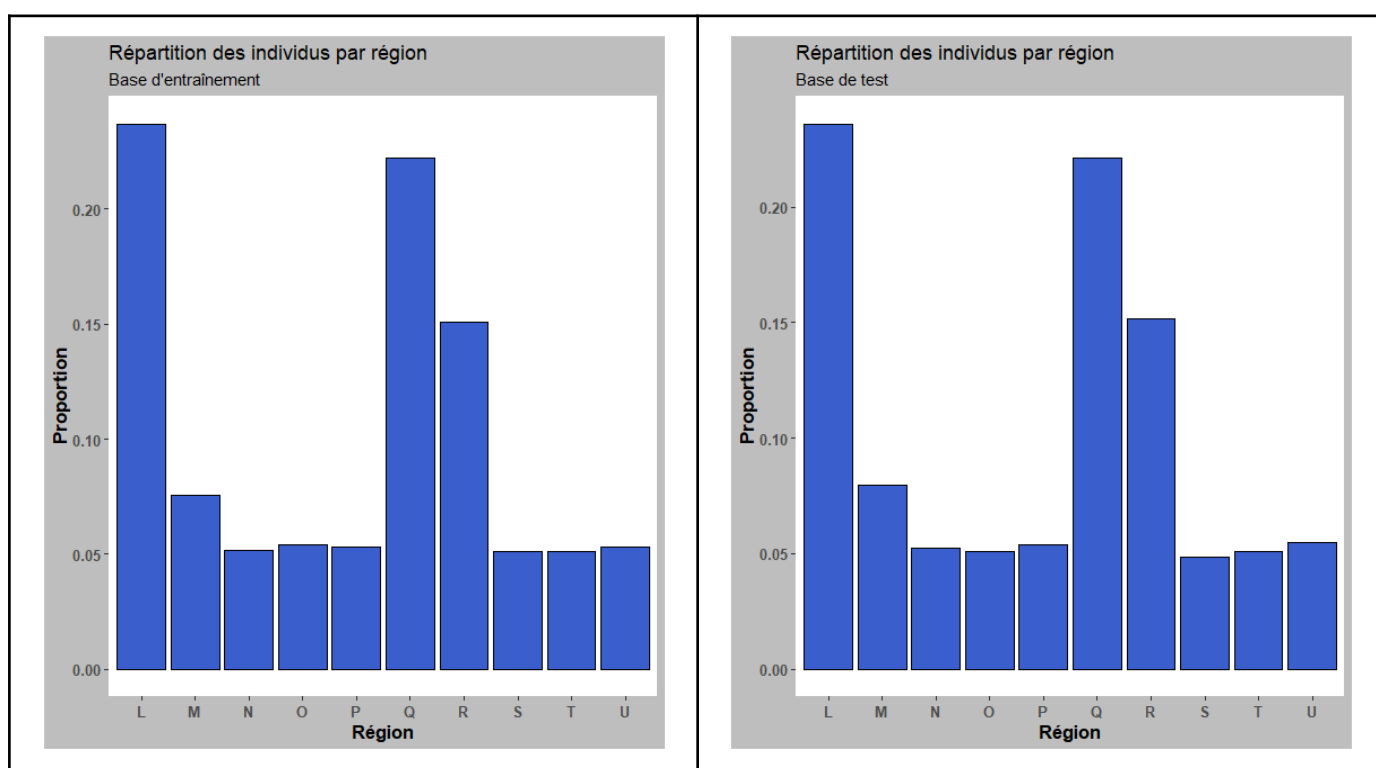
La variable *material* est une variable binaire valant 1 si l'individu possède une couverture matérielle pour son véhicule, et 0 sinon. En effet, depuis 1958, tout conducteur d'un véhicule automoteur doit posséder une couverture RC en cas d'accident. Néanmoins, la couverture matérielle reste facultative et dépend de l'appétence au risque, voire des moyens financiers de l'individu.

Nous pouvons voir une répartition presque équivalente entre les assurés qui n'ont pas de couverture matérielle et ceux qui en possèdent une.



10. La variable Region

La variable *region* est une variable catégorielle à 10 modalités, exprimées sous forme de lettres de L à U, représentant les régions dans lesquelles résident les assurés. Il aurait été intéressant de connaître explicitement le nom des régions pour mieux comprendre les résultats de l'apprentissage statistique qui suivra. Toutefois, nous pouvons d'ores et déjà remarquer 61% des assurés résident dans les régions L, Q et R. La répartition est relativement équilibrée aux alentours de 5% des assurés par région pour les autres régions. Nous pouvons nous attendre à ce que cette variable ait un impact non négligeable lors de la phase d'apprentissage statistique.



11. La variable subRegion

La variable *subRegion* recense 471 modalités qui définissent des sous-régions dans lesquelles résident les individus. Une modalité se décompose suivant la lettre associée à la variable *region* et un nombre qui doit certainement caractériser les subdivisions propres à une région.

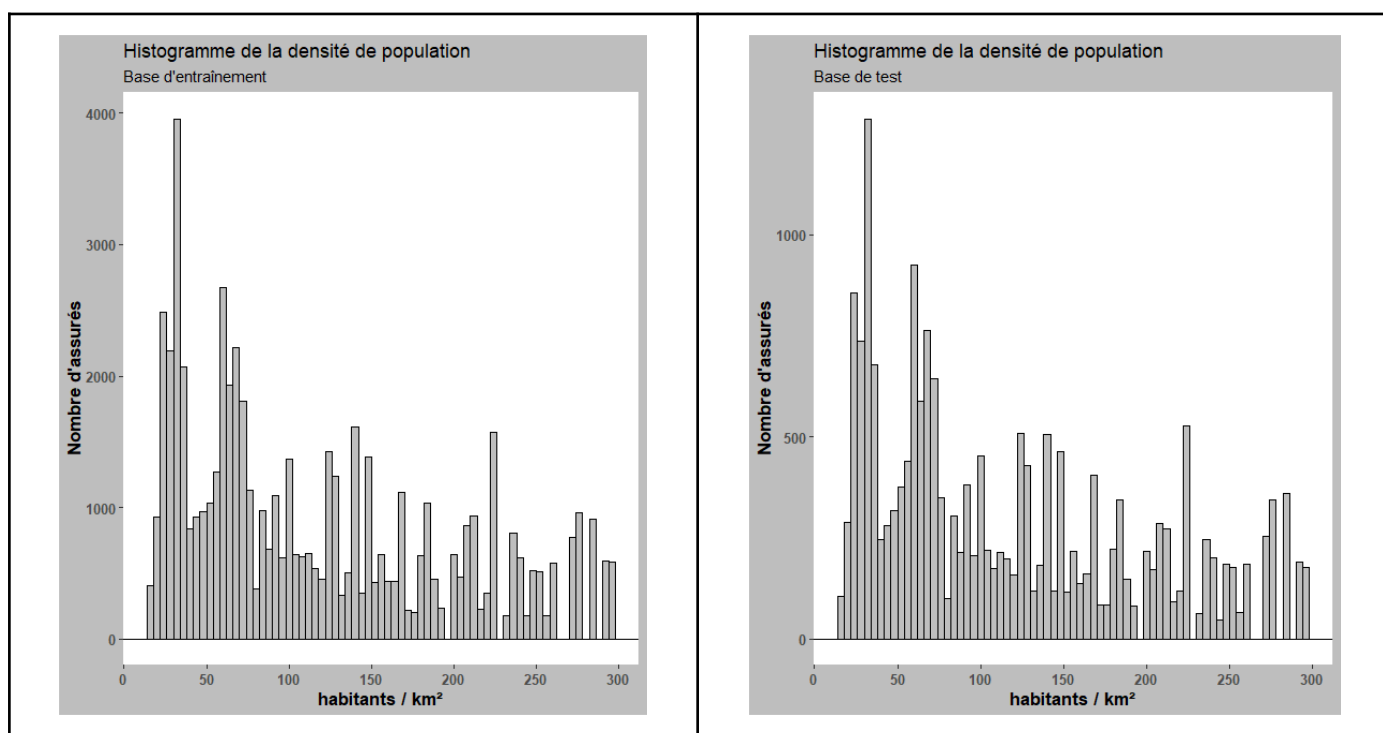
En outre, il serait peu judicieux de donner une représentation graphique compte tenu du nombre considérable de modalités. A ce sujet, nous sommes conscients de la nécessité de

retravailler cette donnée pour la suite, car nous pouvons nous attendre à ce que les modèles d'apprentissage sur-apprennent de cette variable, notamment pour le Random Forest.

Après étude de la variable *cityDensity* (qui suit dans l'analyse), nous avons constaté qu'à une sous-région était associée une ville de résidence. La variable *subRegion* constitue donc une sorte de "jointure" entre les variables *region* et *cityDensity* qu'il est donc nécessaire de conserver.

12. La variable *cityDensity*

Comme son nom l'indique, cette variable continue donne la densité de population des villes de résidence des assurés. Les graphiques ci-dessous permettent d'apprécier la répartition des assurés selon la densité de population des villes dans lesquelles ils résident. Ainsi, nous observons deux principaux extrema locaux pour des villes de densité de population autour de 25 habitants/km² et 60 habitants/km², ce qui s'apparente plutôt à des petites communes.



