BEN ISMAIL Kais (ENSTA 3A Finance quantitative)
DANG Ngoc Hao (Ensae 3A Finances et gestion des risques)
Said Samer (ENSTA 3A Finance quantitative)
FALL Ndeye Yacine (Ensae 3A Actuariat)

ENSAE 3ème Année
Machine Learning pour la Finance
Année scolaire 2018-2019

Projet 15: Machine learning for credit risk analysis.

Contents

1 Introduction

Il est indéniable que l'intelligence artificielle constitue un enjeu majeur pour le secteur financier. L'émergence des techniques de machine learning offre la possibilité aux institutions financières d'exploiter les grandes quantités de données qui sont en leur possession. Cet impact est d'autant plus grand du fait que dans le secteur financier, l'analyse des données est d'une importance capitale.

L'utilisation des algorithmes d'intelligence artificielle est parfois critiquée. En effet, pour certains il y a un risque d'une forme de déshumanisation de la société, avec un remplacement des humains par les machines. Il y a aussi des questions qui se posent sur la protection des données, sur la cybercriminalité entres autres.

Toutefois, l'intelligence artificielle est aussi source d'avantages pour la finance. La mise en place des techniques d'apprentissages statistiques en finance permet un gain de temps important lors du traitement de tâches lourdes ainsi que dans l'analyse de grands volumes d'informations. Les institutions financières peuvent aussi se servir de ses techniques pour mieux appréhender leurs différents risques. C'est le cas notamment des banques, pour qui la maitrise du risque de crédit en fondamentale. La diversité des algorithmes et leur complexité, obligent à être minutieux lors du choix des algorithmes à implémenter dans les modèles. Le choix dépend notamment, de l'objectif visée, par exemple appréhender le risque de défaut d'un client, des types de données utilisées, de leur granularité, de leur confidentialité, etc.

Dans ce qui suit nous allons nous intéresser à l'utilisation du Machine Learning pour l'analyse du risque de crédit. L'importance de ce champ d'application du machine learning sera détaillée un peu plus loin.

L'idée est de tester différentes techniques d'apprentissages supervisées afin de voir laquelle est mieux adaptée à l'analyse du risque de crédit et à nos données. La base de données sur laquelle le travail porte est disponible sur kaggle et porte sur les caractéristiques de prêts aux particuliers. Une présentation de la base de données sera faite aussi un peu plus loin. Notre objectif final est de juger de l'efficacité de certains algorithmes dans la prédiction de faire défaut ou non d'un client. Nous testerons les réseaux de neurones artificiels, les random forest ainsi qu'une régression logistique pour comparer leur péformance. Il sera important aussi d'apporter un regard critique aux résultats notamment en terme de choix des paramètres utilisés, des variables explicatives,etc.

Notre étude s'est basée sur les articles: A Neural Network Approach for Credit Risk Evaluation [1] et Credit Risk Analysis Using Machine and Deep Learning Models [2]. Les algorithmes ont été codés avec le logiciel R.

Le rapport s'articulera donc comme suit : une première section sera consacrée aux enjeux du machine learning pour l'analyse du risque de crédit, dans la section suivante la théorie derrière les différentes méthodes utilisées sera présentée, dans une troisième section, nous présenterons comment les méthodes ont été appliquées aux données avant de présenter les résultats obtenus. Pour finir, une discussion sur les méthodes utilisées ainsi que sur les résultats obtenus sera menée.

2 Enjeux du Machine Learnig pour le risque de crédit

Au cours des dernières décennies, la réglementation bancaire est devenue de plus en plus exigeante et complexe. En effet, le risque est inhérent à l'activité bancaire, il est alors important de le prendre en compte dans les choix de gestion d'une banque. Un des plus grands risques auxquels fait face une banque est le risque de crédit.

Le risque de crédit ou risque de contrepartie sur les marchés financiers peut être défini comme étant le risque de perte due au fait qu'un emprunteur ne rembourse pas, totalement ou partiellement, son crédit aux échéances prévues par le contrat signé entre lui le prêteur. On parle aussi de risque de défaut.

Une bonne évaluation du risque de crédit est très importante. Toutefois l'évaluation du risque de crédit n'est pas évidente. Différentes techniques ont été jusqu'à présent privilégiées par les banques. Une d'elle est le système de scoring qui permet de différencier les bons et les mauvais emprunteurs via un modèle affectant un score à chaque emprunteur. Les modèles utilisés se basent sur des variables explicatives pouvant différencier les emprunteurs selon leur risque. Dans le même esprit, la banque peut utiliser les ratings afin d'analyser le risque de crédit que représente une entreprise. Les banques peuvent aussi s'appuyer sur un système d'expertise. L'analyse du risque de crédit est plus subjective dans ce cas même si l'expertise se base sur des facteurs d'analyses clés.

