Année universitaire 2015-2016

Projet - Big Data et Assurance

Professeur
M. Olivier Lopez

Élèves
M.Amaury d'Aligny
Mlle. Estelle Blumereau
M. Alexandre de La Morinerie
M. Henri Qiu
M. Charles Tremblay
Mlle. Léa Vitrac

Table des matières

Table des figures

Liste des tableaux

Introduction

À la fin de l'année 2012, la compagnie d'assurance BNP Paribas Cardif a lancé un plan de transformation dans le but de faire d'elle un « assureur digital » à l'horizon 2015. Chaque jour, des milliers de clients potentiels se rendent sur les sites et applications en ligne pour consulter leurs avoirs, déclarer ou suivre un sinistre, etc. L'enjeu qui en découle est clair : augmenter considérablement le taux de conversion des internautes qui sont des clients potentiels, et améliorer la satisfaction client par le biais des canaux digitaux. En outre, l'internet des objets se révèle être un moyen de mieux connaître les habitudes, les attentes et les besoins des clients, afin d'améliorer les offres et de les rendre d'autant plus ciblées et performantes.

C'est dans ce contexte que BNP Paribas Cardif cherche à optimiser son processus de suivi et de gestion des sinistres. Ce dernier nécessite différents niveaux de vérification avant que le remboursement ne soit approuvé et versé. En utilisant les innombrables données recueillies *via* les canaux digitaux, ainsi que les techniques statistiques et informatiques désormais accessibles, le processus de gestion des sinistres doit être plus adapté. La compagnie d'assurance a donc décidé de mettre à disposition du public une base de donnée anonymisée dans le cadre d'un challenge Kaggle. Cette dernière comporte deux types de sinistres : ceux pour lesquels l'approbation de paiement peut être accélérée, conduisant ainsi à des remboursements plus rapides, et ceux pour lesquels des renseignements complémentaires sont nécessaires avant approbation. La problématique posée par BNP Paribas Cardiff est alors simple. Il s'agit de prédire la catégorie à laquelle appartient un sinistre, lequel est caractérisé par des motifs recueillis un peu plus tôt dans le processus de gestion des sinistres. Ainsi, la compagnie d'assurance sera à même de faciliter le traitement des sinistres, d'accélerer le processus, tout en faisant gagner du temps au client.

Outre son aspect stratégique, le challenge permettra également aux participants de se familiariser avec le traitement des données massives comportant des centaines de milliers d'individus et des centaines de variables. Ce sera également l'occasion de tester des méthodes statistiques de prédiction sur la variable cible *target*, de définir leur limites, et de les comparer entre elles. Comment prédire au mieux la catégorie de sinistre ? Quelle méthode utiliser par rapport aux autres ? C'est autour de cet axe d'optimisation de méthode que nous avons orienté notre étude. Dans un premier temps, une rapide étude de la base passant par des statistiques descriptives a été menée. Ensuite, nous avons cherché à mettre en place des méthodes d'imputation des valeurs manquantes en utilisant des techniques statistiques robustes. Enfin, divers algorithmes de prédiction ont été appliqués pour tenter de prédire la variable cible. Les résultats ont alors été comparés, sans perdre de vue un esprit critique par rapport aux méthodes utilisées.

Chapitre 1

Une base impropre pour les techniques classiques

1.1 Présentation des données et statistiques descriptives

1.1.1 Des bases d'apprentissage et de test aux caractéristiques analogues

Deux bases de données, issues d'une même base initiale, sont disponibles : une base d'apprentissage et une de test. Elles comportent chacune près de 115 000 observations.

Chacune d'entre elles correspond à un sinistre précisément identifié par la variable *ID*.

De plus, chaque sinistre est décrit par 131 variables explicatives (notées respectivement V suivi du numéro de la variable). Parmi ces variables, nous distinguons 19 variables catégorielles, 4 variables entières et 108 variables numériques. Aucune des variables catégorielles n'est ordinale.

Enfin, la variable *target*, qui est la variable que nous souhaitons prédire sur la base de test, est présente dans la base d'apprentissage. Celle-ci vaut 1 dans 76% de la base de données, ce qui signifie que 76% des réclamations pourraient être accélérées puisqu'il n'est pas nécessaire de demander de justificatifs supplémentaires. Si l'objectif du concours Kaggle était de prédire la probabilité pour que la variable target vaille 1 dans la base de test, il nous est cependant impossible de calculer des performances de modèles sans connaître la variable target. Ainsi, la base test ne sera pas utilisée dans la suite du rapport, et nous diviserons la base d'apprentissage en deux parties (à hauteur de 2/3 des données pour la nouvelle base d'apprentissage, et 1/3 pour la nouvelle base de test).

Les données étant rendues publiques par BNP Paribas Cardif, les variables ont été anonymisées : aucune information n'est fournie sur leur nature. Il est alors impossible de raisonner a priori sur leur signification pour rassembler des variables par exemple afin d'améliorer la qualité d'un modèle.

1.1.2 La spécificité des données

La spécificité de la base de données réside dans le fait que de très nombreuses variables ont beaucoup de valeurs manquantes. Dans la base d'apprentissage, seules 12 variables sont entièrement renseignées. Il est possible de dégager des groupes de variables en fonction de leur proportion de valeurs manquantes comme l'illustre le tableau 1.1. Cette particularité sera étudiée de manière plus approfondie, avec la notion de *pattern*, dans la section 2.1.2.

Proportion de valeurs manquantes	0 %	]0 % ; 0.6 %]	[3 % ; 6 %]	[42 % ; 45 %]	[48 % ; 53 %]
Nombre de variables	12	14	3	100	2

Tableau 1.1 – Des groupes de variables en fonction de leur proportion de valeurs manquantes. *Base de données d'apprentissage.*

1.1.3 Des données préalablement traitées

Il ne semble pas y avoir de données aberrantes dans la base. Les variables quantitatives ont très certainement été normalisées avant la soumission des données sur Kaggle : elles sont presque toujours comprises entre 0 et 20. Et la valeur maximale est souvent très éloignée du troisième quartile c'est-à-dire que de nombreuses variables sont étalées à droite. Concernant les variables catégorielles, certaines comportent beaucoup de modalités. Par exemple, la variable v22 ne sera pas exploitée dans la suite car elle en possède trop. En effet, elle détient plus de 18 000 modalités ce qui représente moins de 8 individus par modalité. L'intégration de cette variable à un modèle ne serait pas judicieuse car elle conduirait quasiment à individualiser les prédictions et le modèle perdrait son pouvoir de généralisation.

1.2 Méthodes robustes aux valeurs manquantes

Les valeurs manquantes sont le point aveugle de l'analyse statistique. En effet, presque toutes les méthodes usuelles supposent une base complète. Les arbres font figure d'exception notable. En conséquence, ils peuvent être appliqués à la base initiale.

1.2.1 Arbres de classification et de régression

Plusieurs algorithmes sont disponibles pour ce type de procédure. Nous présentons ici l'algorithme CART (*Classification And Regression Trees*), l'un des plus connus.

A chaque pas, l'algorithme cherche la variable explicative qui introduit le plus d'hétérogénéité parmi les individus. Il faut fixer un seuil aux variables continues, séparer les modalités en deux groupes pour les variables catégorielles. Avec ce critère, la base est séparée en deux sous-populations, à la fois aussi homogènes (par rapport à la variable dépendante) en leur sein et aussi différentes entre elles que possible. Un arbre est produit dont les noeuds sont autant de critères de décision basés sur les variables explicatives. L'algorithme s'arrête lorsqu'il trouve des groupes triviaux (une seule valeur de la variable dépendante pour toute la feuille). L'arbre complet est généralement de très grande taille : il est élagué pour être exploitable.