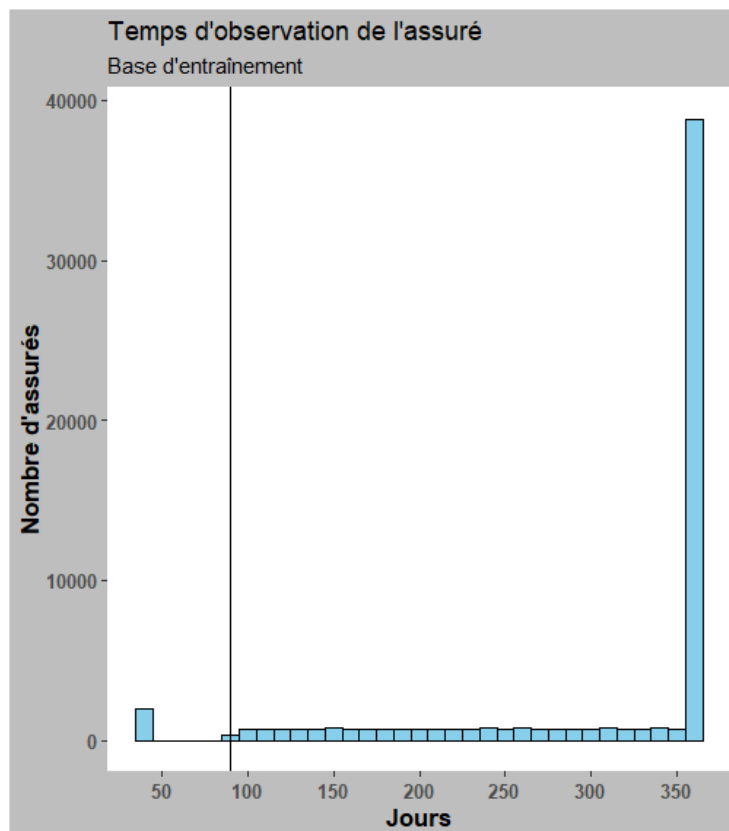
13. La variable *exposure*

La variable *exposure* représente la durée d'observation d'un assuré donné. D'après le contexte de l'étude :

“Vous avez à votre disposition un historique (base dataTrain) pour 60 000 polices observées pendant au moins 3 mois lors de l’année précédente”.

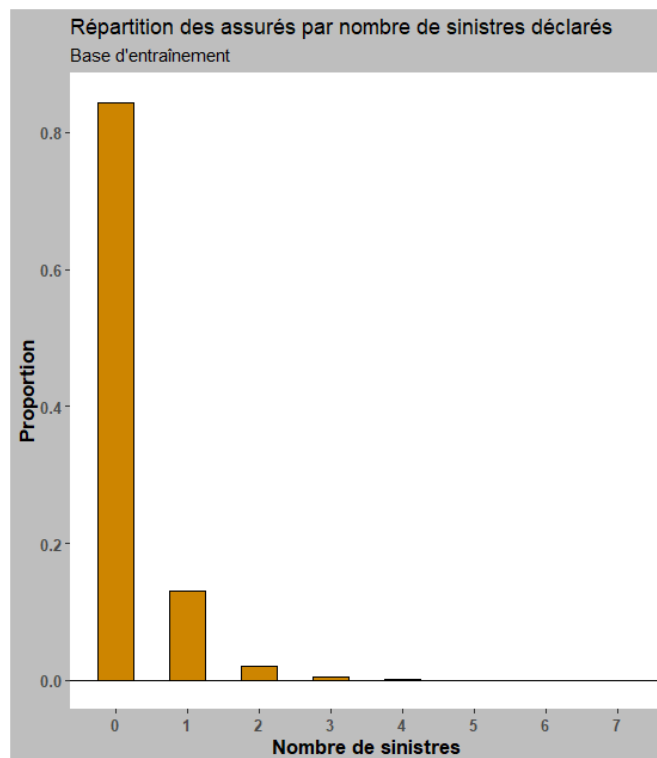
Or, nous remarquons que 2000 assurés ont un temps d’observation égal à 37 jours. Nous en déduisons que ces assurés ne peuvent être exploités, car ils ont une période d’observation inférieure à 3 mois.

Une grande partie des assurés (environ 38 000) ont été observés pendant une année complète. Puisqu’ils ont été observés plus longtemps que les autres, nous leur accordons un plus grand poids dans l’étude. Cette variable servira à l’introduction de poids selon que l’individu a été observé plus ou moins longtemps.



14. La variable *claimNumber*

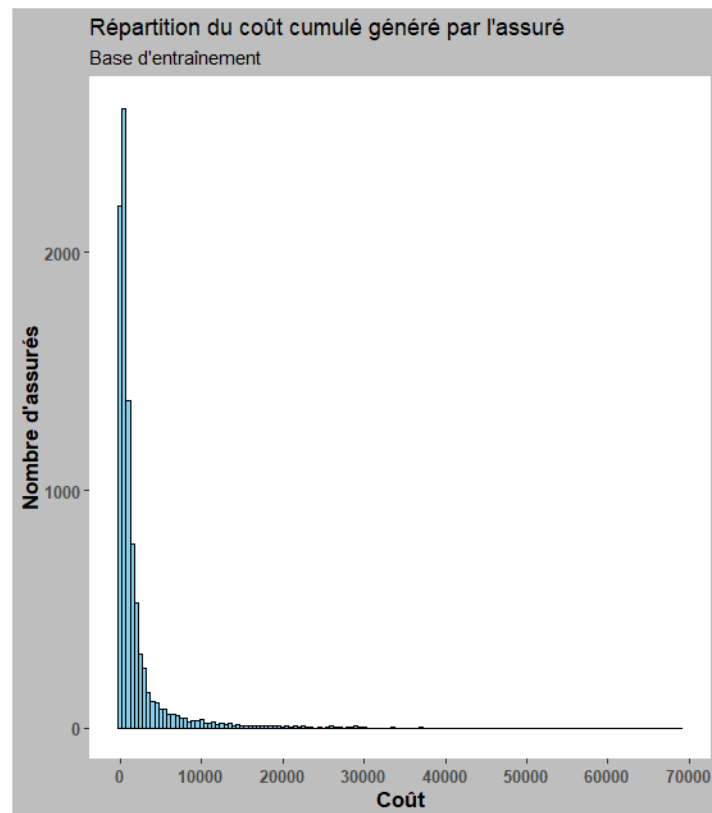
La variable *claimNumber* est une variable à valeurs entières qui donne le nombre de sinistres déclarés par l'assuré durant la période d'observation. Nous pouvons constater que, et heureusement pour l'assureur, plus de 84% des assurés n'ont pas déclaré de sinistres. Cette variable est notre réponse pour l'étape de la classification. Quant à l'étape de la régression nous ne retiendrons que les assurés ayant eu un ou plusieurs sinistres afin d'évaluer le coût moyen espéré d'un sinistre pour un individu donné.



15. La variable *claimValue*

La variable *claimValue* est notre variable à expliquer pour la régression. Elle correspond au montant agrégé des sinistres pour un assuré donné. Il est néanmoins intéressant de voir quelle forme prend la distribution. Selon le rapport France Assureurs de 2021, un sinistre RC matériel coûte en moyenne 1645€ en automobile. Sans considérer les non sinistrés, nous avons un coût agrégé moyen de 2042€, ce qui montre bien que l'automobile est une branche de l'assurance non vie dont la fréquence des sinistres est élevée mais dont l'intensité reste relativement faible. Certains sinistres à plusieurs dizaines de milliers d'euros impliquent probablement un accident corporel. En effet, la RC pour un

accident corporel est souvent bien plus coûteuse pour l'assureur qu'un sinistre matériel seulement.



III. Classification

Dans cette partie, nous allons nous occuper de la classification, c'est-à-dire classer les individus par 1 ou 0, 1 correspondant au fait que l'on prédit qu'un individu ait un sinistre. On cherche donc dans cette partie:

$$P[Y > 0]$$

1. Préparation de la classification

a. Nettoyage de la base de données

À partir de notre analyse univariée, nous décidons de faire les traitements suivants pour nettoyer notre base de données:

- **élimination des individus de plus de 75 ans** (comme certains sont sûrement décédés mais que leur bonus malus sont mis à jour, cela peut biaiser les résultats)
- **élimination des individus dont les voitures ont une valeur de plus de 50 000**
- **élimination des individus avec une exposition inférieure strictement à 90 jours** (ce sont des valeurs aberrantes d'après l'énoncé)
- **one-hot-encoding pour les variables catégorielles** (en particulier region et subRegion)
- enlever la colonne 'id'
- enlever la colonne 'Female'

À l'issue de cela, nous nous retrouvons avec une base de données de 56 000 lignes contre 60 000 lignes au départ, soit 93% de la base de données initiale.

b. Feature engineering et traitements supplémentaires

Pour le calcul de la prime, notre approche est de d'abord trouver un algorithme pour prédire la probabilité que l'individu ait un sinistre dans l'année qui va venir (à partir de la variable *claimNumber* qui vaut 1 s'il a eu un sinistre et 0 sinon), puis de prédire, dans le cas où il a eu un sinistre, le montant de ce sinistre (on constatera par la suite qu'on peut prédire le montant de sinistres à partir du résultat de cette partie et améliorer notre modèle). On doit donc réaliser tout d'abord un **modèle de classification** et ensuite un **modèle de régression**.