Avec l'émergence de l'intelligence artificielle, les institutions financières font de plus en plus appel à des techniques d'apprentissages statistiques afin de mieux définir la probabilité de défaut d'un emprunteur et ainsi améliorer la gestion de leur risque financière.La disponibilité des données en grande quantité ainsi que la puissance des nouvelles technologies favorisent l'utilisation des techniques d'apprentissages statistiques pour évaluer le risque de crédit. En parallèle, ces dernières années, une vaste littérature académique s'est développée sur l'utilisation des méthodes de machine learning pour mieux cerner le risque de crédit. Toutefois, les régulateurs bancaires exigent le plus souvent des modèles simples, clairs, compréhensibles et vérifiables. En conséquence, l'utilisation des techniques d'apprentissages restera surement limitée et très encadrée. En effet, la plupart des méthodes d'apprentissages sont considérées comme étant des boites noires dont l'interprétation est complexe. De plus le règlement général sur la protection des données (RGPD) limite les possibilités de collection et d'utilisation des données des clients.

Une solution possible pour le premier problème serait d'utiliser une approche hybride au lieu de faire uniquement de l'apprentissage statistique. L'idée serait de se servir en premier temps des techniques d'apprentissages pour améliorer le pouvoir de discrimination des individus puis ensuite d'appliquer un modèle classique tel que la régression logistique. La performance globale du modèle pourrait ainsi être améliorée et la mesure du risque de crédit resterait interprétable et contrôlable.

Une solution pour le second problème serait que les banques précisent dans leurs conditions générales de vente la finalité que les données collectées peuvent avoir et demander l'autorisation des clients. Dans ce qui suit, différentes approches de Machine Learning pour modéliser la probabilité de défaut seront présentées.

3 Aspects théoriques

3.1 Présentation des données utilisées

Les données que nous avons utilisées sont fournies par la société Lending Club. Ces fichiers contiennent des données complètes pour tous les prêts émis au cours de la période 2007-2015, y compris l'état du prêt (actuel, en retard, entièrement payé, etc.) et les dernières informations de paiement. Des caractéristiques

supplémentaires telles que les credit score, le nombre de demandes de renseignements financiers, l'adresse, y compris les codes postaux, l'état et les recouvrements, sont présentes dans les données. Le fichier est une matrice d'environ 2260668 observations et 145 variables. Un dictionnaire de données est fourni dans un fichier séparé.

Pour un choix pertinant des variablesLes, on a commencé par retiré les variables vides ainsi que celles qui avaient plus de 10% de valeurs manquantes. Puis on a séléctionné les variables qui avaient une forte corrélation entre elles. Les variables explicatives choisies sont décrites ci-dessous:

- loan_amnt : Le montant indiqué du prêt demandé par l'emprunteur.
- emp_length : Durée de l'emploi en années. Les valeurs possibles sont comprises entre 0 et 10, où 0 signifie moins d'un an et 10, dix ans ou plus.
- grade : Grade d'emploi
- t rm : une période de 36 mois ou 60 mois
- int_rate : Taux d'intérêt du prêt
- a nual_inc : Le revenu annuel total
- application_type : Indique si le prêt est une demande individuelle ou une demande conjointe à deux co-emprunteurs
- loan_status : Variable avec plusieurs niveaux (hors charge, en cours, défaut, payé entièrement, etc.)
- home_ownership : Type de propriété du logement

Nous ajoutons la variable loan_outcome qui prend comme valeur 1 si la variable loan_status correspond aux niveaux (Charged of, Default), 0 si la variable loan_status correspond au niveau Fully Paid et prend No info sinon.

3.2 Modèles utilisés

3.2.1 Réseau de neurones feed-forward (FeedForward Neural Network)

En se basant sur l'idée de l'étude de *Eliana Angelini* et *Giacomo di Tollo* sur le risque de crédit, nous nous intéressons à la classification avec les réseaux de neurones. Dans l'article de E.Angelini et G. di Tollo, ils présentent deux méthodes: le réseau de neurones feed-forward classique et le réseau de neurones feed-forward avec la connection ad-hoc. Par définition, le réseau de neurones feed-forward classique est un réseau de neurones avec une couche d'entrée, deux couches cachées et une couche de sortie. D'un autre côté, le réseau de neurones feed-forward avec la connection ad-hoc est aussi un réseau de neurones feed-forward avec les neurones d'entrée groupés par trois et chaque groupe est connecté à une neurone de la couche suivant.

A cause de la limite des bases des données, ainsi que du temps, nous étudions le réseau de neurones feed-forward classique avec une unique couche cachée et l'apprenons par la rétropropagation. Pour travailler efficacement avec cette méthode, il nous faut normaliser certaines variables de la base de données en premier.
Normalement, la méthode de transformation linéaire de max-min est souvent utilisée pour normaliser. Cependant, pour exploiter le maximum d'informations utiles de la base des données, nous normalisons

les prédicteurs par la formule logarithmique définie comme suit:

$$\bar{x} = log_m(x + 1)$$

avec m, la valeur maximale de la variable analysée. Grâce à cette formule, toutes les valeurs normalisées sont plus petites que 1.
Pour finir, il faut effectuer des tests pour choisir la valeur optimale des paramètres pour le réseau de neurones feed-forward. Dans notre étude, nous considérons le taux d'apprentissage et l'élan (momentum) dont des valeurs optimales sont déterminées par la procédure d'optimisation systématique. En effet, dans cette procédure, nous sélectionnons pour chaque paramètre l'ensemble des valeurs v_i avec $i \in$ {learning.rate, momentum}. Ensuite, nous commençons la procédure d'optimisation systématique par la valeur initiale, en laissant inchangés la valeur du momentum etle nombre de neurones dans la couche cachée. Les valeurs varient dans l'ensemble $v_{learning.rate}$. Une fois la valeur optimale de learning.rate obtenue, nous continuons la procédure d'optimisation pour le deuxième paramètre. Le résultat de cette méthode est présenté dans la partie suivante.