Pour supporter les valeurs manquantes, CART construit des *surrogate nodes*. Si la variable d'un noeud de décision donné est manquante pour un individu, il se réfère à une autre variable, non manquante pour l'individu, qui construit à peu près les mêmes sous-échantillons. Cependant, la présence de valeurs manquantes altère la capacité prédictive de l'arbre.

Cette robustesse au NA est le principal avantage de l'arbre de décision. Il est également facile à interpréter, même pour le néophyte. Cependant, l'arbre obtenu est généralement très sensible à l'échantillon d'apprentissage - même si sa capacité prédictive est stable. Recourir à une forêt aléatoire, qui multiplie des arbres simple via du boosting, peut (au prix d'une complexification du modèle) diminuer cet inconvénient

1.2.2 XGboost, puissant algorithme de boosting

Les algorithmes de boosting ne sont pas gênés par la présence de valeurs manquantes. Dans leur principe, le premier pas considère un modèle de prédiction extrêmement simple, dit *weak learner*. Les individus sont repondérés selon qu'ils sont bien prédits ou non par ce *weak learner*. Puis on itère, le *weak learner* est appliqué à la base repondérée. Par améliorations successives, le modèle final devient très performant.

XGboost (*eXtreme Gradient Boosting*) est un algorithme de boosting particulièrement puissant. Les performances de prédiction sont élevés, les temps de calculs courts et il s'adapte à un grand nombre de données. Ces caractéristiques en font un allié de poids pour le statisticien.

Cependant, il fonctionne en boîte noire incompréhensible au béotien. De plus, s'il est *de facto* robuste aux valeurs manquantes, il est très délicat de reconstituer quel traitement il leur applique. Aussi, une imputation personnalisée des valeurs manquantes pourrait améliorer ses performances.

Chapitre 2

Traitement des valeurs manquantes

2.1 Imputation nécessaire des valeurs manquantes

2.1.1 Typologie des données manquantes

L'occurrence des données manquantes est un problème fréquemment rencontré dans l'analyse statistique d'une base de données. Les ignorer peut entraîner à la fois des biais importants d'estimation des paramètres d'intérêt et une perte de précision. Si plusieurs approches sont possibles pour remédier à ce problème, la méthode d'imputation des données manquantes repose avant tout sur l'identification de leur type : une valeur manquante est-elle due à une erreur d'inattention ou est-elle la conséquence de la logique d'un questionnaire ? Quelle est la cause de l'absence d'une donnée, et quels sont les liens avec les autres variables ? On dénombre généralement trois types de données manquantes : celles *Missing Completely At Random* (MCAR), celles *Missing At Random* (MAR), et celles *Non Missing At Random* (NMAR). Cette typologie a notamment été développée par Little & Rubin en 1987.

Missing Completely At Random - MCAR

Les données de type MCAR désignent les valeurs manquantes d'origine complètement aléatoire. C'est le cas lorsque la probabilité d'absence de données pour une variable Y est indépendante de la variable Y elle-même ou des autres variables X de la base.

Afin d'illustrer ce propos, prenons l'exemple d'une enquête cherchant à identifier les déterminants du revenu, tels que la variable âge. L'hypothèse MCAR n'est pas respectée dans le cas où les sondés qui ne répondent pas à la question sont en moyenne plus jeunes que ceux y ayant répondu : dans ces conditions, l'absence de réponse à l'enquête dépend de l'âge, la probabilité d'absence de réponse n'est donc pas la même pour tous les sondés.
Prenons maintenant le cas suivant : chaque sondé décide de répondre à la question du revenu en lançant au préalable un dé non truqué : si la face 1 apparaît, alors le participant ne donne pas d'élément de réponse. Dans le cas inverse, il doit donner son revenu. Il s'agit ici d'une situation MCAR, la probabilité d'absence de réponse est la même pour tous les sondés.

Missing At Random - MAR

Le cas des données MAR est défini lorsque les données ne sont pas totalement manquantes aléatoirement, il arrive que la probabilité d'absence dépende d'une ou plusieurs autres variables X de la base, sans pour autant être liée à la valeur de la variable Y elle-même.

Reprenons l'exemple précédent de l'enquête sur les déterminants du revenu avec notamment la variable âge. L'hypothèse MAR est satisfaite si la probabilité qu'une donnée soit manquante dépend de l'âge de l'individu, et si la

probabilité d'absence de donnée sur le revenu à l'intérieur des groupes d'âge est indépendante du revenu lui-même.

Non Missing At Random - NMAR

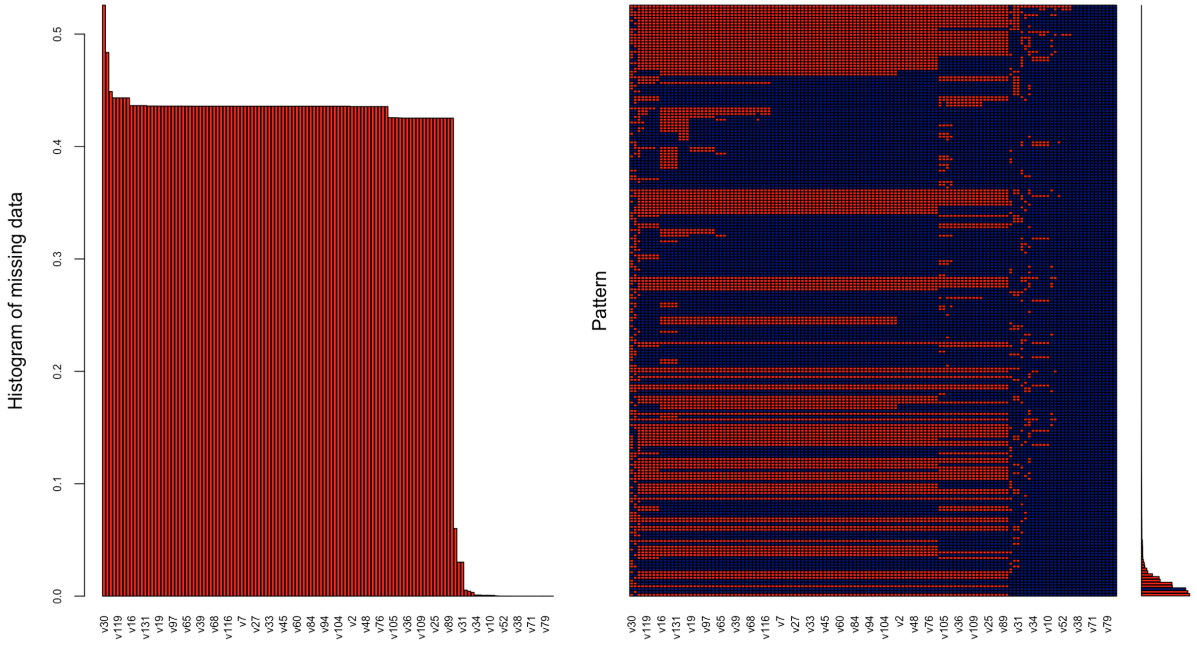
Dans le cas NMAR, les données sont manquantes lorsque la probabilité d'absence dépend de variables inobservées ou de la variable Y elle-même. Les données NMAR induisent une perte de précision sur la variable d'intérêt et de représentation sur l'ensemble de la population.