Pour ce faire, il faut alors réaliser:

- **une modification de la variable *claimNumber*, qui vaut 1 si *claimNumber* > 0 et 0 sinon**

Cependant, si on fait cela, on perdra l'information sur le nombre de sinistres qu'a eu cet individu pendant la période observée. Il y a plusieurs possibilités pour pallier ce problème: on peut par exemple utiliser la colonne initiale *claimNumber* comme poids (le modèle serait dans ce cas moins "flexible") ou bien multiplier les observations lorsqu'un individu a eu plusieurs sinistres pendant la période (le modèle serait alors plus "flexible").

Cela permet aussi au modèle de se focaliser davantage sur des individus avec 3 accidents que ceux avec 1 accident sur la même période observée.

Comme on a remarqué que la base de données est très déséquilibrée et que le poids que l'on donne à chaque observation change beaucoup sur la qualité de la prédiction (en particulier la qualité de la prédiction sur la classe minoritaire, c'est-à-dire les individus qui ont enregistré des accidents), on a décidé de choisir la deuxième option:

- **multiplier de manière artificielle les individus selon le nombre d'accidents qu'il a eu pendant la période avant de recoder la variable *claimNumber* en 0 et 1.**

Voici un schéma de l'opération:

id	...	claimNumber
1	...	1
2	...	3

⇒

id	...	claimNumber
1	...	1
2	...	1
2	...	1
2	...	1

c. Plus de données pour un meilleur modèle ?

Pour améliorer la robustesse et la fiabilité de notre modèle, nous avons proposé à une autre équipe d'échanger nos base de données.

Notons qu'il est possible d'altérer nos données avant de les envoyer à notre concurrence. Pour que cela se passe de manière inaperçue, nous pouvons, par exemple, effectuer des permutations des colonnes *claimValue* et *claimNumber* entre les individus qui ont au moins un sinistre, ou faire quelques permutations sur une colonne avec une grande significativité. On peut également augmenter ou diminuer le montant et le nombre de sinistres pour perturber leur modèle, selon que l'on veut qu'ils récupèrent les assurés et fassent un mauvais résultat comptable, ou qu'ils ne trouvent pas de clients. Comme nous voulons aussi faire un bon résultat, il semblerait plus judicieux de les éliminer du marché en faisant en sorte qu'ils proposent un prix plus élevé.

Bien sûr, l'équipe adverse peut très bien faire de même et nous faire parvenir une base de données faussée. Il faut donc utiliser leur base de données avec précaution, à moins que l'on soit certain que la base est fiable. En nettoyant la base de données de l'équipe adverse, nous gardons 93% de la base initiale. Par la suite, nous allons utiliser la base de données de l'équipe adverse (on leur fait confiance...).

Par bonne conscience et comme il y a plus d'une vingtaine de groupes qui participent, nous avons décidé de leur faire parvenir notre base de données telle que l'on l'a reçue.

Notons que cette stratégie a aussi ses limites puisqu'il est possible qu'il y ait des individus qui soient dupliqués sur nos deux bases de données. Nous allons supposer que cela n'est pas le cas.

d. Imbalanced data



Schéma explicatif de l'undersampling et de l'oversampling

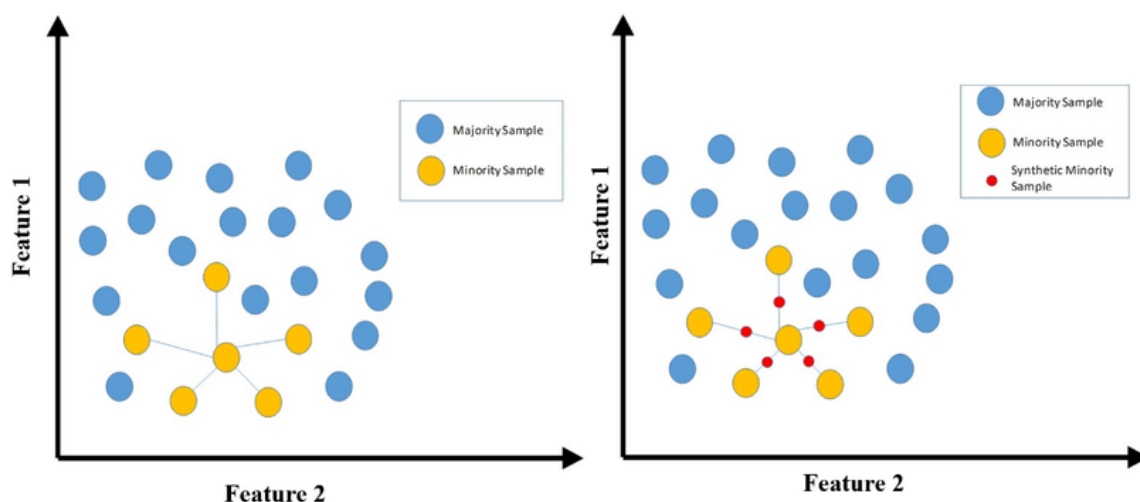


Schéma explicatif du SMOTE

Encore une fois, le problème que nous rencontrons avec la classification est le **déséquilibre des classes** que l'on essaie de prédire: près de 85% des individus n'ont pas eu de sinistres sur la période observée et seulement 2,6% ont eu plus d'un sinistre.

Il y a plusieurs possibilités pour pallier ce problème:

- undersampling
- oversampling
- augmentation du poids de la classe minoritaire par rapport à la classe majoritaire
- SMOTE (Technique de Suréchantillonnage des Minorités Synthétiques) : une méthode de sur-échantillonnage de la classe minoritaire dans un jeu de données afin de rééquilibrer la répartition des classes. L'idée derrière SMOTE est de générer des échantillons synthétiques de la classe minoritaire plutôt que de simplement suréchantillonner en répliquant les échantillons existants. Il le fait en trouvant les k plus proches voisins pour chaque échantillon de la classe minoritaire et en générant de nouveaux échantillons le long de la ligne reliant l'échantillon et ses voisins. Cela peut aider à réduire le surapprentissage, car les nouveaux échantillons ne sont pas des copies exactes des échantillons existants, mais plutôt des versions légèrement perturbées de ceux-ci (définition de chatGPT).

En testant, nous remarquons que l'undersampling présente de meilleurs résultats que l'oversampling. Cela est normal car l'oversampling augmente les chances de faire de l'overfitting, qui est exacerbé par le fait que la classe minoritaire ne représente que 15% de notre base de données. S'il faut choisir, il faut donc privilégier l'undersampling à l'oversampling.

Il est également possible, **pour aider le modèle à généraliser**, d'utiliser des algorithmes de SMOTE sur la base *train*. Faute de temps et dû au fait que l'on a déjà la base de données d'une autre équipe, nous n'avons pas testé cette technique mais pensons qu'avec les bons réglages, le SMOTE peut améliorer la robustesse du modèle (mais l'amélioration serait peu considérable car on se doute que le fait que l'individu ait un sinistre ou pas est aussi en partie imprévisible).

Nous avons décidé d'ajuster les poids différents pour chaque classe pour prendre en compte le déséquilibre des classes. Le poids idéal pour un random forest est de 7 pour la classe 1 contre 1 pour la classe 0.

e. Traitement de la variable exposure

Seuls les individus qui ont été observés au moins 3 mois ont été conservés dans la base d'entraînement. Nous souhaitons donner plus de poids à des individus qui ont été observés sur une période plus grande (365 jours par exemple) que ceux qui ont rejoint le portefeuille de l'assureur plus tard (91 jours). Pour ce faire, la variable *exposure* a été considérée comme une variable de poids. Par exemple, pour un individu i ayant été observé durant une période $t_i \in [91, 365]$ aura un poids au prorata égal à : $\frac{\max(t_i - 90, 0)}{275}$.

On a pris en compte la variable exposure pendant le fitting des modèles, en mettant "exposure" comme poids.

f. L'intelligence artificielle pour assister au machine learning ?

Selon ChatGPT:

" ChatGPT est un modèle de langage de grande envergure développé par OpenAI. Il a été entraîné sur un grand ensemble de données de conversation en utilisant la technique de l'apprentissage par transfert de connaissances.

En plus de ses applications dans les chatbots et les assistants virtuels, ChatGPT peut également être utilisé pour aider à la rédaction de code de l'apprentissage automatique. Par exemple, il peut être utilisé pour générer du code à partir de spécifications ou de descriptions de tâches, ce qui peut être utile pour les développeurs.

En utilisant ChatGPT, les développeurs peuvent également bénéficier de sa capacité à comprendre le langage naturel et à générer du code de manière convaincante et naturelle. Cela peut leur permettre de gagner du temps et de se concentrer sur d'autres tâches

importantes, tout en sachant que le code généré par ChatGPT sera de haute qualité et facile à comprendre.

En résumé, ChatGPT peut être un outil précieux pour les développeurs d'apprentissage automatique qui cherchent à écrire du code en français de manière efficace et de haute qualité. "

Lorsque nous avons eu des doutes, nous avons utilisé ChatGPT pour avoir des conseils sur notre projet. **Les explications étaient claires et le chatbot nous donne également des bouts de code comme exemple pour nous guider.**

ChatGPT a aussi ses limites. Par exemple, en machine learning, il a insisté sur le fait de ne pas utiliser la variable sexe alors que celle-ci est souvent indispensable. Voici un exemple de réponse que ChatGPT apporte :

" It is generally not a good idea to include gender as a predictor in a machine learning model, as it can introduce bias and may not be relevant to the task at hand. Instead, it is important to focus on features that are relevant to the task and that are not discriminatory.

If you do decide to include gender as a predictor in your model, you should be aware of the potential for bias and take steps to mitigate it. This could include ensuring that your training data is representative of the population you are trying to model, and carefully evaluating the performance of your model to ensure that it is not biased against any particular group.