3.2.2 Forêt aléatoire

Cet algorithme appartient à la famille des agrégations de modèles, c'est un cas particulier de bagging(bootstrap aggregating) appliqué aux arbres de décision de type CART (Classification And Regression Trees). Le principe des méthodes de Bagging, et donc en particulier des forêts aléatoires, est de faire la moyenne des prévisions de plusieurs modèles indépendants pour réduire la variance et donc l'erreur de prévision. Pour construire ces différents modèles, on sélectionne plusieurs échantillons bootstrap, c'est à dire des tirages avec remises. En plus du principe de bagging, les forêts aléatoires ajoutent de l'aléa au niveau des variables. Pour chaque arbre on sélectionne un échantillon bootstrap d'individus et à chaque étape, la construction d'un nœud de l'arbre se fait sur un sous-ensemble de variables tirées aléatoirement.

On se retrouve donc avec plusieurs arbres et donc des prédictions différentes pour chaque individu. Pour obtenir l'estimation finale, on choisit la catégorie la plus fréquente dans le cas d'une classification, tandis qu'on fait la moyenne des valeurs prédites dans le cas d'une régression. C'est un algorithme particulièrement performant pour les problématiques de prédiction (utile pour un nombre de variables explicatives important).

3.2.3 Régression logistique

La régression logistique est une technique prédictive. C'est une méthode très utilisée car elle permet de modéliser des variables binaires ou des sommes de variables binaires. Elle vise à construire un modèle permettant de prédire et expliquer les valeurs prises par une variable cible qualitative à partir d'un ensemble de variables explicatives quantitatives ou qualitatives (si on a plus de 2 modalités, on parle de régression logistique polytomique).

La régression logistique et la régression linéaire appartiennent à la même famille des modèles GLM (Generalized Linear Models) : dans les deux cas on relie un événement à une combinaison linéaire de variables explicatives. Pour la régression linéaire, la variable dépendante suit pas une loi normale $\mathsf{N}(\mu, s)$ où μ est une fonction linéaire des variables explicatives. Pour la régression logistique, la variable dépendante, aussi appelée variable réponse, suit une loi de Bernoulli de paramètre p (p la probabilité moyenne pour

que l'événement se produise), lorsque l'expérience est répétée une fois, ou une loi Binomiale (n, p) si l'expérience est répétée n fois. Le paramètre de probabilité p est ici une fonction d'une combinaison linéaire des variables explicatives.

La fonction la plus couramment utilisée pour relier la probabilité p aux variables explicatives est la fonction logistique (on parle alors de modèle Logit). Cette fonction est parfaitement symétrique et sigmoïde. L'expression analytique du modèle est donnée ci-dessous :

$$p = exp(\beta X)/(1 + exp(\beta X))$$

Où βX représente la combinaison linéaire des variables (constante comprise). La connaissance de la loi de distribution de l'événement étudié, permet d'écrire la vraisemblance de l'échantillon. Pour estimer les paramètres β du modèle (les coefficients de la fonction linéaire), on cherche à maximiser la fonction de vraisemblance. Contrairement à la régression linéaire, une solution analytique exacte n'existe pas. Il est donc nécessaire d'utiliser un algorithme itératif.

4 Application des modèles aux données et présentation des résultats

4.1 Application des modèles aux données

Le cadre d'étude ainsi que les données à notre disposition ayant été présentés, il s'agit dans cette partie de mettre en pratique sur nos données les modèles ressortis dans la partie précédente.
Nous avons ainsi produit sous le logiciel R: une forêt aléatoire, une régression logistique et un réseau de neurones .
Pour ce faire, nous avons partitionné de façon aléatoire le modèle en une base d'apprentissage contenant 70% des observations et une base test contenant 30% des observations, afin de contrôler le surapprentissage des différents modèle . Le fichier d'apprentissage sert en particulier à construire le modèle et le fichier test à valider le modèle.

4.1.1 Réseau de neurones feed-forward (FeedForward Neural Network)

Ce modèle a été implémenté sous R grâce à la fonction **mx.model.FeedForward.create** du package **mxnet**.
Nous commençons par la détermination des valeurs optimales des deux paramètres avec comme valeurs initiales choisies intuitivement : *momentum*= 0.9, *num_hidden*= 15 et le nombre de neurones de la couche de sortie est égale 2. Ce processus d'apprentissage est fait d'abord sur [0.01, 0.6] pour *learning.rate* et ensuite sur [0.1, 0.9] pour *momentum*. Les processus sont fait avec le function activation *relu* pour la couche cachée et *softmax* pour la couche de sortie.
Les valeurs optimales obtenues ne sont pas des constantes à cause de hasard des coefficients utilisés pour apprendre. Effectivement, la valeur optimale de taux d'apprentissage varie entre [0.33, 0.45], cependant la puissance de prédiction ne change pas vraiment. Après avoir relancé plusieurs fois le processus d'apprentissage, les valeurs obtenues varient autour de 0.4, cela permet de choisir 0.4 comme la valeur optimale de taux d'apprentissage qui est utilisée pour trouver l'élan (*momentum*) optimal. De la même

manière, nous trouvons que la valeur optimale de l'élan est égale 0.7.