Dans le cas de l'enquête sur le revenu, les données sur le revenu sont de type NMAR dans le cas où des personnes avec un revenu important refusent de le dévoiler. Les valeurs manquantes dépendent donc de la vraie valeur, inobservée, de la variable Y.

2.1.2 Une représentation graphique des valeurs manquantes

Définition d'un pattern Un patron (ou *pattern*) de valeurs manquantes correspond à une combinaison de valeurs manquantes ou renseignées. Sur le Graphique 2.1 de droite, les lignes correspondent à ces combinaisons de valeurs manquantes (en rouge) ou renseignées (en bleu). Le diagramme à l'extrême droite fournit le nombre d'individus possédant chaque combinaison.

Le diagramme en bâtons à gauche sur le Graphique 2.1 montre la proportion de valeurs manquantes pour chaque variable. La grande majorité des variables a plus de 40 % de valeurs manquantes. Ainsi, une imputation de celles-ci s'avère nécessaire pour ne pas perdre environ la moitié de la base d'apprentissage au cours de l'inférence statistique. Ce diagramme ainsi que celui représentant les patrons (ou *patterns*) de la base de données (à droite sur le Graphique 2.1) indiquent que de nombreuses variables ont des valeurs manquantes simultanément. Ainsi, pour ces variables, il y a une dépendance des valeurs manquantes vis-à-vis des autres variables. Les variables concernées ne seraient donc pas MCAR mais au moins MAR.



Graphique 2.1 – Proportions de valeurs manquantes et patrons (ou *patterns*) sur la base de données d'apprentissage.

Cette conclusion graphique est cependant à nuancer : toutes les variables ne sont pas forcément MAR en particulier celles ayant peu de valeurs manquantes. C'est pourquoi, des tests vont être effectués dans la sous-section suivante

sur deux variables, à titre d'exemple.

2.1.3 Des tests du caractère complètement aléatoire

Les deux variables analysées sont v1 qui a 43.6 % de valeurs manquantes ainsi que v34 avec 0.1% de valeurs manquantes qui est plus susceptible d'être manquante complètement aléatoirement.

Trois méthodes ont été successivement essayées pour ces deux variables : la régression logistique sur le fait d'être manquant, le test de Little et les tests du package R *MissMech*. Ce processus de test pourrait être généralisé à l'ensemble des variables.

Régression logistique sur le fait d'être manquant

Une première façon de voir si une variable est manquante complètement aléatoirement est de créer une indicatrice transcrivant le fait que la valeur est manquante ou non. Une régression logistique peut être menée sur cette indicatrice en prenant pour cofacteurs les autres variables. Si les coefficients associés à ces variables sont significativement non nuls, alors ces variables ont un impact sur le caractère manquant de la variable étudiée. Celle-ci est donc au minimum MAR (et éventuellement NMAR).

Cette méthode a été mise en œuvre en prenant pour cofacteurs les 12 variables entièrement renseignées ce qui introduit 44 paramètres dans la régression logistique.

Lorsque la régression logistique est implémentée sur l'indicatrice issue de v1, 14 paramètres sont significativement non nuls à 5 %. Cela met en évidence que le fait que v1 soit manquante dépend des autres variables et donc que v1 est au moins MAR.

En revanche, pour v34, seul un paramètre est significativement non nul à 5 %. Or, une régression logistique sur une variable générée aléatoirement suivant une loi de Bernoulli montre qu'il se peut, de façon fortuite, qu'un ou deux coefficients soient significativement non nuls. Ainsi, il paraît difficile de conclure du caractère non complètement aléatoire de v34 au vu des résultats de la régression logistique. Des tests supplémentaires sont nécessaires.

Test de Little

Little ([1]) a proposé un test multivarié du caractère complètement aléatoire des données manquantes. Il suppose que les variables sont distribuées selon une loi normale multivariée. Ce test n'est donc pas approprié aux variables qualitatives. Même si les variables ne sont pas distribuées suivant une loi normale (ce qui est le cas ici), le test reste valide asymptotiquement. L'idée de Little est de tester l'égalité des valeurs moyennes de chaque *pattern* de valeurs manquantes.

Le test de Little a été mené, successivement sur v1 et v34, avec les 4 seules variables quantitatives complètes (v38, v62, v72, v129). La p-valeur étant très proche de 0 pour la variable v1, l'hypothèse nulle que la variable est MCAR est rejetée. En revanche, l'hypothèse que v34 est MCAR n'est pas rejetée à 5 % car la p-valeur est de 20 %.

Les tests du package *MissMech*

Jamshidian et Jalal ([2]) ont introduit deux autres tests du caractère MCAR des données manquantes. L'un suppose la normalité des données (test de Hawkins) alors que l'autre est non-paramétrique et ne suppose donc pas

de distribution. L'idée commune est de tester l'égalité des matrices de variance-covariance de chaque *pattern* de valeurs manquantes.

Les tests ont été menés, pour des raisons computationnelles, sur un sous-échantillon aléatoire de 1000 observations de la variable v1 avec les 4 variables quantitatives complètes (v38, v62, v72, v129). Même si l'hypothèse de normalité n'est pas vérifiée comme évoqué précédemment, le test de Hawkins invite à rejeter l'hypothèse nulle de données MCAR à 5 %. La conclusion est la même avec le test non-paramétrique. En revanche, le test ne fonctionne pas pour la variable v34 en raison d'un nombre insuffisant de *patterns*.

2.1.4 Et si les variables étaient NMAR ?

Le fait que les données manquantes soient NMAR est très difficilement vérifiable en pratique. Pour la variable v1, cela pourrait être possible par exemple si v1 était très corrélée avec une variable qui est renseignée lorsque v1 ne l'est pas. Cette autre variable indiquerait alors le caractère particulièrement élevé ou faible des valeurs manquantes de v1 si elles étaient renseignées. Cela attesterait donc du caractère NMAR de v1. Or, parmi le peu de variables quantitatives bien renseignées, aucune n'a de corrélation « forte » avec v1. Le fait que les valeurs manquantes de v1 soient NMAR ou pas n'est donc pas vérifiable par la méthode précédente.

L'important dans cette partie était essentiellement de voir si les valeurs manquantes l'étaient complètement aléatoirement auquel cas la suppression des observations incomplètes aurait été valide [1]. Or, pour l'immense majorité des variables, les valeurs manquantes ne le sont pas complètement aléatoirement mais dépendent des autres variables explicatives. Cela justifie l'utilisation, dans la prochaine partie, d'une méthode d'imputation tenant compte des autres variables.

2.2 Imputation des valeurs manquantes et remplissage de la base

Presque toutes les méthodes usuelles de prédiction s'appuient sur une base de données complète. L'imputation des valeurs manquantes est donc un prérequis incontournable.

2.2.1 Traitement des variables catégorielles

Certaines variables ne peuvent être intégrées tel quel à l'exercice, essentiellement pour un motif de temps de calcul.