In general, it is important to be mindful of the potential for bias in machine learning and to take steps to minimize it. This can involve careful selection of features, use of techniques such as fair representation learning, and ongoing evaluation of the performance of your model. "

2. Métriques d'évaluation

a. Evaluation de la performance du classificateur

Certaines métriques sont plus sensibles aux bases de données avec des classes déséquilibrées que d'autres. Comme on cherche surtout à prédire les accidents, il faut une métrique adaptée.

Voici quelques métriques d'évaluation qui peuvent être utilisées pour évaluer la performance d'un modèle d'apprentissage sur un jeu de données à déséquilibre de classes :

- **Précision** (Precision) : cette métrique mesure la proportion de prédictions positives vraies faites par le modèle, et peut être utile pour évaluer la performance du modèle sur la classe minoritaire.
- **Rappel** (Recall) : cette métrique mesure la proportion de cas positifs réels qui ont été correctement prédits par le modèle, et peut être utile pour évaluer la performance du modèle sur la classe minoritaire.
- **Score F1** (F1-score) : cette métrique est la moyenne harmonique de la précision et du rappel, et peut être utile pour évaluer la performance globale du modèle sur les deux classes minoritaires et majoritaires.
- **AUC-ROC** : cette métrique mesure la capacité du modèle à distinguer entre les classes positives et négatives, et est moins sensible au déséquilibre de classe que la précision.

En effet, il est important de choisir une métrique d'évaluation appropriée qui tienne compte du déséquilibre de classe dans le jeu de données. Dans certains cas, il peut être nécessaire d'utiliser plusieurs métriques pour obtenir une vision complète de la performance du modèle.

Nous constatons que les modèles ont des facilités à détecter la classe 0, mais plus de difficulté à détecter la classe 1. Or, l'objectif est aussi de détecter la classe 1.

Pour notre modèle, nous avons essayé d'optimiser le score F1, le rappel et la précision de la classe minoritaire tout en faisant attention que les mêmes scores pour la classe majoritaire soient acceptables.

b. Evaluation des classificateurs qui produisent des prédictions probabilistes

On peut utiliser deux autres métriques:

- la **perte logarithmique (logloss)**: c'est la log-vraisemblance négative de l'étiquette vraie donnée la probabilité prédite. La perte logarithmique est définie comme suit:

$$L = -(1/n) * \sum [y * \log(p) + (1 - y) * \log(1 - p)]$$

où L est la perte logarithmique, n est le nombre d'échantillons, y est l'étiquette vraie (0 ou 1) et p est la probabilité prédite de la classe positive.

La perte logarithmique varie de 0 (prédictions parfaites) à l'infini (pire performance). Une perte logarithmique inférieure indique une meilleure performance.

- le **score de Brier (Brier score)** : c'est une métrique qui mesure l'erreur quadratique moyenne entre les probabilités prédites et les étiquettes vraies. Il est défini comme suit:

$$B = (1/n) * \sum [(y - p)^2]$$

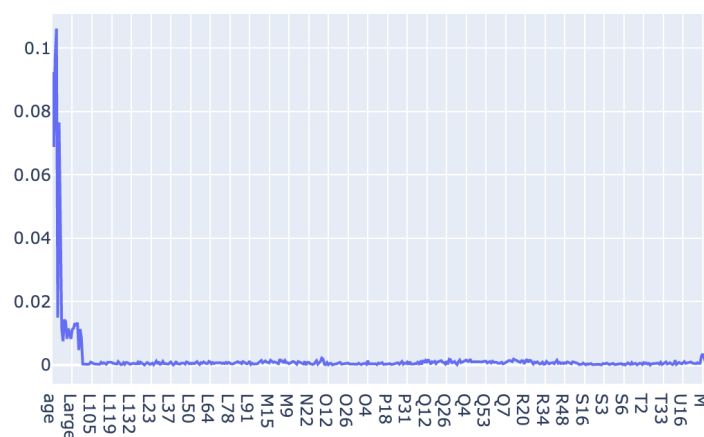
où B est le score de Brier, n est le nombre d'échantillons, y est l'étiquette vraie (0 ou 1) et p est la probabilité prédite de la classe positive.

Le score de Brier varie de 0 (prédictions parfaites) à 1 (pire performance). Un score de Brier inférieur indique une meilleure performance.

Les deux pertes logarithmiques et scores de Brier sont sensibles aux probabilités prédites et peuvent être utilisées pour comparer les performances de différents modèles de classification. Cependant, la perte logarithmique est plus sensible aux probabilités extrêmes (proches de 0 ou 1) et pénalise davantage ces dernières, tandis que le score de Brier est plus sensible à l'exactitude des probabilités prédites et pénalise les déviations par rapport aux étiquettes vraies de manière plus équitable.

3. Modèle du Random Forest

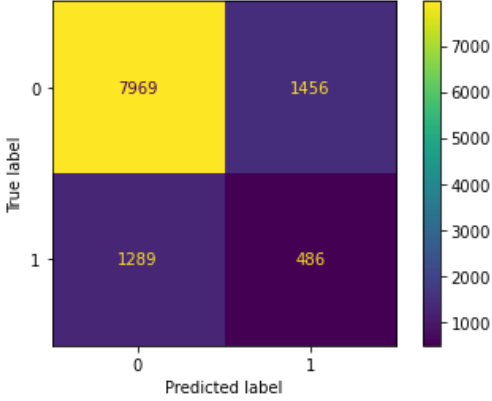
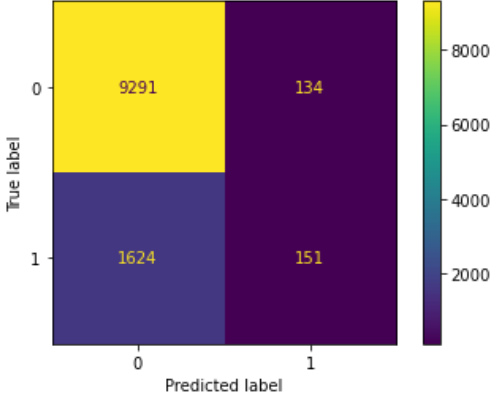
a. Commentaires modèle CART



Graphique donnant les features importance pour la classification (arbre CART)

Ce graphique montre que, pour le problème de classification, les régions, en particulier, les sous-régions, ne contribuent pas beaucoup au modèle. De plus, une petite partie des variables est déjà pertinente. Il est possible d'enlever les variables régions et sous-régions pour notre modèle, mais cela serait trop laborieux et n'aurait pas d'intérêt. On

peut garder toutes les variables, même lorsqu'elles sont moins pertinentes et jouer sur d'autres paramètres du modèle comme la profondeur des arbres pour forcer le modèle à ne pas utiliser ces variables.

	
<p>Pour l'arbre maximal, on voit que l'algorithme prédit avec justesse un 0 dans 85% des cas, mais ne prédit avec justesse 1 que dans 25% des cas. Ceci est dû au fait que notre base de données est fortement déséquilibrée: seulement 15% de la base de données est sinistrée.</p>	<p>Après pruning, on a 98% de prédictions correctes dans le cas où il n'y a pas de sinistres et seulement 9% de prédictions correctes dans le cas où il y a un sinistre.</p> <p>Comme la base de données est déjà fortement déséquilibrée, le pruning n'améliore pas la prédiction sur les 1 mais que sur les 0.</p>

Il faut impérativement corriger le déséquilibre dans la base de données de départ.

On peut envisager du oversampling, undersampling, jouer sur les poids, etc.

b. Résultats

Nous avons décidé de commencer par le modèle de Random Forest, modèle constitué de plusieurs arbres de décision CART. De ce fait, le Random Forest est plus robuste qu'un CART.

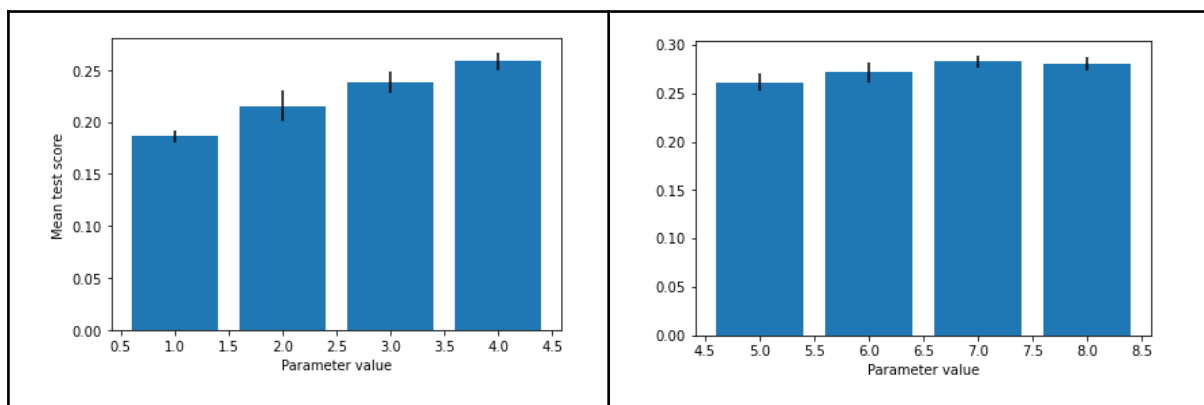
On va regarder le modèle Random Forest est suivant:

```
<bound method BaseEstimator.get_params of
RandomForestClassifier(class_weight='balanced', max_depth=15, max_features=0.6,
                        min_samples_leaf=7)>
```