Une fois les paramètres déterminées, nous les appliquons pour l'apprentissage par réseau de neurones feed-forward. Les résultats sont affichés dans le tableau suivant:

Nombre de neurones	Apprentissage	Test
15	0.80402209	0.80343514
16	0.80402209	0.80343514
17	0.80402209	0.80343243
18	0.80402326	0.80343243
19	0.80402326	0.80343243
20	0.80402326	0.80342973
21	0.80402442	0.80342973
22	0.80402326	0.80342973
23	0.80402442	0.80342973
24	0.80402442	0.80342973
25	0.80402442	0.80342973

Table 1: Performance du réseau feed-forward selon le nombre de neurones.

Le tableau résume les performances obtenues pour le modèle selon le nombre de neurones dans la couche cachée. Il nous montre la stabilité de la performance pour la base de données test. En effet, la performance est constante lorsqu'on fait varier entre 20 et 25 le nombre de neurones de la couche cachée, tandis qu'il existe une faible volatilité des performances calculées sur la bases d'apprentissage.

Nombre de neurones	Specificity	Sensitivity
15	0.99181	0.03682
16	0.99181	0.03682
17	0.99181	0.03682
18	0.99181	0.03682
19	0.99181	0.03682
20	0.99181	0.03682
21	0.99181	0.03682
22	0.99181	0.03682
23	0.99181	0.03682
24	0.99181	0.03682
25	0.99181	0.03682

Table 2: Sensitivity et specificity du réseau feed-forward selon le nombre de neurones sur la base d'apprentissage.

Nombre de neurones	Sensitivity	Specificity
15	0.03478	0.99143
16	0.03478	0.99143
17	0.03478	0.99143
18	0.03478	0.99143
19	0.03478	0.99143
20	0.03478	0.99143
21	0.03478	0.99143
22	0.03478	0.99143
23	0.03478	0.99143
24	0.03478	0.99143
25	0.03478	0.99143

Table 3: Sensitivity et specificity du réseau feed-forward selon le nombre de neurones sur la base de test.

4.1.2 Forêt aléatoire

Ce modèle a été implémenté sous R grâce au package **RandomForest**.

Les paramètres suivants étaient présents dans le modèle :

- ntree: désigne le nombre d'arbres intervenants dans le modèle
- mtry : indique le nombre de variables testées à chaque découpage. La valeur par défaut étant la racine carrée du nombre de variables. C'est le principal paramètre à modifier, avec ntree.
- subset : indique si on ne travaille que sur certaines parties de notre jeu de données.
- replace : spécifie si l'echantillonage a été fait avec ou sans remise.
- nodesize : spécifie la taille maximale des noeuds (plus elle est élevée, plus les arbres seront courts).

Pour notre projet, nous avons joué sur 2 paramètres ntree et mtry et laissé les autres paramètres par défaut. Nous avons testé ntree = 100 à 300 et mtry= 1 à 9. Nous avons tout d'abord testé pour ntree fixé à 50, des valeurs de mtry allant de 1 à 9 et avons obtenu les résultats suivants :

mtry	erreur OOB
1	19.62 %
2	19.51%
3	20.05%
4	21.39%
5	22.46%
6	22.87
7	23.06%
8	23.23%
9	23.18%

Table 4: optimisation mtry.

On choisit donc comme mtry celui qui minimise l'erreur OOB à savoir mtry=2.

L'erreur out of bag : on rappelle que chaque arbre est entraîné sur une fraction des données, qui est considérée comme « in-bag ». Ce qui permet à chaque arbre, une fois construit, d'estimer son taux d'erreur sur les données qu'il a laissé « out-of-bag » . Plus ce taux est faible, plus le modèle est juste. Ce chiffre à lui seul pourrait servir d'indicateur de performance du modèle.

En se basant sur cette valeur de mtry, nous otimisons également le nombre d'arbres et nous obtenons comme valeur optimale : 150 correspondant à une erreur out of bag de :19, 48%