Parmi les variables catégorielles, trois présentent plus de 50 modalités différentes, ce qui pose problème pour la construction de forêts aléatoires ou l'imputation gaussienne. En effet, les packages plafonnent le nombre de modalités autorisées, pour des raisons de temps de calcul. Deux de ces variables présentent un nombre de modalités raisonnable (v56 : 122 modalités, V125 : 90 modalités). Pour réduire artificiellement le nombre de modalités, nous choisissons de traiter les individus appartenant à des modalités « orphelines » (moins de 1% de la population) comme des données non observées. Nous avons également envisagé un regroupement par arbre de classification (sur les 12 variables complètes). La troisième variable (v22) présente en revanche plus de 18 000 modalités, soit moins de 8 individus par modalité en moyenne. Dans ces conditions le traitement des modalités « orphelines » (même à 0.1%) conduit à une large majorité de données non-observées, et le regroupement des modalités par un arbre de classification (sur les 12 variables complètes) se heurte à des limites de capacité de calcul. Nous avons donc décidé d'exclure la variable v22 de la suite de l'analyse.

1. Cela n'aurait pas introduit de biais.

2.2.2 Méthodes d'imputation des données manquantes

Imputation gaussienne multivariée

L'imputation gaussienne multivariée est la méthode standard. Elle repose sur deux hypothèses relativement strictes :

— Les données complètes (observées et manquantes) suivent une loi normale multivariée
— Les données sont *Missing At Random* : le pattern des valeurs manquantes ne dépend que des données observées

Le package Amelia (J. Honaker, G.King et M. Blackwell) permet de réaliser une imputation gaussienne. On génère par bootstrap m jeux de données. Pour chacun d'eux, les paramètres de la loi normale multivariée sont estimés par un algorithme espérance-maximisation (dit algorithme EM). La valeur de chaque donnée manquante est ensuite imputée conditionnellement aux données observées de l'individu et à la distribution des paramètres (obtenue grâce au bootstrap). m valeurs possibles sont générées pour chaque valeur manquante, qu'il est simple d'agréger en une valeur unique. La valeur de m n'a pas besoin d'être élevée pour obtenir de bons résultats (les concepteurs recommandent $m = 5$).

Dans cette technique, une variable factorielle à p modalités est traitée comme *p-1* variables binaires. Les valeurs imputées permettent d'associer une probabilité aux modalités pour chaque valeur manquante. Dans notre cas où les variables factorielles génèrent un grand nombre de modalités, le nombre de variables augmente sensiblement et le temps de calcul augmente en conséquence. Nous aurions pu opter pour une réduction plus poussée du nombre de modalités de certaines variables, mais nous avons choisi d'explorer d'autres voies.

Imputation par arbre

Afin de relâcher les hypothèses fortes de l'imputation gaussienne, nous proposons une méthode alternative : l'imputation par arbre. L'idée est de réaliser une imputation des variables incomplètes à partir des 12 variables complètes, à l'aide d'un modèle d'arbres.

Pour chaque variable V, un modèle est calibré sur l'ensemble des individus pour lesquels cette variable est renseignée :
— Pour une variable continue, un arbre de régression
— Pour une variable catégorielle, un arbre de classification
Pour l'imputation de la base test, deux possibilités sont envisageables : réutiliser les arbres calibrés sur la base train ou recalibrer les arbres sur la base test. Les bases train et test étant deux échantillons aléatoires de la même base, ces deux options sont *a priori* équivalentes.

L'imputation des variables catégorielles peut être un sujet de difficultés en cas de modalités à faible effectif. Nous envisageons dans le tableau 2.1 les différents cas de figure, selon que la modalité apparaît seulement dans la base d'apprentissage ou seulement dans la base de test.

	Modalité dans la base d'apprentissage (*train*) uniquement	Modalité dans la base test uniquement
Variable complète	Aucun retraitement nécessaire	Retraitement de la nouvelle modalité (à très faible effectif, étant donné l'échantillonnage aléatoire de train et test) : traitement en valeur manquante et imputation à partir des autres variables complètes ou recalibrage des arbres sur la base test
Variable incomplète		Retraitement en valeur manquante avant imputation

Tableau 2.1 – Traitement des modalités à effectif faible

Précisons qu'au sein d'une même base (train ou test), pour une variable incomplète V, il existe deux sous-populations,

définies comme suit :

— V_Renseignée : individus pour lesquels la variable V est renseignée

— V_NA : individus pour lesquels la variable V est manquante

Ces deux sous-échantillons peuvent présenter des catégories différentes au sein des variables factorielles complètes. Si les données sont MCAR, les sous-populations sont obtenues par échantillonnage aléatoire ce qui rend la situation peu probable, mais le cas peut se présenter pour une catégorie à faible effectif. Si les données sont MAR, la situation est plus probable dans la mesure où le pattern des valeurs manquantes dépend des valeurs observées : si la catégorie X d'une variable complète a un impact positif sur la probabilité que la variable V soit manquante, X sera surreprésentée voire uniquement représentée dans la sous-population V_NA. Le tableau 2.2 présente les deux cas possibles.

	Modalité dans V_renseignée uniquement	Modalité dans V_NA uniquement
Variable factorielle complète	Aucun retraitement nécessaire	Retraitement arbitraire ou pré-imputation à partir des autres variables complètes.

Tableau 2.2 – Traitement des modalités à effectif faible des variables factorielles complètes

Afin d'évaluer la force prédictive des arbres construits pour chaque variable incomplète V, nous procédons comme suit :

— La sous-population des individus dont la variable V est renseignée (V_renseignée) est aléatoirement répartie en une base « train » (V_rens_train) comportant 75% des individus et une base « test » (V_rens_test) comportant les 25% restants.

— Un arbre est construit sur la base V_rens_train, puis utilisé pour prédire la variable V sur la base V_rens_test.

Pour un arbre de classification, la performance est mesurée comme le taux de classification exacte. Pour un arbre de régression, nous mesurons la performance comme le coefficient de détermination de la régression des valeurs prédites sur les valeurs réelles.

Imputation par forêt aléatoire

Une autre méthode d'imputation moins standard est disponible dans R, à l'aide d'un package dédié.

L'imputation par forêt aléatoire est proposée en particulier dans le package MissForest. Son premier avantage est de traiter simultanément les variables continues et catégorielles (ce que ne font pas les autres méthodes) en tenant compte de leurs interactions potentielles.

L'imputation elle-même est basée sur une forêt aléatoire. Un grand nombre d'arbres sont générés par boostrap. Les arbres construits sont relativement simples : un petit nombre de variables tirées au sort (le nombre de variables utilisées dans chaque arbre est fixé inférieur au nombre total de variables disponibles), une faible profondeur. L'agrégation fournit un résultat stable et robuste.

En utilisant l'erreur *out-of-bag* de la forêt, l'erreur d'imputation est estimée sans recourir à un échantillon de test. Ce dernier point simplifie les procédures. D'après les auteurs du package [5], leur étude comparative conclut aux meilleures performances de l'imputation par forêt aléatoire par rapport aux méthodes standards, en particulier si les relations entre les variables sont complexes ou non-linéaires.

Etant donnée l'ampleur des calculs menés, les temps de calcul sont raisonnables. Cependant, la taille de notre base de données ne permet pas de mettre en œuvre cette méthode.

2.2.3 Comparaison des performances entre l'imputation gaussienne et l'imputation par arbre de régression

Nous souhaitons comparer les résultats de l'imputation par arbres de régression à ceux de l'imputation gaussienne. Pour ce faire, nous considérons une base de 20% des individus (obtenue par échantillonnage aléatoire) comportant

uniquement les variables continues. Ces choix sont motivés par des contraintes de temps de calcul. Pour chaque variable V prise isolément, nous échantillonnons à nouveau la base :

— L'échantillon 1 (67% des individus) n'est pas modifié
— Les valeurs observées de V au sein de l'échantillon 2 (33% des individus) sont transformées en données non-observées.