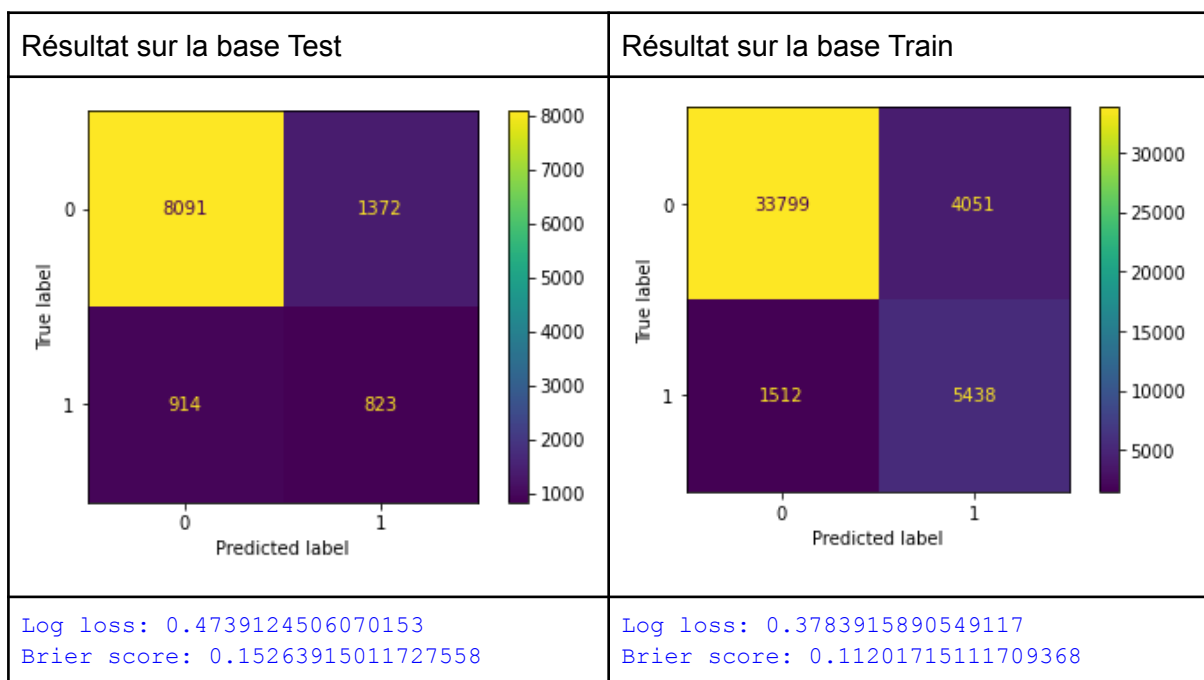
En effet, comme il y a un déséquilibre des classes assez marqué, trouver le bon poids des classes améliore considérablement la performance de notre modèle. De plus, pour chaque arbre, comme il y a beaucoup de features, on peut s'en limiter à 60% à chaque fois. De plus, changer la profondeur des arbres améliore aussi le modèle.

A noter que `class_weight = balanced` signifie qu'on va pondérer avec selon la proportion relative des classes 0 et 1. On peut aussi jouer avec des poids différents.

Voici un tableau qui résume nos différents tests pour des poids allant de 1 à 8 et le score F1 obtenu.



Le Random Forest obtenu en choisissant judicieusement les poids, le nombre maximal de features et la profondeur est clairement meilleur que le modèle CART précédent :



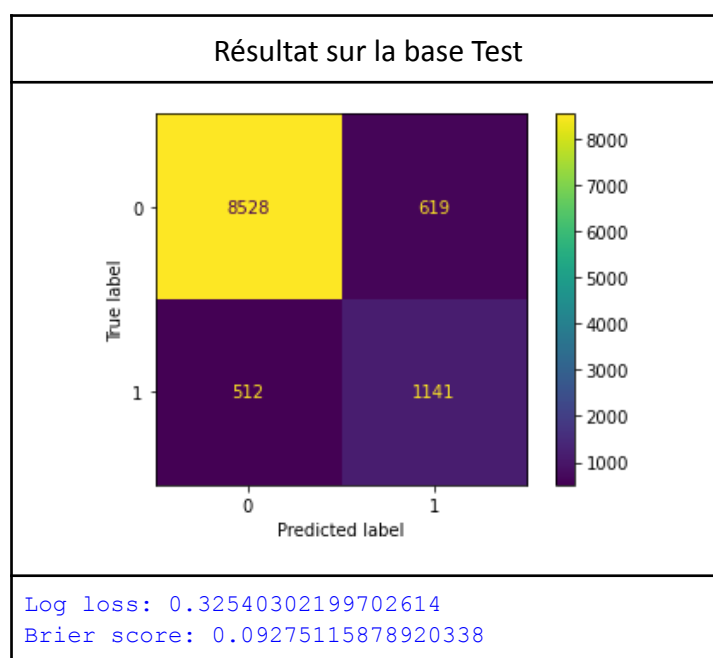
On remarque que le modèle a surappris sur la base d'entraînement.

On retient quand même les paramètres de ce Random Forest. Notons qu'on peut chercher à optimiser davantage en faisant un grid search autour des valeurs que l'on a trouvé. Cela prendrait cependant beaucoup de temps.

On remarque aussi que choisir une autre pondération que 'balanced' nous donne aussi un bon modèle.

On gardera le modèle avec une poids de 7 pour la classe minoritaire.

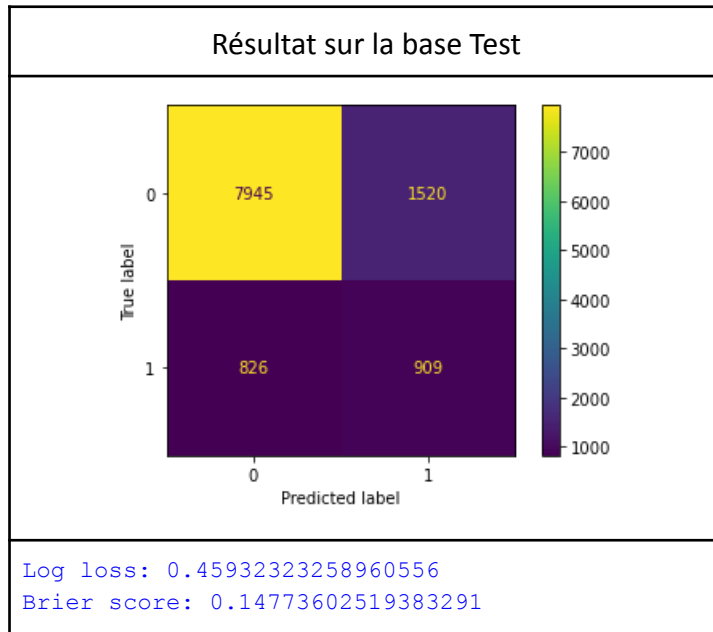
```
RandomForestClassifier(class_weight={0: 1, 1: 7}, max_depth=30,  
                        max_features=0.8, min_samples_leaf=4, n_estimators=300,  
                        n_jobs=4, warm_start=True)
```



4. Modèle XGBoost

Le modèle *XGBoost* n'est pas aussi performant que le *Random Forest* dans notre situation.

```
XGBClassifier(base_score=0.5, booster='gbtree', colsample_bylevel=1,  
              colsample_bynode=1, colsample_bytree=1, enable_categorical=False,  
              gamma=0, gpu_id=-1, importance_type=None,  
              interaction_constraints='', learning_rate=0.07, max_delta_step=0,  
              max_depth=4, min_child_weight=1, missing=nan,  
              monotone_constraints='()', n_estimators=300, n_jobs=3,  
              num_parallel_tree=1, predictor='auto', random_state=0,  
              reg_alpha=0, reg_lambda=1, scale_pos_weight=3, subsample=0.2,  
              tree_method='exact', validate_parameters=1, verbosity=None)
```



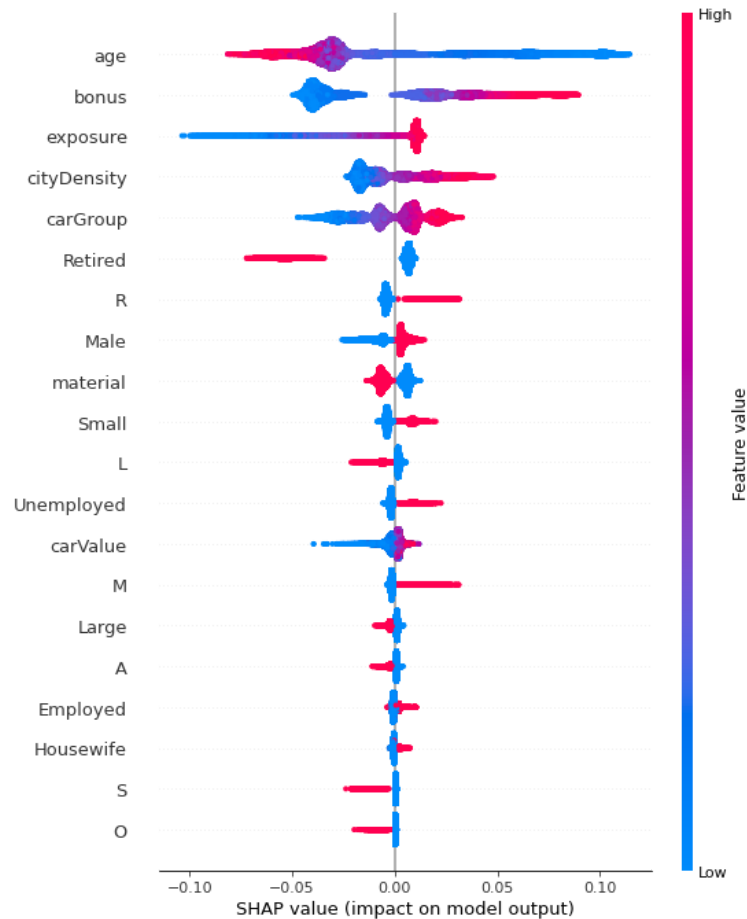
5. D'autres modèles possibles

D'autres modèles sont envisageables, parmi eux:

- une régression logistique : cependant, nous pensons que la régression ne serait pas aussi efficace que les modèles d'arbre.
- KNN
- SVM

6. Conclusion et Shapley

a. Shapley summary plot



Graphique de SHAP pour le modèle de classification XGBoost

```
XGBClassifier(base_score=0.5, booster='gbtree', colsample_bylevel=1,
colsample_bynode=1, colsample_bytree=0.3, enable_categorical=False, gamma=0,
gpu_id=-1, importance_type=None, interaction_constraints='', learning_rate=0.1,
max_delta_step=0, max_depth=6, min_child_weight=1, n_estimators=30, n_jobs=3,
num_parallel_tree=15, predictor='auto', reg_alpha=0, reg_lambda=1,
scale_pos_weight=3, subsample=0.2, tree_method='exact')
```

Ce graphique de Shapley montre que :

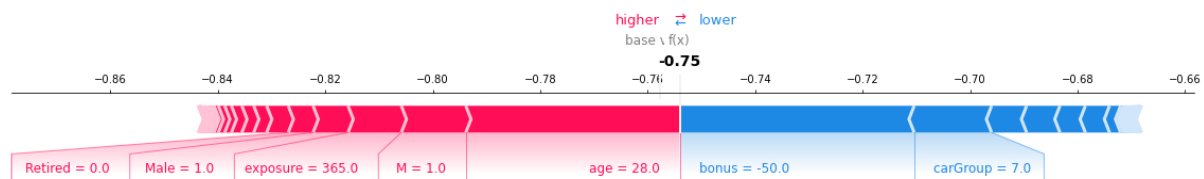
- l'âge, le bonus-malus, la densité de la ville, l'exposition, le fait d'être retraité, un homme, influencent le plus la probabilité d'avoir un accident
- en particulier, les individus jeunes ont plus de chances d'avoir un accident que les personnes âgées
- les individus avec un malus élevé ont plus de chances d'avoir un accident
- les gens avec une exposition importante ont plus de chances d'avoir un accident
- la densité de la ville augmente les chances d'avoir un accident

- les personnes en âge de travailler ont plus de chances d'avoir un accident
- les hommes ont plus de chances d'avoir un accident

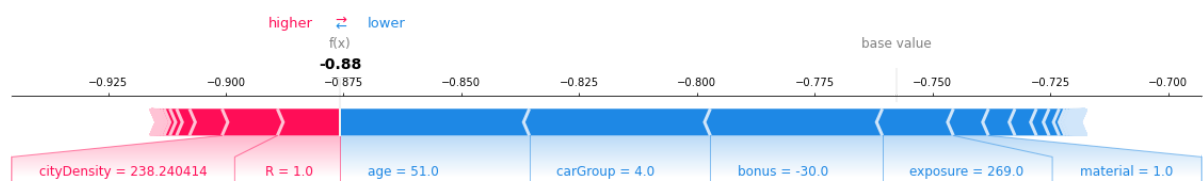
Le graphique de Shapley entier montre que beaucoup d'autres régions et sous-régions n'influencent pas ou très peu la probabilité d'avoir un accident.

b. Shapley force plot

Exemple de force plot pour l'individu 1



Exemple de force plot pour l'individu 2



c. Conclusion de la partie classification

Les graphiques de Shapley montrent l'impact de chaque variable dans le modèle *XGBoost*. On l'a réalisé sur le modèle *XGBoost*, car le calcul est plus rapide avec ce modèle qu'avec le modèle Random Forest.

Comme le modèle Random Forest est plus performant que *XGBoost* dans notre cas (et bien plus évident à "tuner"), nous allons retenir le dernier modèle de Random Forest mentionné.

IV. Régression

1. Préparation de la Régression

a. Première alternative

Nous réalisons le même traitement de la base de donnée déjà fait dans la partie **III. Classification**, sauf que nous avons cette fois-ci retiré les assurés qui n'ont pas eu de sinistres, c'est-à-dire les assurés dont la variable '**claimNumber**' vaut 0.

Nous avons aussi pris le logarithme de la variable d'intérêt '**claimValue**' pour recentrer les données et diminuer la dispersion. Nous souhaitons obtenir des valeurs homogènes afin d'améliorer la qualité de prédiction de notre modèle. Ensuite, nous avons pris l'exponentielle de la prédiction de notre variable d'intérêt pour récupérer les montants de sinistres prédits en valeur monétaire.

Comme pour la classification, nous avons utilisé la variable durée d'exposition '**exposure**' comme pondération.

Cependant, les modèles que nous avons testés, à savoir le *Random Forest* et le *XGBoost* n'ont pas de bonnes performances selon les critères R2, MAE (Mean Absolute Error) et RMSE (Root Mean Square error).

Nous allons voir comment nous avons fait pour pallier ce problème.

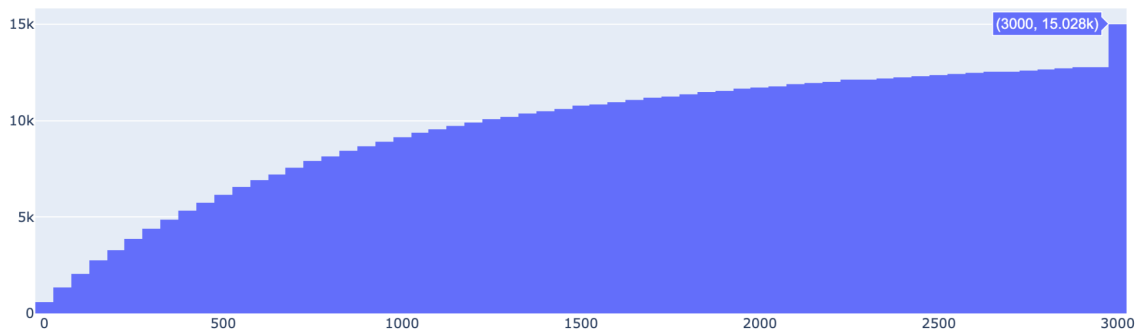
b. Deuxième alternative

Comme seulement 15% des individus dans la base de données ont eu des sinistres, le fait d'avoir enlevé les non sinistrés empêche le modèle de généraliser ce qu'il apprend. Nous décidons donc d'**inclure tous les individus, même lorsque claimValue vaut 0**.

Le fait que *claimValue* vaille 0 est implicitement donné par notre premier modèle qui prédit la probabilité que l'individu soit dans la classe 1. Avec cette approche, il faut donc **intégrer la variable que nous avons calculée dans le modèle de régression en tant que variable explicative** pour le modèle final.

De plus, le modèle a du mal à se généraliser pendant la régression. Ceci est dû au fait que les montants de sinistres sont fortement dispersés.

Nous avons choisi le seuil de claimValue de 3000. Au-delà de ce seuil, les montants de sinistres sont trop volatils. Nous faisons donc une **censure au-delà de 3000** afin de ne pas perdre cette information.



Fonction de répartition de claimValue sur la base Train avec une split de 90-10 (pas de 50)

Nous allons voir que ces trois décisions permettent d'améliorer considérablement le modèle.

2. Métrique d'évaluation de la Régression

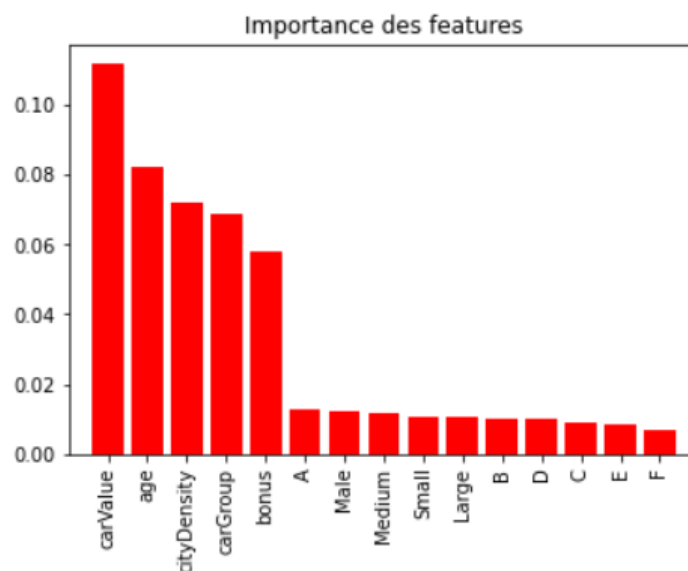
Pour évaluer la performance de notre Régression, il existe plusieurs métriques que nous pouvons utiliser comme :

1. **Mean Absolute Error (MAE)** : cette métrique calcule la moyenne des erreurs absolues. elle est souvent utilisée, car elle est facile à interpréter. Elle est sensible aux erreurs extrêmes (outliers).
2. **Mean Squared Error (MSE)** : cette métrique calcule la moyenne des erreurs au carré. Elle est souvent utilisée car elle est plus robuste face aux outliers que MAE, mais elle peut être difficile à interpréter car elle ne donne pas une mesure de l'erreur en unités réelles.