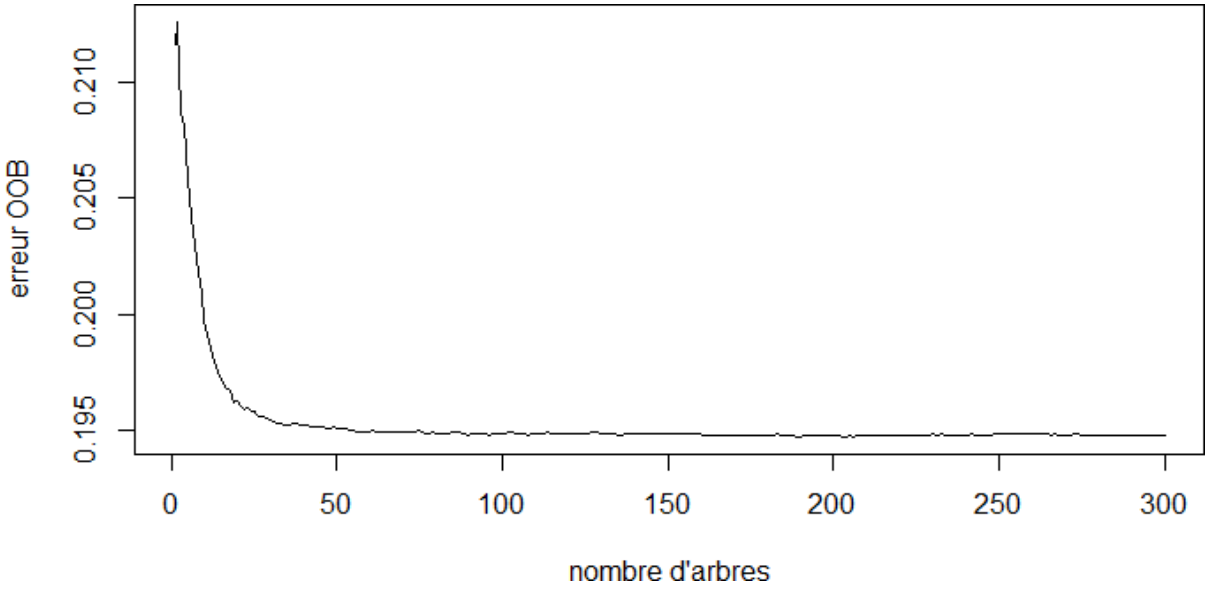

Figure 1: erreur OOB en fonction du nombre d'arbres

- Le modèle fournit également les variables les plus influentes sur les résultats à savoir : le taux d'intérêt, le revenu annuel et le montant du prêt.

On obtient ainsi dans notre modèle :

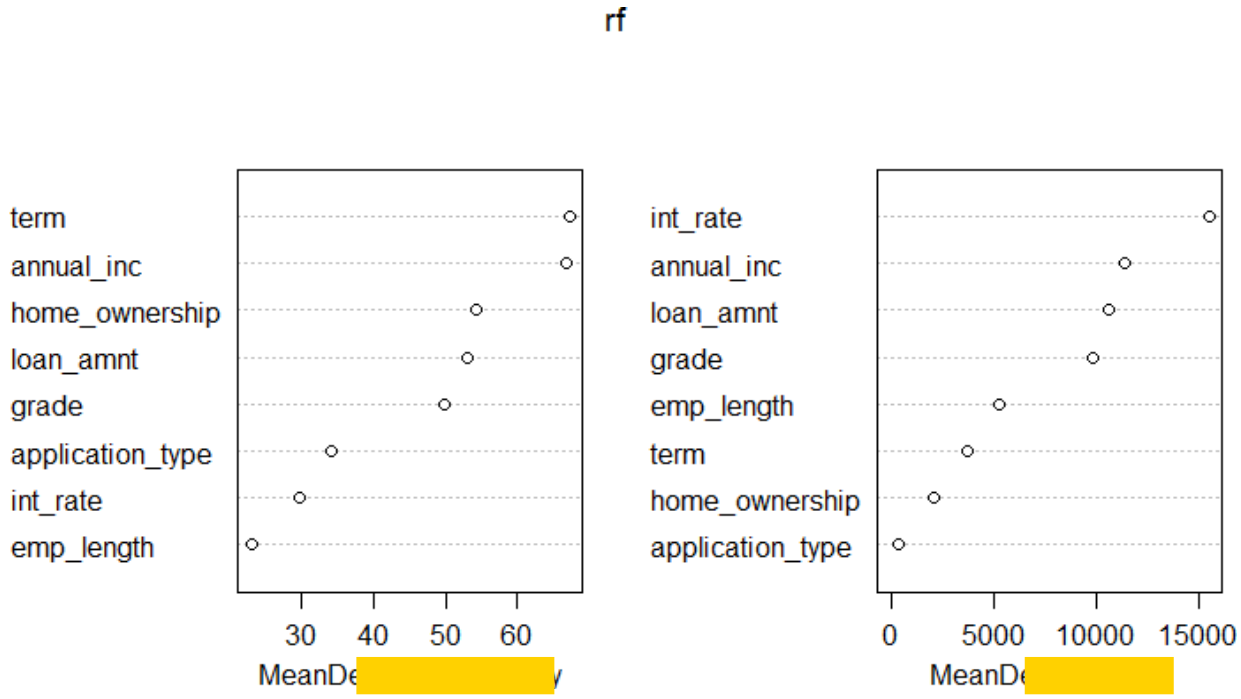

Figure 2: importance variable

4.1.3 Regression logistique

Ce modèle a été implémenté sous R grâce à la fonction **glm**.
Les variables significatives qui ressortent sont : le montant prêté, une durée d'emprunt de plus de 10 ans, un rating entre B et G, le taux d'intérêt, le fait de posséder ou non une maison et le revenu annuel.