On impute ensuite la base avec le package FastImputation, qui approxime l'algorithme utilisé par Amelia pour le rendre plus rapide. Les données imputées de la variable V sur les « fausses » données manquantes sont alors régressées sur les valeurs effectivement observées, et la performance de l'imputation mesurée à l'aune du coefficient de détermination, comme précédemment pour les arbres de régression.

La performance comparée des deux méthodes d'imputation (gaussienne ou par arbre) apparaît hautement dépendante de la variable concernée. Dans le tableau 5 (voir en Annexe), la colonne Rapport donne le rapport du coefficient de détermination associé à l'imputation gaussienne sur celui associé à l'imputation par arbre. On observe ainsi que pour 74 variables, l'imputation gaussienne est plus performante (avec un R^2 jusqu'à 100 fois plus élevé), tandis que pour 34 autres, c'est l'imputation par arbre qui fournit le meilleur résultat (avec un R^2 jusqu'à 5 fois plus élevé). Ces résultats conduisent globalement à privilégier l'imputation gaussienne, dont la performance pourrait être encore améliorée par l'inclusion des variables factorielles (sous réserve de capacités de calcul suffisantes).

Chapitre 3

Prédiction de la variable target

Nous avons désormais deux bases à disposition. L'une, la base initiale, est entachée de nombreuses valeurs manquantes, mais quelques méthodes robustes aux NA prédisent directement la variable cible. L'autre, complétée par imputation gaussienne (la plus efficace), autorise le recourt à toutes les méthodes usuelles. Il s'agit maintenant de comparer les performances prédictives des différentes options.

3.1 Méthodes robustes aux NA : comparaison des résultats sur base initiale et base imputée

Les méthodes robustes aux valeurs manquantes ont été présentées en première partie. L'imputation préalable de la base améliore-t-elle leur performance ?

3.1.1 Méthode CART

La méthode CART a été utilisée à la fois sur la base complétée et la base non-complétée. Au préalable, l'on a réalisé une cross-validation afin de déterminer le *cp* optimal (paramètre de complexité permettant de *pruner* (découper) les ramifications où il y a risque de surapprentissage).

Tableau 3.1 – Les indicateurs de performance de CART implémenté sur la base non complétée et sur la base complétée

	Indice de Gini	**AUC**	**AUCH**
Base non complétée	0.184	0.592	0.600
Base complétée	0.333	0.666	0.666

Ce tableau nous permet de constater que indices de performances calculés sur la base complétée sont sensiblement supérieurs à ceux calculés sur la base non-complétée. Dans le cas de l'indice de Gini, on observe une augmentation de plus de 80% sur la base imputée par rapport à la base brute. Pour l'AUC et l'AUCH, l'augmentation est un peu moins nette et atteint près de 13% sur la base imputée par rapport à la base brute. On en déduit logiquement que le travail préalable de traitement des données manquantes améliore la qualité de la prédiction, sous contrainte bien entendu que l'imputation des données manquantes se fassent de façon réfléchie.

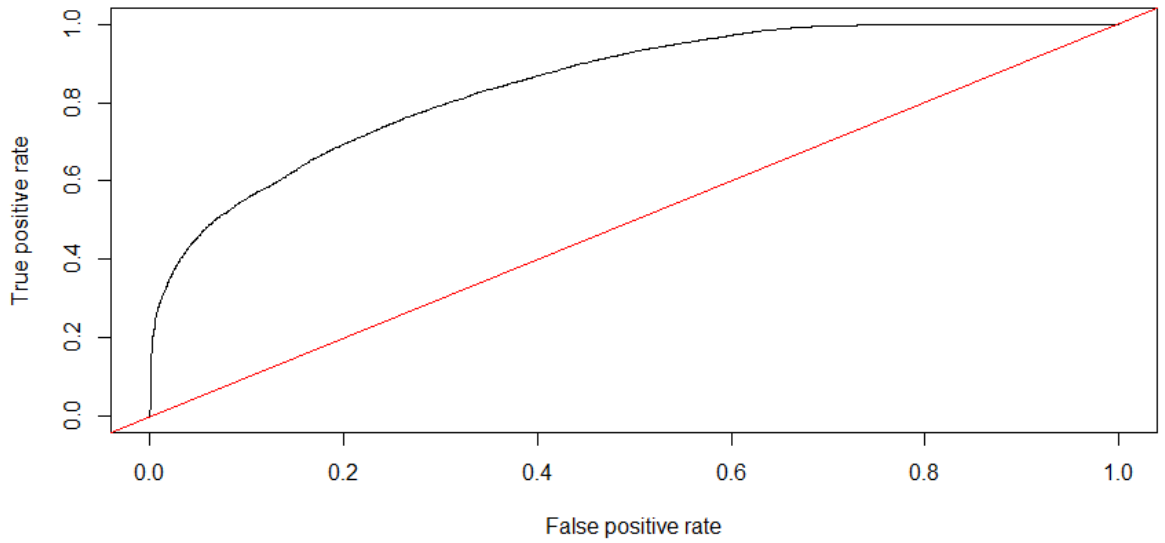
3.1.2 XGboost

L'algorithme de boosting est appliqué successivement aux deux bases de données. Le tableau 3.2 compare ses performances. Les trois indicateurs retenus sont nettement supérieurs pour la base préalablement complétée : l'AUC et l'AUCH augmentent de 15%, l'indice de Gini de 47%. L'imputation des valeurs manquantes améliore donc les performances de cet algorithme.

	Indice de Gini	**AUC**	**AUCH**
Base non complétée	0.471	0.735	0.736
Base complétée	0.694	0.847	0.848

Tableau 3.2 – Les indicateurs de performance des XGBoost implémentés sur la base non complétée et sur la base complétée.
Bases de données d'apprentissage non complétées et complétées par EM.

Le graphique 3.1 ci-dessous représente la courbe ROC issue de l'implémentation de l'algorithme XGboost sur la base imputée. Plus l'aire entre la courbe et la première bissectrice, donnée par l'indice AUC (*Area Under Curve*), est élevée, meilleure est la qualité de la prédiction. Dans le présent cas, XGboost aboutit aux meilleurs résultats.



Graphique 3.1 – Courbe ROC obtenue par XGBoost en validation croisée sur la base imputée.

3.2 Méthodes standards sur base complète : résultats obtenus

3.2.1 Modèle logistique

Avec une base complète et une variable cible binaire, une régression logistique s'impose comme méthode de prédiction usuelle.

La taille de la régression initiale (129 variables pour près de 115 000 individus) provoque des temps de calculs très longs. Pour contourner provisoirement ce problème, 20% des lignes de la base sont tirées aléatoirement. Le modèle avec toutes les variables explicatives est ajusté sur elle. Les variables non significatives à 10 % sont écartées

de l'analyse. La régression logistique avec 37 variables utilise un temps de calcul raisonnable pour la base complète. Le modèle est alors ajusté en supprimant itérativement les variables non significatives à 5%. Le modèle final retient 34 variables explicatives. Les coefficients sont présentés en annexe, tableau 6.

Pour estimer la performance du modèle, la régression logistique est réestimée sur 2/3 de la base. Les coefficients estimés prédisent la variable target sur le tiers restant et peuvent être comparé aux valeurs réelles de la variable target. Les indicateurs de performance sont renseignés dans le tableau 3.3, ils sont largement inférieurs à ceux du modèle XGBoost.

