Projet Machine Learning - Groupe 2

Léo Dutertre, Axel Gardahaut, Guy Tsang

Résumé
Cet article présente les enjeux du projet de Machine Learning et les données mises à disposition. Dans un premier temps, la démarche du projet est détaillée dans les grandes lignes afin de justifier les choix effectués dans le traitement et la construction des modèles de prédiction puis dans un second temps, le fonctionnement de certains modèles (régression logistique pénalisée et Extreme Gradient Boosting) sera explicité (intuition, mécanisme, avantages et inconvénients). Une partie s'attardant sur les résultats conclura cet article.

1 Introduction

Une société d'assurance souhaite affiner sa capacité à tarifer ses contrats automobiles avec ses clients. L'objectif est de faire payer à chaque assuré son « juste prix ». Ainsi, à partir d'un jeu de données sur ses clients, le but sera de construire un modèle qui prédit, pour chaque assuré, sa probabilité de déposer une réclamation au cours de la prochaine année. Plus cette probabilité est élevée, plus la tarification sera élevée pour l'assuré.

Ceci constitue un problème classique de modélisation aboutissant à un score. La métrique d'évaluation des prédictions devra être pertinente à cette problématique, sachant que la base de données fournie présente un déséquilibre de la cible.

Le but de cet article scientifique est de présenter les données, la démarche du projet, les modèles de prédiction utilisés et enfin, les résultats obtenus.

2 Données

Les données sont fournies par la société d'assurance, comportant une base d'apprentissage et de test. L'apprentissage et la validation des modèles doivent se faire sur la première base tandis que la seconde base doit servir uniquement à tester la performance du modèle retenu. Des valeurs manquantes sont présentes dans les deux échantillons.

Parmi les prédicteurs, on retrouve :

- des variables concernant l'assuré en personne ("ind"),
- des variables concernant la région de l'assuré ("reg"),
- des variables concernant la voiture de l'assuré ("car"),
- des variables calculées ("calc").

La variable cible ("target" dans la base) est binaire et indique si une réclamation a été déposée ("1") par l'assuré ou non ("0"). Cette variable est déséquilibrée, avec moins de 4% de labels positifs. Chaque ligne de la base de données correspond à un assuré automobile. Les intitulés des colonnes sont anonymisées.

1. Gradient Boosting Machine

3 Démarche du projet

3.1 Définition du cadre d'analyse

L'objectif de ce problème de classification est de réaliser un scoring des clients. Le score correspond à la probabilité pour chaque client de déposer une réclamation dans l'année à venir. La métrique retenue pour évaluer la performance prédictive des modèles est celle du Coefficient Normalisé de Gini. Le coefficient de Gini permet de comparer la proportion cumulée des labels positifs prédits avec la proportion théorique.

Le coefficient de Gini est étroitement lié à l'aire sous la courbe ROC (AUC) :

$$\text{Gini} = 2 \times \text{AUC} - 1$$

Ainsi, maximiser le coefficient (normalisé) de Gini revient à maximiser l'AUC.

3.2 Statistiques Descriptives

Les statistiques descriptives sont faites pour vérifier la nature des données traitées, la présence de valeurs manquantes, la présence de valeurs aberrantes et les distributions. De plus, on repère les types de variables (continues, catégorielles, ordinales, etc.) pour spécifier le traitement au cas par cas.

L'inférence faite à partir de modalités rares (moins de 5% en général) n'est pas robuste et peut conduire à du surapprentissage. Les statistiques descriptives sur les variables catégorielles permettent de détecter ces modalités.

Enfin, l'étude des variables continues permet de prévoir l'utilité de transformations (normalisation ou standardisation).

3.3 Benchmark d'ouverture

Il est intéressant de placer un benchmark afin d'avoir une valeur de la métrique objectif (Coefficient Normalisé de Gini) à dépasser. Le score obtenu par un Light GBM [1] est généralement élevé et rapidement obtenu. Celui-ci est de : 0.2719 par validation croisée sur 5 blocs (5fCV). Bien que le score obtenu est sujet à fluctuer, il est intéressant de refaire un Light GBM

après chaque étape du traitement afin de voir si les modifications apportées augmentent ou diminuent le score.

3.4 Pré-traitement des données

Le pré-traitement des données consiste essentiellement à la gestion des valeurs manquantes. Plusieurs variables ont été identifiées dans les statistiques descriptives (figure 1). Selon le taux de valeurs absentes, la manipulation appliquée sera différente.