	Estimate	Std. Error	z value	Pr(>|z|)
(Intercept)	-3.1861	0.0208	-153.01	0.0000
loanamnt	0.0000	0.0000	29.02	0.0000
emplength1 year	0.0170	0.0142	1.19	0.2326
emplength10+ years	-0.0486	0.0109	-4.46	0.0000
emplength2 years	-0.0167	0.0132	-1.26	0.2059
emplength3 years	-0.0043	0.0136	-0.32	0.7525
emplength4 years	-0.0162	0.0147	-1.10	0.2725
emplength5 years	-0.0185	0.0146	-1.27	0.2050
emplength6 years	-0.0512	0.0160	-3.20	0.0014
emplength7 years	-0.0345	0.0162	-2.13	0.0332
emplength8 years	0.0036	0.0161	0.23	0.8219
emplength9 years	-0.0031	0.0170	-0.18	0.8566
gradeB	0.6546	0.0142	46.04	0.0000
gradeC	1.0622	0.0176	60.48	0.0000
gradeD	1.2532	0.0232	54.12	0.0000
gradeE	1.3764	0.0292	47.12	0.0000
gradeF	1.3950	0.0372	37.53	0.0000
gradeG	1.4421	0.0471	30.64	0.0000
term 60 months	0.4234	0.0071	59.72	0.0000
intrate	0.0417	0.0018	22.80	0.0000
annualinc	-0.0000	0.0000	-30.69	0.0000
applicationtypeJoint App	-0.0151	0.0216	-0.70	0.4844
homeownershipOTHER	0.5179	0.2598	1.99	0.0462
homeownershipOWN	0.1902	0.0099	19.29	0.0000
homeownershipRENT	0.3478	0.0063	55.00	0.0000

On remarque ainsi que plus la durée du prêt augmente, moins l'on a de risque de faire défaut.

Par ailleurs, plus le taux d'intérêt est élevé, plus le risque de défaut augmente. Aussi, plus le revenu annuel augmente, plus la probabilité de faire défaut diminue.

4.2 Etude de la performance des modèles et résultats

En classification binaire, l'on utilise plusieurs outils basés sur la **matrice de confusion** afin d'avoir une idée de la performance du modèle . La matrice de confusion se présente comme suit :

	observés faux	observés vrai
prédits faux	vrais négatifs(VN)	faux négatifs(FN)
prédits vrai	faux positifs (FP)	vrais positifs(VP)

Table 5: matrice de confusion.

Dans notre cas d'etude:

- les **vrai positifs** sont les individus que le modèle a prédit comme faisant défaut et qui ont effectivement fait défaut.
- les **faux négatifs** représentent les individus que le modèle prédit comme n'ayant pas fait défaut mais qui en réalité font défaut.
- les **faux positifs** sont les individus que le modèle prédit comme risqués mais qui en réalité ne sont pas risqués.
- les **vrai négatifs** sont ceux que le modèle prédit comme n'etant pas risqués et qui ne sont effectivement pas risqués.

Une fois la matrice de confusion construite, l'on peut faire ressortir les critères d'evaluation suivants :

- sensibilité $=\frac{VP}{VP+FN}$: représente la proportion d'individus risqués correctement identifiés par le modèle.
- spécificité$=\frac{VN}{VN+FP}$: représente la proportion d'individus non risqués correctements identifiés par le modèle.
- taux accuracy$=\frac{VP+VN}{nombre\ d\ observations}$: représente la probabilité de bien classer un individu dans la base. C'est la portion d'individus dont le statut (défaut/pas défaut) a été correctement identifié par le modèle

En plus de ces critères, nous évaluons également la stabilité du modèle, plus particulièrement la robustesse au surapprentissage.

En se basant sur ces critères, nous obtenons les résultats suivants :

- GLM : Dans les tableaux suivants, les lignes représentent les valeurs prédites et les colonnes les valeurs observées.

	0	1
0	685851	163006
1	4961	5701

Table 6: matrice de confusion trainset.

	0	1
0	293757	70146
1	2082	2381

Table 7: matrice de confusion testset.

metrics	resultats glm
stability	0.06
accuracy-test	0.80
sensitivity-test	0.033
specificity-test	0.99

Le modèle est stable avec un taux d'erreur faible et estime bien la classe des 0 avec comme spécificity 99%.

- Random forest

	0	1
0	683890	160490
1	6922	8217

Table 8: matrice de confusion trainset.

	0	1
0	292906	69110
1	2933	3417

Table 9: matrice de confusion testset.

metrics	results_rf
stability	0.08
accuracy-test	0.804
specificity-test	0.99
sensitivity-test	0.047

Table 10: metrics table.

On se rend compte que tout comme la régression logistique, le modèle est stable dans le sens où il ne surapprend pas. Il a un bon taux d'accuracy et estime très bien la classe des 0 (99% de réussite) tandis qu'il y a un problème d'estimation en ce qui concerne la classe des 1 (4,7% de réussite). Il estime donc mieux la classe des 0.Néanmoins, il estime mieux la classe des 1 comparé au modèle logistique.

On remarque de façon globale que les modèles détectent mieux le fait de ne pas faire défaut. En effet, la spécificité est toujours plus élevée que la sensibilité.
Les différents modèles sont également tous stables. Mais l'on constate que la régression logistique a la meilleure sensibilité, est stable et a un bon taux d'accuracy.