	Indice de Gini	**AUC**	**AUCH**
Base complétée	0.422	0.711	0.713

Tableau 3.3 – Les indicateurs de performance de la régression logistique

3.2.2 Régression LASSO

Une régression LASSO (*Least Absolute Shrinkage and Selection Operator*) a été effectuée sur les deux premiers tiers de la base *train*. Il s'agit d'une régression GLM classique, mais qui pénalise la taille des coefficients. La norme des coefficients est majorée par un coefficient λ, qu'il s'agit ensuite d'optimiser pour obtenir des estimateurs des coefficients de régression non biaisés.

Malheureusement, le package *gmlnet* utilisé pour implémenter ce modèle en R ne permet pas de l'appliquer à des données manquantes, contrairement à ce qui peut être effectué dans la théorie.

Les performances de cette méthode sont illustrées par les indicateurs du tableau 3.4. Celles-ci sont nettement inférieures, sur la base complétée, à celles du modèle XGBoost. Elles sont cependant légèrement supérieures à celles de la régression logistique - ce qui est cohérent puisque le LASSO est une extension des GLM.

	Indice de Gini	**AUC**	**AUCH**
Base complétée	0.460	0.730	0.731

Tableau 3.4 – Les indicateurs de performance de la régression Lasso implémentée sur la base non complétée et sur la base complétée.

Conclusion

Notre travail a permis de mettre en évidence plusieurs points fondamentaux. D'abord, l'imputation des données manquantes s'est révélée determinante pour obtenir une meilleure qualité de prédiction. La méthode d'imputation gaussienne se démarque sensiblement de ses concurrentes. Enfin, parmi toutes les méthodes de prédiction testées, l'algorithme XGboost a abouti aux meilleurs résultats de prédiction. Néanmoins, son opacité est un frein important à son utilisation. De par son caractère "boîte noire", l'interprétabilité de la méthode devient difficile.
Un des principaux points d'amélioration que l'on aurait pu et même dû inclure réside dans l'insertion de nouvelles variables explicatives. De fait, le sujet de l'étude soulignait la nécessité de prédire la catégorie du sinistre, en se demandant si des renseignements complémentaires sont nécessaires : jusqu'où faut il poser des questions pour approuver le paiement du sinistre ? Comment optimiser le nombre minimal de questions à poser ? Dans un tel contexte, les données manquantes devenaient elles-mêmes une information cruciale du sujet. Il aurait donc été pertinent d'étudier les patterns et d'inclure des variables traduisant le fait que toutes les observations avec beaucoup de valeurs manquantes ne sont pas à traiter de la même manière que celles qui n'en ont pas.
La présente étude a enfin été un bon prétexte pour manipuler un nombre important de données d'assurance. Le traitement de la base et des valeurs manquantes, l'application de méthodes robustes de prédiction ont été autant de difficultés rencontrés mais pourtant quotidiennes dans le métier d'un actuaire. Malgré tout, l'important reste de bien prendre du recul par rapport aux méthodes utilisées et aux données à notre disposition.

Bibliographie

[1] LITTLE R.J.A., « Little's test of missing completely at random », The Journal of The American Association, Décembre 1988, p. 1198.

[2] JAMSHIDIAN M., JALAL S., JENSEN C. « MissMech : An R Package for Testing Homoscedasticity, Multivariate Normality, and Missing Completely at Random (MCAR) », Journal of Statistical Software, Janvier 2014, Volume 56, Issue 6.

[3] Marina Soley-Bori, "Dealing with missing data : Key assumptions and methods for applied analysis", Boston University School of Public Health, Mai 2013.

[4] Richard Williams, University of Notre-Dame, "Missing Data Part 1 : Overview, Traditional Methods", Janvier 2015.

[5] STEKHOVEN Daniel J. BÜHLMAN Peter, "MissForest - nonparametric missing value imputation for mixed-type data", 2011

.1 Annexe : Performances comparées de deux méthodes d'imputation

	Variable	Type	Nombre_NA	Imputation_arbres	Imputation_gaussienne	Rapport
90	v69	numeric	49895	0.01	0.85	99.53
104	v86	numeric	49832	0.00	0.44	96.18
38	v16	numeric	49895	0.01	0.79	84.44
18	v115	numeric	49895	0.01	0.83	82.26
1	v1	numeric	49832	0.01	0.60	74.85
64	v42	numeric	49832	0.01	0.50	65.40
96	v78	numeric	49895	0.01	0.64	59.05
13	v11	numeric	49836	0.01	0.62	53.84
35	v131	numeric	49895	0.01	0.43	47.27
93	v73	numeric	49836	0.01	0.61	46.55
63	v41	numeric	49832	0.01	0.49	41.20
88	v67	numeric	49832	0.01	0.46	40.88
65	v43	numeric	49836	0.01	0.51	40.10
54	v32	numeric	49832	0.01	0.44	39.15
102	v84	numeric	49832	0.02	0.62	37.33
37	v15	numeric	49836	0.01	0.32	37.24
26	v122	numeric	49851	0.01	0.21	36.12
8	v105	numeric	48658	0.02	0.81	35.94
59	v37	numeric	49843	0.02	0.64	35.58
83	v60	numeric	49832	0.02	0.54	32.74
46	v25	numeric	48619	0.02	0.73	30.39
116	v97	numeric	49843	0.03	0.60	23.45
76	v54	numeric	48619	0.02	0.45	18.45
107	v89	numeric	48619	0.02	0.42	17.54
21	v118	numeric	49843	0.03	0.51	16.50
31	v127	numeric	49832	0.01	0.12	15.89
108	v9	numeric	49851	0.01	0.15	15.00
98	v80	numeric	49851	0.02	0.27	14.63
11	v108	numeric	48624	0.02	0.21	14.07
85	v63	numeric	48619	0.02	0.30	14.06
112	v93	numeric	49832	0.04	0.46	12.52
97	v8	numeric	48619	0.02	0.22	11.51
47	v26	numeric	49832	0.01	0.11	10.63
34	v130	numeric	49843	0.03	0.33	9.37
32	v128	numeric	48624	0.02	0.20	9.24
113	v94	numeric	49832	0.01	0.06	8.58
68	v46	numeric	48619	0.02	0.21	8.49
71	v5	numeric	48624	0.01	0.10	7.08
60	v39	numeric	49836	0.01	0.09	6.95
89	v68	numeric	49836	0.01	0.07	6.84
73	v51	numeric	50678	0.03	0.17	6.60