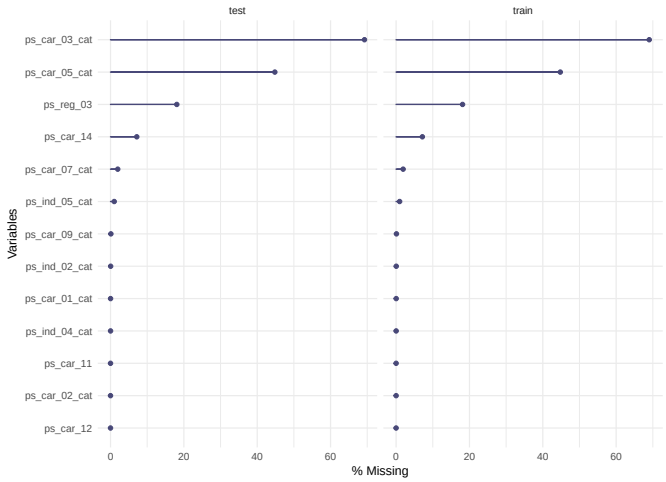

Figure 1 – Valeurs manquantes par variable

Deux colonnes présentent trop de valeurs manquantes (plus de 40%) pour une imputation sophistiquée, ainsi, elles sont remplacées par une valeur spéciale (-999 par exemple) pour conserver l'information issue de l'absence de valeur. En effet, en croisant ces variables avec la cible, il est noté que le taux croisé d'effectif est significativement différent à ceux des valeurs normales.

Pour le reste des variables, une imputation par forêt aléatoire est faite. Le procédé d'imputation est le suivant :

- Utiliser uniquement l'échantillon d'apprentissage pour construire la forêt aléatoire.
- Modéliser la variable étudiée en s'assurant qu'il s'agisse du bon type d'arbres (classification ou régression) en utilisant les autres variables comme régresseurs.
- Prédire les valeurs de la variable étudiée pour l'ensemble des observations manquantes des deux échantillons (apprentissage et test).
- Remplacer les valeurs manquantes et s'assurer qu'il n'y a plus de valeurs manquantes pour la colonne traitée.

Suite aux imputations faites, le Light GBM de référence renvoie un Gini de 0.2721, toujours par validation croisée sur 5 blocs. On note une augmentation par rapport au benchmark bien que la différence soit trop faible pour en tirer des conclusions.

2. Analyse Factorielle de Données Mixtes

3.5 Sélection des variables

Le Light GBM effectué après les imputations permet également d'identifier l'importance des variables dans le pouvoir prédictif du modèle. L'importance d'une variable peut être mesurée de différentes manières. Celle adoptée ici est le « Gain », i.e. la contribution de la variable au modèle. Ainsi, plus une variable contribue au pouvoir prédictif d'un modèle, plus son « Gain » associé sera élevé. Arbitrairement, on garde la première moitié des variables les plus contributives au modèle.

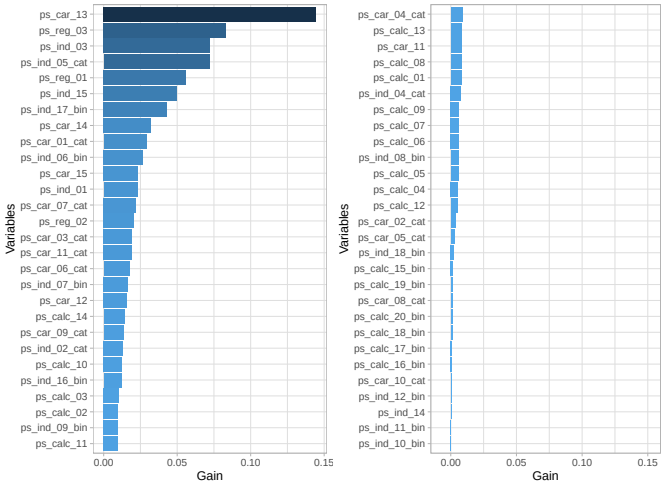

Figure 2 – Importance des variables

En plus des importances issues du Light GBM, une forêt aléatoire (non paramétrée de façon optimale) est construite pour réaliser une seconde liste d'importance en termes de « permutation » (notion équivalente au « Gain »). De même, cette seconde liste est issue de la première moitié des variables les plus importantes au modèle. L'union de ces deux listes constitue la liste finale des variables retenues pour la suite. Au total, ce sont 35 régresseurs sur les 57 initiaux qui sont retenus.