3. **Root Mean Squared Error (RMSE)** : cette métrique est simplement la racine carrée de MSE. Elle donne une mesure de l'erreur en unités réelles, ce qui peut être plus simple à interpréter que le MSE.
4. **R^2 score (coefficient de détermination)** : cette métrique mesure la qualité de la prédiction du modèle par rapport à la prédiction naïve qui consiste à toujours prédire la valeur moyenne de la variable cible. Plus le score R^2 est proche de 1, meilleure est la prédiction du modèle.

3. Modèle Random Forest pour la régression

a) Features importance



Graphique des features importance

Dans ce graphique, nous avons affiché les 15 variables les plus importantes pour notre modèle. Nous pouvons voir que la valeur de la voiture assurée, l'âge du conducteur, la densité de la ville, le groupe de voiture, et le coefficient bonus-malus sont les variables explicatives les plus importantes pour notre modèle. Ce sont les variables qui impactent le plus la tarification, ce qui est logique !

b) Paramètres de notre modèle

Le modèle a été réalisé avec le langage **Python**. Afin de choisir les paramètres qui conviennent le mieux à notre situation, nous avons effectué un tuning du nombre d'arbres '**n_estimators**' (**ntree** en R) et de la profondeur des arbres '**max_depth**' (**maxnodes**) à l'aide du module **GridSearchCv** de la bibliothèque **scikit-learn**.

Les valeurs optimales de nos hyperparamètres sont **900** pour le nombre d'arbres et **80** pour la profondeur maximale.

Dans ce modèle, nous avons utilisé le **RMSE** et le **MAE** afin d'évaluer la qualité de notre modèle.

Voici un tableau récapitulatif pour ces 2 mesures:

Mesures Resultats en Euro		
0	RMSE	3889.442777
1	MAE	1273.466015

En moyenne, nous nous trompons de 1273 € sur l'estimation du montant de sinistres qu'un individu va avoir, et donc sur sa prime.

c) Résultat : première alternative

En résumé, après avoir déterminé nos paramètres optimaux, nous avons appliqué notre modèle à la base *test*, pris l'exponentielle pour repasser en valeur monétaire, et calculé nos indicateurs de performance.

d) Résultat: deuxième alternative

Avec les traitements mentionnés en 1.b), sur le modèle censuré, on obtient ces résultats suivants avec le modèle de *Random Forest* :

Résultats sur les données censurées :

```
Mean Absolute Error test (censored): 108.34
Mean Absolute Error train (censored): 79.83
```

```
rmse test: 342.19
rmse train: 249.20
```

On peut regarder le résultat sur les données brutes pour la base train non censurée (l'apprentissage se fait sur une base censurée et est comparée à un y_test non censuré)

```
Mean Absolute Error test (uncensored): 235.06
rmse test (uncensored): 1652.54
r2 test (uncensored): 0.16
```

On voit que le MAE est multiplié par 2, mais que le rmse est multiplié par 6. Cette différence s'explique par la censure que l'on a appliquée aux données sur la base *train* et *test* (on verra ensuite que cette dégradation du RMSE et du R2 est principalement observée chez les individus qu'on a prédits comme n'ayant pas de sinistres et que la régression est plus fiable chez les individus de *claimValue* élevée).

On peut aussi s'intéresser à notre résultat comptable avec cette prédiction, en particulier la **prime fixe**.

En comparant avec les données censurées (y_test censuré), on a un résultat comptable de -138 770. Pour passer dans le positif, il faut augmenter le prix de tout le monde de 12.849.

En comparant avec les données non censurées (y_train censuré, y_test non censuré), on a un résultat comptable de:

```
-1 214 678.84
```

Pour passer dans le positif, il faut augmenter le prix de tout le monde de 140.56. Si l'on augmentait le prix à 141 euros, on aurait un résultat de 4736.83 en situation de monopole. Chaque unité supplémentaire sur chaque individu nous fait gagner en plus autant qu'il y a d'individus.

On peut aussi envisager une **prime variable** selon la valeur des sinistres prédite pour les clients, qui permettra d'attirer les bons clients et de pénaliser les mauvais clients. Cela va permettre d'abaisser davantage la prime fixe et proposer un prix compétitif aux bons clients. Par exemple, on peut proposer 125 euros en prime fixe et 10% en prime variable, auquel cas on fera un bénéfice de 5721.15 euros.

Cette analyse n'est que approximative car on n'a pas utilisé la même base train et test dans la partie classification et régression. En effet, pour la régression, on a utilisé la colonne probabilité prédite sur le modèle de classification qui a appris sur tout le modèle, ce qui risque de donner de meilleures performances qu'en réalité. **Pour être plus rigoureux (et il faut l'être car on a vu l'impact que chaque euro a sur notre résultat comptable), les deux modèles doivent être construits séquentiellement, sur une même base test et train.**

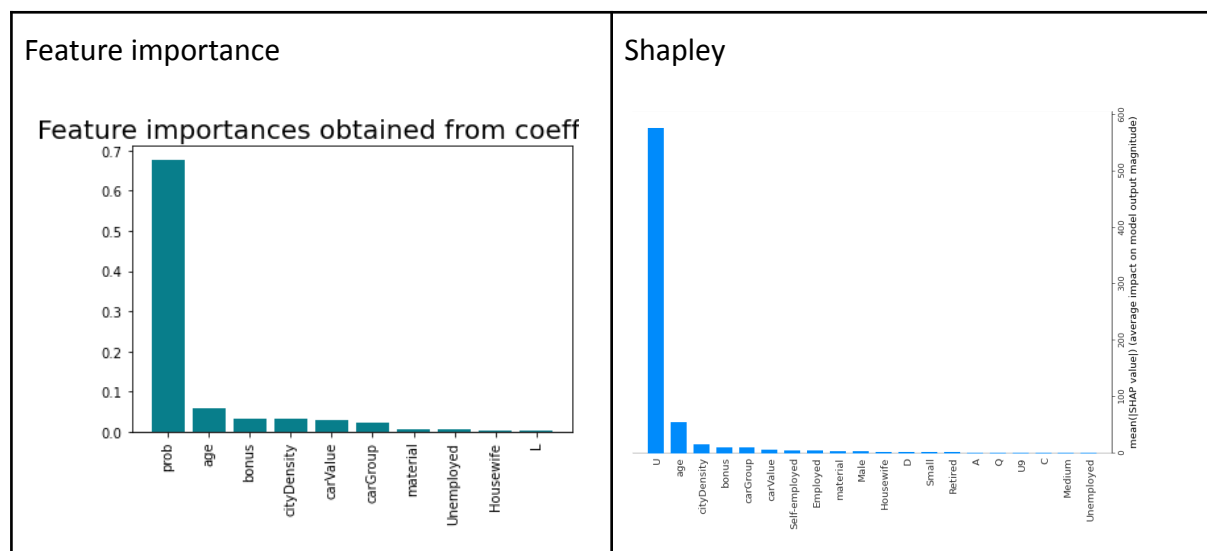
On va maintenant rassembler les deux modèles et les analyser.

Lorsque l'on imbrique les deux modèles l'un après l'autre, la qualité de la prédiction se dégrade car on accumule les erreurs des deux modèles.

```
rmse test (censored): 412.19
rmse train (censored): 234.96
r2 test (censored): 0.45
r2 train (censored): 0.93
```

En effet, le premier modèle aide beaucoup le deuxième, mais il a aussi des limites (**F1-Score** sur la classe minoritaire près de 0.68 alors qu'elle est de 0.94 sur la classe majoritaire).

Nous avons bien fait d'utiliser la variable qui donne la probabilité d'avoir un sinistre du modèle précédent car elle a été utilisée à plusieurs reprises par le deuxième modèle, même si la valeur de Shapley nous montre que cette probabilité a peu d'impact sur la variable qui nous intéresse.



Le graphique de Shapley ressemble au graphique du premier modèle de régression (qui n'inclut pas le vecteur prob et les individus qui n'ont pas de sinistres)

Tableau récapitulatif de l'analyse du modèle (sur 10800 individus de la base Train):

	censure à 3000	résultat comptable
prime fixe minimum pour avoir un résultat positif	183	3996
couple (prime fixe, prime variable à 10%) pour avoir un résultat comptable positif	171	7185

couple (prime fixe, prime variable à 20%) pour avoir un résultat comptable positif	159	10374
---	-----	-------

En abaissant la prime fixe et en augmentant la prime variable (pénaliser davantage les individus à risque), on obtient un meilleur résultat comptable. De plus, on diminue nos risques sur les hautes tranches. Cependant, si la prime variable est trop élevée, on risque aussi de perdre des clients et de ne pas avoir un assez bon résultat comptable.

Globalement, cette méthode fonctionne bien car on a traité le déséquilibre des classes dans le modèle de classification et cette information est portée par la variable explicative obtenue du modèle de classification.