Tableau 5 – Performances comparées de deux méthodes d'imputation

	Variable	Type	Nombre_NA	Imputation_arbres	Imputation_gaussienne	Rapport
25	v121	numeric	49840	0.03	0.17	6.38
50	v29	numeric	49832	0.02	0.10	6.14
99	v81	numeric	48624	0.01	0.08	6.05
77	v55	numeric	49832	0.03	0.15	5.31
109	v90	numeric	49836	0.01	0.04	5.05
40	v18	numeric	49832	0.01	0.05	4.89
6	v103	numeric	49832	0.01	0.04	4.63
33	v13	numeric	49832	0.02	0.09	4.58
92	v70	numeric	48636	0.01	0.05	3.97
61	v4	numeric	49796	0.04	0.16	3.94
5	v102	numeric	51316	0.03	0.12	3.57
91	v7	numeric	49832	0.02	0.06	3.53
118	v99	numeric	49832	0.01	0.05	3.52
24	v120	numeric	49836	0.00	0.01	3.22
75	v53	numeric	49836	0.01	0.02	2.99
17	v114	numeric	30	0.32	0.91	2.84
7	v104	numeric	49832	0.01	0.03	2.56
42	v2	numeric	49796	0.03	0.07	2.44
62	v40	numeric	111	0.38	0.90	2.39
87	v65	numeric	49840	0.03	0.06	2.31
56	v34	numeric	111	0.38	0.86	2.26
58	v36	numeric	48624	0.02	0.04	2.22
81	v59	numeric	49796	0.01	0.02	1.96
67	v45	numeric	49832	0.01	0.02	1.92
106	v88	numeric	49832	0.04	0.06	1.71
28	v124	numeric	48619	0.02	0.03	1.46
36	v14	numeric	4	0.47	0.67	1.40
27	v123	numeric	50678	0.04	0.05	1.25
20	v117	numeric	48624	0.03	0.04	1.23
2	v10	numeric	84	0.59	0.70	1.19
115	v96	numeric	49832	0.01	0.01	1.17
43	v20	numeric	49840	0.02	0.02	1.11
49	v28	numeric	49832	0.01	0.01	1.10
44	v21	numeric	611	0.27	0.27	1.00
23	v12	numeric	86	0.48	0.46	0.96
41	v19	numeric	49843	0.03	0.03	0.95
45	v23	numeric	50675	0.01	0.01	0.84
14	v111	numeric	49832	0.03	0.02	0.80
19	v116	numeric	49836	0.01	0.01	0.73
82	v6	numeric	49832	0.01	0.01	0.68
4	v101	numeric	49796	0.03	0.02	0.68
79	v57	numeric	49832	0.02	0.01	0.60
3	v100	numeric	49836	0.05	0.03	0.59
114	v95	numeric	49843	0.02	0.01	0.55
101	v83	numeric	49832	0.03	0.02	0.52
95	v77	numeric	49832	0.01	0.00	0.46
69	v48	numeric	49796	0.04	0.02	0.43
111	v92	numeric	49843	0.01	0.00	0.41
80	v58	numeric	49836	0.05	0.02	0.39
57	v35	numeric	49832	0.01	0.00	0.38
66	v44	numeric	49796	0.03	0.01	0.30
55	v33	numeric	49832	0.03	0.01	0.25
86	v64	numeric	49796	0.03	0.01	0.23
48	v27	numeric	49832	0.01	0.00	0.20
39	v17	numeric	49796	0.03	0.00	0.16

	Variable	Type	Nombre_NA	Imputation_arbres	Imputation_gaussienne	Rapport
72	v50	numeric	86	0.36	0.04	0.11
103	v85	numeric	50682	0.03	0.00	0.10
84	v61	numeric	49796	0.04	0.00	0.08
100	v82	numeric	48624	0.02	0.00	0.08
94	v76	numeric	49796	0.03	0.00	0.08
70	v49	numeric	49832	0.01	0.00	0.07
117	v98	numeric	48654	0.02	0.00	0.04
30	v126	numeric	49832	0.50	0.01	0.03
9	v106	numeric	49796	0.04	0.00	0.02
12	v109	numeric	48624	0.02	0.00	0.01
22	v119	numeric	50680	0.04	0.00	0.00
105	v87	numeric	48663	0.02	0.00	0.00

.2 Annexe : Coefficients de la régression logistique

Tableau 6 – Coefficients de la régression logistique

	Estimate	Std. Error	z value	Pr(>|z|)
(Intercept)	1.4775	9.1048	0.16	0.8711
v6	0.1407	0.0202	6.96	0.0000
v8	0.0611	0.0171	3.58	0.0003
v11	-1.3565	0.2925	-4.64	0.0000
v13	0.3508	0.0364	9.65	0.0000
v18	0.1229	0.0409	3.00	0.0027
v27	0.1410	0.0271	5.21	0.0000
v28	-0.0249	0.0079	-3.14	0.0017
v32	-0.8167	0.0788	-10.37	0.0000
v35	0.1356	0.0191	7.11	0.0000
v39	0.8776	0.3245	2.70	0.0068
v43	-1.3165	0.2916	-4.51	0.0000
v57	0.2036	0.0285	7.14	0.0000
v59	0.1033	0.0125	8.29	0.0000
v61	0.1002	0.0277	3.61	0.0003
v63	-0.0505	0.0160	-3.15	0.0016
v64	0.0937	0.0327	2.87	0.0041
v68	0.9572	0.3509	2.73	0.0064
v70	0.0338	0.0096	3.52	0.0004
v73	-0.6826	0.2851	-2.39	0.0166
v98	0.0672	0.0097	6.94	0.0000
v102	0.0023	0.0009	2.68	0.0073
v104	-0.2554	0.0398	-6.41	0.0000
v106	-0.1434	0.0410	-3.50	0.0005
v120	0.9970	0.3511	2.84	0.0045
v127	0.0434	0.0128	3.38	0.0007

	Estimate	Std. Error	z value	Pr(>|z|)
v24B	0.1016	0.0475	2.14	0.0324
v24C	0.3093	0.0434	7.12	0.0000
v24D	-0.0873	0.0417	-2.10	0.0362
v24E	-0.3474	0.0431	-8.05	0.0000
v30B	0.3899	0.2178	1.79	0.0733
v30C	-0.0259	0.0476	-0.54	0.5862
v30D	-0.2034	0.0557	-3.65	0.0003
v30E	-0.1379	0.0555	-2.49	0.0129
v30F	0.1099	0.0657	1.67	0.0946
v30G	-0.1254	0.0512	-2.45	0.0143
v31B	1.0536	0.0642	16.41	0.0000
v31C	0.5229	0.1108	4.72	0.0000
v56AG	-0.4391	0.1016	-4.32	0.0000
v56AS	-0.1816	0.0867	-2.09	0.0362
v56AW	-0.7470	0.0958	-7.80	0.0000
v56BJ	0.0159	0.0942	0.17	0.8661
v56BL	-0.1729	0.0779	-2.22	0.0264
v56BV	-1.2477	0.1006	-12.40	0.0000
v56BW	-1.8303	0.0885	-20.68	0.0000
v56BX	-2.0580	0.1209	-17.02	0.0000
v56BZ	-0.4904	0.0898	-5.46	0.0000
v56CN	-1.0259	0.0967	-10.60	0.0000
v56CY	-0.6676	0.0898	-7.43	0.0000
v56DF	-1.3886	0.1124	-12.35	0.0000
v56DH	-0.2808	0.1141	-2.46	0.0138
v56DI	-0.9482	0.0897	-10.57	0.0000
v56DJ	-1.2802	0.1127	-11.36	0.0000
v56DO	-2.0871	0.0941	-22.17	0.0000
v56DP	-1.0998	0.0937	-11.74	0.0000
v56DS	0.1272	0.0860	1.48	0.1391
v56DX	-1.3267	0.1007	-13.17	0.0000
v56DY	-1.1192	0.1032	-10.84	0.0000
v56N	-0.4029	0.0982	-4.10	0.0000
v56P	-2.0739	0.0930	-22.30	0.0000
v56U	0.2266	0.1011	2.24	0.0250
v56V	-1.2068	0.1266	-9.54	0.0000
v66B	-0.4763	0.0264	-18.02	0.0000
v66C	0.8890	0.0314	28.34	0.0000
v74B	1.0298	0.4272	2.41	0.0159
v74C	1.9268	0.4498	4.28	0.0000
v79B	1.9508	0.1671	11.67	0.0000
v79C	1.8021	0.1806	9.98	0.0000
v79D	1.0812	0.1854	5.83	0.0000
v79E	0.8795	0.1747	5.04	0.0000
v79F	2.2265	0.2328	9.56	0.0000
v79G	-0.3135	0.8412	-0.37	0.7094
v79H	1.7261	0.1936	8.92	0.0000
v79I	1.3098	0.1870	7.01	0.0000
v79J	0.8627	0.2210	3.90	0.0001
v79K	-0.0275	0.1784	-0.15	0.8775
v79L	7.0523	43.9545	0.16	0.8725
v79M	1.2086	0.1778	6.80	0.0000
v79N	1.2624	0.5522	2.29	0.0223
v79O	0.2001	0.1715	1.17	0.2433