3.6 Traitement des données

Lors de la phase des statistiques descriptives, plusieurs variables présentaient des modalités rares (effectif inférieur à 5%). Celles-ci doivent être fusionnées pour pouvoir inférer des résultats robustes (par AFDM [2] ou tableaux de contingence).

La stratégie de fusion peut passer par une projection des modalités d'une variable sur le premier plan factoriel d'une AFDM. Chaque modalité rare sera alors associée à la modalité fréquente la plus proche d'elle (et si possible projetée dans la même direction), ou un ensemble de modalités rares peuvent se regrouper pour former un groupe supplémentaire (clusters).

La stratégie de fusion peut également passer par des tableaux de contingence (figure 3) qui croisent les variables problématiques avec la cible. Les modalités rares sont fusionnées soit entre elles si elles se comportent de façon similaire, soit avec une modalité fréquente si le comportement s'y rapproche.

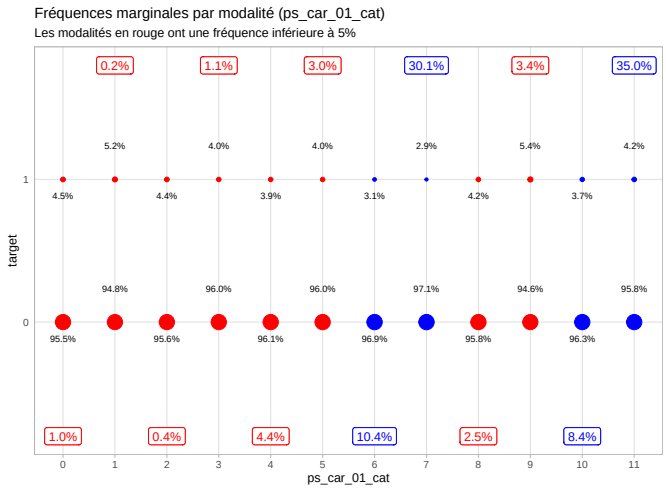

Figure 3 – Exemple de projection de modalités par croisement

* * *

Une variable présente un grand nombre de modalités (104). Plutôt que de fusionner ces modalités, on peut s'en servir comme régresseur unique d'une régression logistique avec la variable cible comme variable à expliquer. Les modalités seront alors remplacées par les coefficients estimés et la variable deviendra alors numérique et continue.

Le Light GBM fournit un score de 0.2729 (5fCV) pour la stratégie par AFDM et de 0.2757 (5fCV) pour la stratégie par tableaux croisés. Le score ayant fortement augmenté pour la seconde stratégie, elle sera retenue pour la suite.

3.6.1 Normalisation des données

Certaines variables continues ont une distribution qui se rapprochent d'une distribution normale. Il serait intéressant de tenter de les normaliser à travers différentes méthodes (box-cox, logarithme, racine carré).

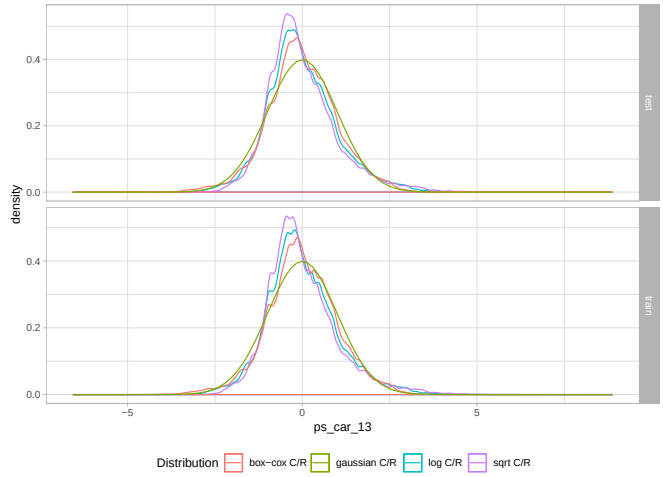

Figure 4 – Exemple de transformation

Il est notable que la transformation de box-cox est la meilleure pour normaliser cette distribution (figure 4). Cette étude est menée pour les variables continues.

En réutilisant un Light GBM pour mesurer les conséquences des normalisations faites, il est noté que la métrique diminue, suite à chacune des deux stratégies de fusion de modalités. On passe respectivement à un Gini de 0.2729 (AFDM) et 0.2752 (tableaux croisés). Ces diminutions sont minimes mais la normalisation des variables n'est pas obligatoire. Par conséquent, cette étape est laissée de côté pour la suite.