Ci dessous, un aperçu du résultat pour 10 lignes de la base :

	loanamnt	grade	term	int_rate	annualinc	loanoutcome	proba	pred.rf
873389	18000	E	36 months	21.00	65000.00	0	0.19	0
862979	8000	B	36 months	12.12	29000.00	0	0.01	0
34504	10000	B	60 months	9.44	33655.00	0	0.08	0
747619	35000	B	36 months	9.44	124800.00	1	0.01	0
1087346	15000	C	60 months	13.35	85000.00	0	0.01	0
61113	29800	B	36 months	10.49	175000.00	0	0.01	0
1025502	5000	D	36 months	16.29	23000.00	0	0.03	0
934839	19000	A	36 months	7.90	120000.00	0	0.00	0
435284	10000	B	36 months	11.53	50000.00	0	0.02	0
695520	20000	B	36 months	11.99	70000.00	0	0.00	0

Table 11: résultat random forest.

	loanamnt	grade	term	int_rate	annualinc	loanoutcome	proba	preds.logit
632846	8000	C	36 months	13.66	51000.00	0	0.22	0
46769	15000	C	36 months	14.49	260000.00	0	0.17	0
678740	19200	C	36 months	14.99	69540.00	0	0.19	0
222701	19800	B	36 months	8.49	56650.00	0	0.15	0
1045767	30000	D	60 months	15.61	100000.00	0	0.31	0
256145	18000	E	60 months	24.74	90000.00	1	0.50	0
769579	28000	B	36 months	8.24	315000.00	0	0.07	0
958447	16000	D	36 months	17.86	80500.00	0	0.22	0
32989	1750	A	36 months	6.08	88000.00	0	0.06	0
509485	21000	A	36 months	5.32	54000.00	0	0.05	0

Table 12: résultat regression logistique.

5 Discussion des résultats

Les forêts aléatoires et les réseaux de neurones sont reputés en général pour leur bonne qualité de prédiction comparés aux autres modèles.

Néanmoins lorsqu'il existe une relation linéaire entre les variables, l'on constate de meilleurs résultats avec la régression logistique.

Plusieurs facteurs peuvent également expliquer les résultats obtenus par nos modèles, notamment:

5.1 Les valeurs manquantes

Les données manquantes peuvent causer de sérieux problèmes dans le processus de prédiction. En effet, celles ci peuvent entrainer la détérioration des performances en terme d'erreurs des techniques utilisées.
Il s'avère que notre base contient énormément de données manquantes.
Par exemple sur la base, sur 145 variables, 44 variables ont plus de 30% de valeurs manquantes.
Il a donc fallu gérer ces données manquantes et pour cela, 2 choix s'imposaient à nous : l'imputation ou la suppression. Ainsi, une suppression des donnée manquantes a été effectuée.
Néanmoins, la suppression réduit le champ d'observations et donc le modèle a moins de données pour apprendre, pouvant impacter ainsi sa performance.

5.2 Déséquilibre des classes

Notre base de données présentait une proportion très déséquilibrée entre la classe des 0 (c'est-à-dire ceux qui ne font pas défaut) et la classe des 1 (ceux qui font défaut). Ces derniers sont en effet clairement en minorité (environ 17%).
Les problèmes de ce type peuvent créer toutes sortes de difficultés dans des modèles prédictifs. En effet, dans notre cas, il a été facile d'obtenir une accuracy d'environ 80% avec une prédiction comportant quasiment que des individus de classe 0. En effet, la prévalence importante de cette classe nous amènerait à un niveau d'accuracy apparemment élevé.
Un autre problème avec le déséquilibre des classes est qu'il a une forte incidence sur la performance des algorithmes d'apprentissage car ceux ci tendent à ignorer la classe minoritaire compte tenu de son manque de présence statistique. Cela est particulièrement problématique dans les situations où cette classe minoritaire est exactement la plus pertinente comme c'est le cas dans notre problème.
Néanmoins, plusieurs techniques ont été développées dans le but d'aider les algorithmes d'apprentissage à surmonter le problème soulevé par le déséquilibre des classes à savoir : les méthodes d'échantillonnage qui manipulent les données d'apprentissage pour changer la répartition des classes.
L'idée est de générer artificiellement de nouveaux exemples de la classe minoritaire en utilisant les plus proches voisins de ces cas ou de sous echantilloner les exemples de classes majoritaires, ce qui mène à un ensemble de données plus équilibré. En ce qui nous concerne, nous avons procédé au sous échantillonnage de la classe 0 et avons analysé l'impact sur nos métriques. On a réalisé un premier test avec une base qui contenait 50% de 0 et 50% de 1 et un second test avec 60% de 0 et 40% de 1.

metricsbis	results_rf1	results_rf2
proportion rééquilibrage	50-50	40-60
accuracy-test	0.647	0.665
sensitivity-test	0.677	0.430
specificity-test	0.617	0.822

Table 13: résultat rééquilibrage des classes.

On se rend donc compte que: plus la classe est prévalente dans la base, mieux elle est prédite par l'algorithme.