Cependant, on verra par la suite que pour être compétitifs, on ne doit pas augmenter la prime variable car notre modèle de régression a bien plus de précision sur les individus avec un *claimValue* important mais a bien moins de précision sur les individus avec un *claimValue* faible.

e) Bucketing ?

i) Idée d'implémentation

Il est également possible de transformer ce problème en un problème de classification, en transformant la variable explicative *claimValue* continue en variable catégorielle par tranches de 150 euros allant de 0 à 3 000 euros par exemple.

On a testé cette méthode et on remarque que même en incluant les individus avec *claimNumber* et *claimValue* nuls, le modèle apprend assez facilement. Encore une fois, ceci est dû au fait que le déséquilibre des classes a été traité dans le premier modèle.

La prime variable demandée pourrait être par exemple égale à une moyenne pondérée (ou une espérance) des probabilités d'être dans chaque classe et la moyenne des sinistres dans chaque classe :

$$\text{prime_variable}(\text{individu } i) = \text{probabilité Classe 1 (i)} * \text{Valeur sinistre Classe 1 (i)} + \text{probabilité Classe 2 (i)} * \text{Valeur sinistre Classe 2 (i)} + \dots + \text{probabilité Classe n (i)} * \text{Valeur sinistre Classe n (i)}$$

En effet, avec la méthode précédente, la plupart des individus ont une prime qui est égale à la prime fixe. De ce fait, le prix est assez peu variable d'un individu à un autre et si les

autres équipes proposent une prime fixe bien en dessous de nous, nous n'allons pas récupérer les primes des bons risques pour nous prémunir contre les mauvais risques.

Dans cette méthode de bucketing, nous pouvons aussi cibler des groupes d'individus précis.

De la même manière, on calcule la prime fixe minimale pour avoir un résultat comptable positif.

ii) Une analyse intéressante

Lorsque l'on fait ce bucketing, on remarque que c'est sur des montants de sinistres prédits faibles que l'on court le plus de risque. Le RMSE est plus élevé sur des individus dont on pense avoir pas ou peu de sinistres (< 300), alors qu'il est bien plus bas lorsque l'on pense que l'individu a un sinistre.

Résultat par classes de taille 200 *

MAE		RMSE		Profit par classe	
100	98.666377	100	98.666377	100	23100
226.3741393632	23100	1237.731691331	23100	-413034.5870073218	47.05722097
300	47.05722097	300	47.05722097	300	23300
51.5110647137	23300	60.1741882298	23300	-437.886294813	117.67076561
500	58.835382805	500	62.9517216605	500	26100
249.1031858735	26100	1955.0319623798	26100	-19041.94477498	-7.775524
700	7.775524	700	7.775524	700	28100
63.7420132545	28100	118.9021381378	28100	2372.5243360885	12.25808274
900	12.25808274	900	12.25808274	900	29100
47.2088820556	29100	55.1653621005	29100	775.05313104	-42.17341163
1100	61.634871815	1100	65.1422028011	1100	30500
47.6335198918	30500	56.3268769872	30500	65.0978585	-12.73657
1300	12.73657	1300	12.73657	1300	36900
48.9078948808	36900	58.4924537427	36900	387.2412466292	28.37612
1500	28.37612	1500	28.37612	1500	37300
50.3486062466	37300	58.5126628593	37300	349.419133074	-10.64119187
1700	10.64119187	1700	10.64119187	1700	39300
55.024845895	39300	61.4824548925	39300	80.29589352	31.195648
1900	31.195648	1900	31.195648	1900	45900
41.7819046181	45900	47.37133702	45900	-42.28427338	75.0294594
2100	75.0294594	2100	75.0294594	2100	

* Profits par classe si on demande une prime égale au label de la classe (label 100 correspond à la tranche [0,200[, 300 à la tranche [200,400[)

Contrairement à ce que l'on a supposé précédemment, on remarque que c'est dans la classe la plus basse des valeurs prédites entre 0 et 200 euros que l'on fait plus de pertes (environ 413 000 euros). Les individus des autres classes sont prédits avec bien plus de précision.

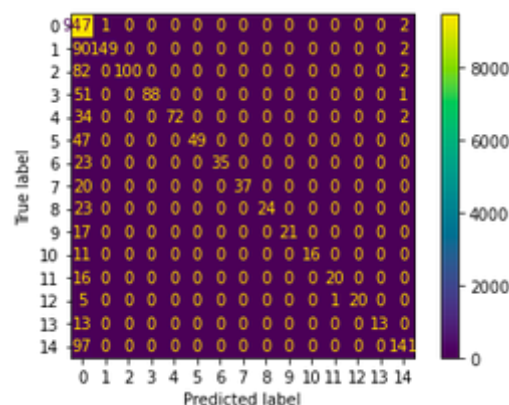
Si on n'ajuste pas nos prix, on fait une perte globale de 430 000 : les profits sur les hautes tranches peinent à compenser la perte de la première tranche.

On peut alors ajuster nos prix sur chaque classe selon le nombre d'individus présents dans la classe.

Faute de temps, on ne développera pas davantage cette méthode.

Cependant, **c'est le prix sur la tranche [0,200[qui fera réellement une différence**. Comme on a remarqué que la classe 1 apporte beaucoup de déficit et est trop risquée, on va devoir augmenter nos primes sur ces individus (même si on accepte de ne pas les avoir sur le marché)

Cette matrice montre que la première classe a le plus de chances d'être mal classée (sur donnée censurée à 3000 euros, 15 catégories progressives tous les 200 euros)



f) Shapley

Enfin, il serait aussi intéressant d'utiliser Shapley, en particulier **pour expliquer à chaque individu la prime pure qu'on leur demande**. On peut aussi utiliser Shapley **pour voir la contribution de chaque variable sur la prime pure**.

Faute de temps, nous n'avons pas pu faire cela.

g) Rendu final

Pour le rendu final, nous allons faire apprendre au modèle sur toute la base de données que nous disposons.

On gardera le modèle de régression, auquel on ajoutera environ + 60 euros à tout le monde et demandera une prime élevée aux tranches en dessous de 800 euros (+ 20 euros) et en particulier aux tranches entre 0-200 (prime à 240 euros sur toute la tranche + 60 euros affecté à tout le monde) et 400-600 (+ 50 euros + 60 euros affecté à tout le monde)

(cf. tableau e) ii))

V. Simulation du marché pour mesurer l'efficacité des algorithmes retenus face à la concurrence

1. Algorithme qui nous donne le résultat comptable

Nous avons créé une fonction qui prend en entrée le prix proposé par 6 entreprises différentes et les sinistres dans l'année pour tous les individus. L'algorithme nous donne en sortie le résultat comptable de chaque entreprise, sachant que les individus vont se diriger vers le contrat proposé par l'entreprise qui a proposé le prix le plus bas.

Nous avons voulu tester cet algorithme avec les différents prix que l'on a proposés. Cependant, faute de temps nous n'avons pas pu le faire.

Par exemple, pour avoir un meilleur prix, on peut alors faire interagir différents couples (prime fixe, prime variable à x%) comme proposé précédemment puis regarder leur résultat comptable. **On retiendra le modèle avec le résultat comptable le plus élevé tout en ayant une prime variable importante pour que le modèle soit robuste.**

a. Situation de concurrence

Le marché étant à son état initial, il est difficile de prédire comment changera notre résultat comptable face aux autres entreprises. En effet, il est possible que des groupes proposent des prix bien trop bas et déséquilibrent totalement le marché. Un prix trop élevé par un groupe ne déséquilibre pas autant le marché qu'un prix très bas.

En conclusion, en situation de concurrence, le résultat comptable est imprévisible d'une part parce qu'il y a plusieurs acteurs sur le marché et d'autre part parce que le marché est à son état initial.

Comme il est difficile d'anticiper les prix des autres équipes, nous avons décidé d'être plus conservateurs sur nos prix et proposer des prix assez élevés sur les individus à faible risque (même s'ils sont plus nombreux) et faire une petite marge sur les individus à risque élevé.

2. Commentaires et conclusion

En conclusion, ce projet de data science a été très formateur. En effet, nous avons utilisé de nouveaux modèles qui nous ont été introduits en cours, notamment le modèle de *Random Forest* et *XGBoost*. Ces modèles fonctionnent mieux que les modèles de régression classiques sur notre base de données, notamment lorsque les relations entre les variables sont non linéaires.

En plus des ressources en ligne, nous avons aussi utilisé le nouveau chatbot ChatGPT de Open AI (nous profitons du chatbot avant qu'il soit commercialisé et devienne payant).

Ce qui a vraiment fait la différence dans la qualité de la prédiction n'a pas été les modèles choisis, mais plutôt la manière dont on traite les données avant de les utiliser dans le modèle. Par exemple, pendant la classification, il faut jouer sur le poids des classes et multiplier les individus autant de fois qu'ils ont de sinistres afin que le modèle puisse se concentrer davantage sur la prédiction des 1. Ce constat justifie bien le proverbe "Garbage in, garbage out" que l'on a constaté à plusieurs reprises au cours de ce projet.

Enfin, en situation de concurrence, nous nous reposons sur la capacité de notre modèle à identifier les mauvais risques et faire des bénéfices sur ces individus.

VI. Références

[1] Decision Trees : <https://scikit-learn.org/stable/modules/tree.html>

[2] ChatGPT : <https://chat.openai.com/>

[3] Classification :

https://larevueia.fr/comment-gerer-le-desequilibre-des-classes-en-machine-learning/?fbclid=IwAR2zepoMusLEFb5eC1jw8NILWU4I5RhHVgOBGvT8dco95I_aWN-CQvziLfE