À la fin de cette étape, le traitement se résume à la fusion de modalités en utilisant des tableaux de contingence.

3.7 Étude des corrélations et dépendances

L'objectif de cette étape est d'étudier les liens entre variables. On évalue les corrélations et les dépendances entre variables puis avec la cible. Ceci permet de détecter des problèmes de colinéarité et ou dépendance entre prédicteurs :

- les liens entre variables continues se font grâce aux corrélations,
- les liens entre les variables continues et la cible se font grâce à un test d'ANOVA [3],
- les liens entre variables catégorielles (et la cible) se font grâce aux V de Cramer [4].

Les variables qui dépassent les seuils arbitrairement fixés ou qui ne passent pas le test d'ANOVA sont laissées de côté. L'étude n'a pas abouti à de suppression de variables.

3.8 Paramétrisation des modèles

Une dizaine d'algorithmes sont testés pour tenter d'avoir le meilleur pouvoir prédictif. On y trouve des modèles de base (Analyse Discriminante Linéaire (LDA), K plus proches voisins (KNN), régression logistique (pénalisée ou non), Machine à Vecteurs de Support (SVM)), puis des modèles de bagging (Forêt Aléatoire) et de boosting (ADAboost, GBM, XGBM, LGBM).

L'ensemble des modèles ensemblistes et la régression logistique pénalisée sont optimisés selon leurs paramètres de pré-apprentissage [5].

Pour chaque modèle, la question du ré-échantillonnage est traitée. Pour l'ensemble des modèles, les stratégies de down-sampling, up-sampling et SMOTE sont testées en plus du cas sans resampling.

4 Méthodes utilisées

Les deux méthodes à traiter a minima pour notre groupe dans le cadre de ce projet de classification sont la régression logistique pénalisée et le XGboost dont les principes sont radicalement différents.

3. Analysis of Variance
4. Le V de Cramer est la statistique du χ^2 standardisée de façon à ce qu'elle varie entre 0 et 1
5. Les algorithmes KNN et SVM ne sont pas optimisés faute du temps d'exécution nécessaire au calcul de matrices immenses de distance

4.1 Régression pénalisée

La pénalisation de la métrique évaluant la prédiction est un principe applicable à toutes les méthodes d'estimation où l'on a des combinaisons linéaires de variables avec des coefficients à estimer (réseaux de neurones, SVM linéaire, etc).
L'idée étant que les hypothèses classiques de la régression linéaire permettent d'obtenir le meilleur estimateur non biaisé i.e. l'estimateur sans biais de variance minimale. Dans les faits on obtient un estimateur de faible biais mais de variance plus élevée.

Or, en considérant l'écart quadratique moyen (EQM) qui est la fonction de perte la plus largement utilisée on a :

$$\mathbb{E}\left[(y^* - \hat{y}^*)^2\right] = \sigma^2 + (\mathbb{E}\left[\hat{y}^*\right] - y^*)^2 + \mathbb{E}\left[(\hat{y}^* - \mathbb{E}\left[\hat{y}^*\right])^2\right]$$

On a respectivement σ^2 la variance incompressible de la cible Y, le biais et la variance de la prévision. Ainsi si on s'autorise un certain biais, on peut imaginer qu'il est possible de réduire plus que proportionnellement la variance et ainsi gagner en pouvoir prédictif. Pour illustrer ceci, en utilisant les notations de l'équipe scikit-learn on note alors ce problème :

$$\left| \begin{array}{l} \min\limits_{\beta_1,\ldots,\beta_p} \sum\limits_{i=1}^{n} \left(y_i - \sum\limits_{j=1}^{p} \beta_j z_{ij} \right)^2 + \lambda\, R(\beta) \\ \text{sous contraintes : } R(\beta) \leq \tau \end{array} \right.$$

où l'on aura pris soin de centrer et réduire nos variables afin de leur accorder la même importance.