5.3 Paramètres utilisés dans le modèle

Les paramètres utilisés pour construire les modèles peuvent avoir joué un rôle sur la performance des modèles.
En effet, nous avons testé différentes valeurs de paramètres mais il était impossible de toutes les tester. Des valeurs autres que celles utilisées auraient peut être permis d'aboutir à de meilleurs résultats.
Par exemple, nous n'avons pas pu optimiser tous les paramètres,dans la mesure où certaines fonctions telles que randomForest et mx.mlp disposent de nombreux paramètres. Aussi, l'on pourrait sans doute améliorer nos résultats en augmentant le nombre de couches cachées de notre modèle de réseaux de neurones.
La problématique du seuil optimal de probabilité est également un paramètre important à considérer, comme le révèle l'étude menée par Gilbert Reibnegger et Walter Schrabmair qui consiste à trouver le seuil de coupure binaire optimal d'un test de diagnostic.
En ce qui concerne notre étude, dans les résultats précédents, nous avions fixé comme seuil de probabilité 0.5. L'on s'est rendu compte que ce seuil nous permettait d'obtenir un taux d'accuracy assez élevé. cependant, l'accuracy n'est pas une métrique suffisante dans la mesure où l'on est face à un problème de classification binaire et que la classe pertinente ici c'est le défaut. Les métriques d'intérêt seraient donc la sensibilité et la spécificité.
De ce fait, l'on a jugé intéressant d'effectuer des variations du seuil de probabilité et d'analyser leur impact sur les paramètres permettant de calculer ces métriques.
Suite à celà, l'on s'est rendu compte que sur le modèle de régression logistique par exemple, plus on baisse le seuil de probabilité, plus le nombre de vrais positifs s'améliore ainsi que le nombre de faux négatifs. Mais plus le seuil augmente, plus la prédiction des non risqués s'améliore.

	threshold	good_prediction	true_positive	true_negative	false_negative	false_positive
1	0.10	141921	67302	74619	5225	221220
2	0.20	232389	48622	183767	23905	112072
3	0.30	280699	26488	254211	46039	41628
4	0.40	294061	11030	283031	61497	12808
5	0.50	296138	2381	293757	70146	2082
6	0.60	295854	65	295789	72462	50

Un axe d'amélioration possible serait donc de trouver un seuil qui permettrait d'avoir un bon équilibre .

5.4 Impact du choix des variables

Afin de mener notre étude, un choix de variables explicatives nous semblant pertinentes a été fait. Le choix de variables peut jouer sur la stabilité du modèle utilisé. Le choix d'un nombre réduit de variables explicatives est important dans la pratique. En effet, il est pratique d'avoir un modèle parcimonieux notamment pour optimiser le temps de calcul. De plus, en demandant au client de fournir un nombre réduit d'informations, on peut réduire le délai de prise de décision de prêter ou non au client.
Toutefois, les variables choisies doivent bien décrire le risque que l'emprunteur représente. Il est important de détecter les variables pertinentes. Pour mesurer l'importance des variables, il est possible d'utiliser la "permutation feature importance". La méthode consiste à comparer les performances du modèle en prédiction avec et sans la variable dont on veut mesurer l'importance.

6 Conclusion

Au terme de ce projet, nous retenons que le défaut est déterminé par plusieurs facteurs . Plus précisement, le taux d'intérêt,la durée d'emprunt et le montant prêté.
Afin de faire ressortir ces variables, 3 modèles ont été implémentés : les forêts aléatoires, le modèle logistique et les réseaux de neurones. Ces modèles estiment bien la classe de ceux qui ne font pas défaut mais estiment moins bien ceux qui font défaut.
Néanmoins plusieurs facteurs ont pu altérer la qualité de prédiction de ce modèle parmi lesquels la qualité des données à savoir la présence de données manquantes, le caractère déséquilibré des classes et les variables que nous avions à notre disposition.
En ce qui concerne les pistes d'amélioration du problème, l'on pourrait :

- tester d'autres modèles tels que l'extrême gradient boosting qui est réputé pour ses bonnes performances lors des concours Kaggle.
- étudier en profondeur la problématique du seuil optimal de probabilité.

References

[1] Angelini, Eliana, Giacomo di Tollo, and Andrea Roli. *A neural network approach for credit risk evaluation.* 2008.

[2] Peter Martey Addo, Dominique Guégan and Bertrand Hassani *Credit Risk Analysis Using Machine and Deep Learning Models*.2018.

[3] Interpretable Machine Learning,Christoph Molnar, https://christophm.github.io/interpretable-ml-book/, 2019, A Guide for Making Black Box Models Explainable

[4] Fatma Hamdi. *Apprentissage en distributions déséquilibrées*, https://lipn.univ-paris13.fr/~bennani/THESES/These_Hamdi.pdf, 2012

[5] Landau, I., Landau *Data mining et machine learning dans les big data*. 2016

[6] R.Rakotomalala. *Traitement des classes déséquilibrées*, http://eric.univ-lyon2.fr/~ricco/tanagra/fichiers/fr_Tanagra_Imbalanced_Dataset.pdf, 2015

[7] Farrokh Habibzadeh, Parham Habibzadeh,and Mahboobeh Yadollahie *On determining the most appropriate test cut-off value: the case of tests with continuous results*, https://www.ncbi.nlm.nih.gov/pmc/articles/PMC5082211/, 2015

[8] Gilbert Reibnegger and Walter Schrabmair *Optimum binary cut-off threshold of a diagnostic test: comparison of different methods using Monte Carlo technique*, https://www.ncbi.nlm.nih.gov/pmc/articles/PMC4253606/, 2014