	Estimate	Std. Error	z value	Pr(>|z|)
v79P	0.5935	0.2066	2.87	0.0041
v79Q	-0.4637	0.1814	-2.56	0.0106
v79R	0.9376	0.4252	2.20	0.0275
v113AA	0.7532	0.4040	1.86	0.0623
v113AB	0.1339	0.1077	1.24	0.2138
v113AC	-0.0363	0.0993	-0.37	0.7149
v113AD	0.3766	0.1166	3.23	0.0012
v113AE	-0.2235	0.1280	-1.75	0.0809
v113AF	-0.0457	0.1006	-0.45	0.6498
v113AG	-0.1299	0.1059	-1.23	0.2200
v113AH	0.1883	0.1159	1.63	0.1041
v113AI	-0.0469	0.1527	-0.31	0.7587
v113AJ	-0.0903	0.1186	-0.76	0.4466
v113AK	1.7521	1.0475	1.67	0.0944
v113B	0.1240	0.1104	1.12	0.2617
v113C	0.0664	0.1014	0.66	0.5121
v113D	-0.2616	0.2152	-1.22	0.2242
v113E	0.0726	0.1139	0.64	0.5240
v113F	0.3095	0.1025	3.02	0.0025
v113G	0.0808	0.0974	0.83	0.4070
v113H	0.0872	0.1119	0.78	0.4357
v113I	0.1531	0.1020	1.50	0.1334
v113J	0.3416	0.1220	2.80	0.0051
v113L	0.1531	0.1304	1.17	0.2405
v113M	-0.3234	0.1053	-3.07	0.0021
v113N	-0.0441	0.1460	-0.30	0.7628
v113O	0.2525	0.1299	1.94	0.0519
v113P	-0.0509	0.1075	-0.47	0.6357
v113Q	0.0075	0.1082	0.07	0.9447
v113R	0.4227	0.1237	3.42	0.0006
v113S	-0.4188	0.1202	-3.49	0.0005
v113T	-0.1104	0.1149	-0.96	0.3367
v113U	-0.1430	0.1015	-1.41	0.1590
v113V	-0.0377	0.1093	-0.35	0.7300
v113W	-0.0724	0.0981	-0.74	0.4600
v113X	0.0158	0.1098	0.14	0.8853
v113Y	-0.3009	0.1093	-2.75	0.0059
v113Z	0.2510	0.1333	1.88	0.0596
v125AC	0.0748	0.0640	1.17	0.2425
v125AK	0.1727	0.0664	2.60	0.0093
v125AN	0.1737	0.0651	2.67	0.0077
v125AP	-0.0025	0.0702	-0.04	0.9711
v125AR	0.0127	0.0777	0.16	0.8705
v125AZ	0.1495	0.0796	1.88	0.0604
v125B	0.0644	0.0760	0.85	0.3968
v125BD	0.0751	0.0761	0.99	0.3239
v125BH	0.1550	0.0680	2.28	0.0226
v125BJ	0.1931	0.0675	2.86	0.0042
v125BK	0.0470	0.0661	0.71	0.4773
v125BL	0.0339	0.0719	0.47	0.6371
v125BM	0.1237	0.0651	1.90	0.0574
v125BU	0.3633	0.0778	4.67	0.0000
v125BW	0.0353	0.0699	0.51	0.6131
v125BX	0.1529	0.0885	1.73	0.0840

	Estimate	Std. Error	z value	Pr(>|z|)
v125BY	-0.0032	0.0704	-0.05	0.9641
v125CA	0.1240	0.0767	1.62	0.1062
v125CD	0.1245	0.0703	1.77	0.0765
v125CE	0.1174	0.0657	1.79	0.0740
v125CG	0.1574	0.0628	2.51	0.0121
v125CJ	0.0408	0.0825	0.49	0.6212
v125E	0.0578	0.0742	0.78	0.4355
v125G	0.0274	0.0752	0.36	0.7155
v125H	0.0993	0.0706	1.41	0.1594
v125K	0.1505	0.0734	2.05	0.0404
v125L	-0.0022	0.0747	-0.03	0.9767
v125P	0.0596	0.0810	0.74	0.4619
v125R	0.0589	0.0643	0.92	0.3596
v125V	0.1302	0.0672	1.94	0.0527
v125Z	0.2062	0.0761	2.71	0.0067

.3 Annexe : Coefficients de la régression Lasso

Tableau 7 – Coeficients de la régression Lasso

	1
(Intercept)	-1.491
v1	-0.063
v2	0.018
v4	0.081
v5	0.037
v6	0.179
v7	0
v8	0.075
v9	0.091
v10	0
v11	-0.188
v12	-0.037
v13	0.070
v14	0.036
v15	-0.242
v16	-0.120
v17	-0.017
v18	0.146
v19	0.101
v20	-0.076
v21	0
v23	0.009
v25	0
v26	-0.051
v27	0
v28	-0.032
v29	0
v32	0
v33	0
v34	0
v35	0
v36	-0.016
v37	-0.045
v38	0.051
v39	-0.030
v40	-0.035
v41	0
v42	0
v43	0
v44	0.018
v45	-0.024
v46	-0.095
v48	0.090
v49	0
v50	0.682

v51	0.010
v53	0.160
v54	0
v55	-0.136
v57	-0.003
v58	0.004
v59	0.056
v60	0
v61	0.017
v62	0
v63	0
v64	0
v65	-0.017
v67	0
v68	0
v69	0
v70	0.007
v72	0.072
v73	0.0003
v76	-0.044
v77	-0.003
v78	-0.113
v80	-0.007
v81	0.00002
v82	-0.002
v83	0
v84	0
v85	-0.025
v86	0
v87	-0.011
v88	0.030
v89	0.037
v90	0.360
v92	0.066
v93	0.140
v94	0.120
v95	0.019
v96	0
v97	0
v98	0.036
v99	0.101
v100	0
v101	-0.032
v102	0.002
v103	0.026
v104	0.0005
v105	-0.003
v106	0
v108	-0.078
v109	0.041
v111	0.118
v114	-0.022
v115	-0.033

v116	0
v117	-0.024
v118	-0.001
v119	-0.015
v120	0.050
v121	0
v122	0
v123	-0.010
v124	0
v126	0
v127	0.002
v128	-0.043
v129	0.073
v130	0
v3	-0.329
v24	0.065
v30	-0.044
v31	0.406
v47	-0.047
v52	0.005
v56	-0.007
v66	0.272
v71	0.049
v74	0.802
v75	-0.055
v79	-0.009
v91	-0.0002
v107	0.020
v110	0.240
v112	0.003
v113	-0.002
v125	0.002