Il y a donc 3 types de pénalisation utilisés régulièrement :

1. en norme L^2 on parle alors de régression ridge : $R(\beta) = \sum_{j=1}^{p} \beta_j^2$
2. en norme L^1 on parle alors de régression LASSO : $R(\beta) = \sum_{j=1}^{p} |\beta_j|$
3. ElasticNet qui est une combinaison linéaire des 2 : $R(\beta) = \alpha \sum_{j=1}^{p} |\beta_j| + (1-\alpha) \sum_{j=1}^{p} \beta_j^2$

Notons que λ est un paramètre du modèle qu'il faut calibrer également.
On a une relation inverse entre τ et λ. Lorsque $\lambda = 0$ on retrouve les MCO et plus λ est grand, plus les coefficients sont contraints (plus de biais, moins de variance). On peut voir les coefficients se déformer en fonction de λ.
La régression ridge est plus souple que la régression LASSO qui fait converger beaucoup plus vite les coefficients vers 0 ce qui fait qu'on s'en sert comme méthode de sélection de variable.
Néanmoins, on obtient au maximum N variables prédictives (identifiabililité) ce qui peut être préjudiciable quand le nombre de variables est très supérieur au nombre d'observations ($p \gg N$).
De plus, elle choisit arbitrairement une variable parmi un groupe de variables corrélées quand la regression ridge permet de pondérer les influences.
L'Elastic Net permet de combiner les avantages des deux méthodes en contrepartie d'une complexité accrue.

Pour déterminer λ on procède généralement par apprentissage-test lorsque c'est possible ou par validation croisée K-folds lorsque la base est petite, en minimisant le RMSE. Des formules basées sur des hypothèses simplificatrices existent également.
En régression ridge, une astuce permet d'accélérer drastiquement les calculs en validation croisée Leave-One-Out grâce à une décomposition en valeurs singulières de X.
Par ailleurs, de manière assez naturelle on obtient l'estimateur avec $\hat{\beta}_{\text{Ridge}} = (X'X + \lambda I_p)^{-1} X'y$

Pour la régression LASSO, on n'a pas de formulation explicite de β (car la fonction valeur absolue n'est pas différentiable) ainsi on procède de manière itérative (LARS, Forward Stagewise). On fait varier de manière sélective les coefficients des variables. En grande dimension, on privilégiera la descente de gradient ou ses variantes pour l'estimation de β.

4.1.1 Régression logistique pénalisée

Dans le cadre classique de la discrimination binaire par la régression logistique, on suppose (x_i, y_i) *iid* d'une même distribution inconnue.
On modélise $\mathbb{P}(Y = 1|X = x_i) = \frac{1}{1+e^{-(\beta' x)}} = p_i$.
Pour le cas particulier de la régression logistique pénalisée on cherche à minimiser :

$$-\frac{1}{n} \sum_{i=1}^{n} \left[y_i \ln p_i + (1 - y_i) \ln (1 - p_i) \right] + \lambda \times R(\beta)$$

Les principes de la pénalisation sont les mêmes que précédemment, seulement on ne pénalise pas la constante mais elle doit faire partie de l'estimation.

4.1.2 Performance

Resampling	down	up	SMOTE	aucun
Gini Normalisé[a]	0.2566	0.2576	0.2483	0.2573

a. obtenu par validation croisée 5 blocs

4.2 XGboost

L'algorithme XGboost est l'abréviation de e**X**treme **G**radient **boost**ing, approche introduite par Friedman.
Comme la plupart des méthodes basées sur des arbres de décisions, il peut être utilisé en classication comme en régression.
L'approche combine 2 mécanismes : le boosting et la descente de gradient.
Le boosting est une méthode ensembliste qui consiste à agréger des classifieurs élaborés séquentiellement sur un échantillon d'apprentissage dont les poids des individus sont corrigés au fur et à mesure, les individus mal classés se voyant affecter un poids plus important. Les classifieurs sont pondérés selon leurs performances.
D'autre part, la descente du gradient est une technique itérative qui permet d'approcher la solution d'un problème d'optimisation.

En apprentissage supervisé, la construction du modèle revient souvent à déterminer les paramètres (du modèle) qui permettent d'optimiser une fonction objectif.
Enfin, outre les deux mécanismes précédents, l'approche possède aussi des paramètres liés à l'utilisation d'arbres comme classifieurs.
Ainsi, l'algorithme XGboost dépend d'un grand nombre de paramètres :

1. Caractéristiques des arbres individuels : Profondeur T, effectifs minimums de coupe
2. Constante d'apprentissage η
3. Nombre d'arbres K
4. Taux d'échantillonnage des individus β
5. Échantillonnage des variables

La prévision se base sur K arbres. On construit donc itérativement ces K arbres, à chaque étape on ajoute celui qui minimise une fonction objectif régularisée qui est la somme d'une fonction de perte et d'une fonction de régularisation.

$$\mathcal{L} = \sum_i l\left(\hat{y}_i, y_i\right) + \sum_k \Omega\left(f_k\right)$$

La fonction de perte est généralement la déviance binomiale en classification binaire :

$$-\frac{1}{n}\sum_{i=1}^{n}\left[y_i \ln p_i + (1-y_i)\ln\left(1-p_i\right)\right]$$

On utilise la descente de gradient pour optimiser cet objectif à chaque étape.
On peut ajouter de l'aléa dans l'échantillon test en utilisant seulement un échantillon des données pour construire les arbres. Ceci a l'avantage d'accélérer les calculs et de se prémunir contre le surapprentissage.
L'algorithme propose également une gestion efficace des valeurs manquantes.
On optimise les paramètres de l'algorithme par grid search.
Il est évident que plus le nombre d'itérations est grand, plus on s'approche de la solution optimale, cependant celà s'effectue au détriment du temps de calcul.
Le compromis pour $\eta \in]0,1[$ est réalisé entre vitesse de convergence et surapprentissage.
Si l'algorithme est très efficace, souple avec le choix des fonctions de coûts, adaptables à différents problèmes, permettant de prendre efficacement en compte des interactions non linéaires cela reste un modèle difficilement interprétable (bien qu'on puisse calculer des mesures d'importance des variables), lourd en mémoire et intensif avec beaucoup de paramètres à optimiser et un risque de surapprentissage qui peuvent rendre sa mise en œuvre efficiente moins aisée.

4.2.1 Performance

Resampling	down	up	SMOTE	aucun
Gini Normalisé[a]	0.2678	0.2728	0.2687	0.2778

a. obtenu par validation croisée 5 blocs

Le modèle sans resampling se classe premier en validation sans compter les modèles de stacking construits par la suite.

5 Résultats

Plusieurs modèles ont été paramétrés par grid search pour chaque stratégie de resampling puis évalués en validation croisée 5 blocs. La figure 5 répertorie les résultats de performance obtenus.

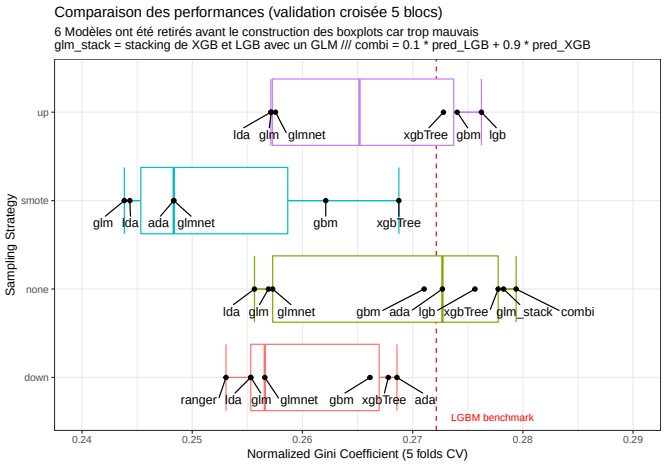

Figure 5 – Performance de validation des modèles

Certains modèles construits tels que les arbres de décision ne figurent pas dans la figure parce que leur performance est trop faible. Les modèles de stacking (glm_stack et combi) ont été construits en choisissant les deux modèles les plus performants. Le méta-classifieur du stacking est une régression logistique et le modèle "combi" correspond à un classifieur qui combine linéairement les scores issus de la validation croisée des modèles.

Au final, le modèle combiné dépasse le reste des modèles avec un coefficient normalisé de Gini en validation croisée de 0.2794. L'application à l'échantillon test renvoie une valeur de 0.28809.

Références

[1] Abu-Rmileh, A. The multiple faces of 'feature importance' in xgboost. Towards data science, 8 Février 2019.
[2] Chen T., Guestrin C. Xgboost : A scalable tree boosting system, university of washington.
[3] Friedman, J. Greedy function approximation : A gradient boosting machine, 2001.
[4] Gandhi, R. Boosting algorithms : Adaboost, gradient boosting and xgboost. Hacknernoon, 5 Mai 2018.
[5] Hale, J. Smarter ways to encode categorical data for machine learning. Towards data science, 11 Septembre 2018.
[6] Hastie T., Tibshirani R., and Friedman J. The elements of statistical learning, 2001.
[7] Juhi. Gini coefficient and lorenz curve explained. Towards data science, 6 Mars 2019.
[8] Rakotomalala.R. Université de Lyon 2, http ://eric.univ-lyon2.fr/~ricco/cours/slides/regularized_regression.pdf.
[9] Rstats on pi : predict/infer. Be aware of bias in rf variable importance metrics. R-bloggers, 19 Juin 2018.