Projet : Prédiction de Défauts de Paiement

SINADINOVIC Marko

Data Mining 2025

Contents

1 Introduction **2**

2 Pré-traitement des données **2**
2.1 Découverte des données . . . 2
2.2 Suppression . . . 3
2.3 Données manquantes . . . 3
2.4 Conversions . . . 3

3 Analyse exploratoire des données **3**
3.1 Histogramme . . . 3
3.2 Box-plot . . . 6
3.3 Density plots . . . 7
3.4 Vérification . . . 10
3.4.1 Density Plot . . . 10
3.4.2 Mosaïque . . . 10
3.4.3 Tableau de contingence . . . 11
3.5 Liens . . . 11
3.6 Déduction . . . 12

4 Définition de la méthode d'évaluation des classifieurs **12**
4.1 Taux et indices . . . 13
4.1.1 Taux de succès . . . 13
4.1.2 Mesures de sélection d'attribut - Les différents indices . . . 13
4.2 Matrice de confusion . . . 14
4.3 Matrice de coût . . . 15
4.4 Les mesures . . . 16
4.5 Courbe ROC et AUC . . . 17
4.5.1 Courbe ROC . . . 17
4.5.2 L'air sous la courbe - AUC . . . 18

5 Définition des données d'apprentissage et de test **19**
5.1 Avec des valeurs inconnus . . . 19
5.2 Traitement des valeurs inconnues . . . 19
5.2.1 La moyenne . . . 19
5.2.2 Réduction de la base de données . . . 20

6 Construction et évaluation des classifieurs **20**
6.1 Rpart . . . 20
6.2 C50 . . . 22
6.3 TREE . . . 23
6.4 Random Forest . . . 24
6.5 K-Nearest Neighbors . . . 25
6.6 Support Vector Machine . . . 26
6.7 Naïve Bayes . . . 27
6.8 Réseau de neurones . . . 28

7 Choix du classifieur le plus performant **29**
7.1 Comparaison générale . . . 29
7.2 Structure . . . 31
7.3 Performances . . . 31

8 Résultats de l'application du classifieur **32**
8.1 Pré-traitement des données . . . 32
8.2 Prédiction . . . 32
8.3 Prédiction des probabilités . . . 32
8.4 Les résultats . . . 33

9 Conclusion **34**

10 Bibliographie et sources **35**

1 Introduction

Nous avons pour objectif de créer un modèle de prédiction du risque de défaut de paiement pour les clients. Pour ce faire, nous disposons d'un fichier au format `.csv` contenant de nombreuses données concernant les clients, notamment l'information sur leur éventuel défaut de paiement. Nous allons étudier les différentes variables et les exploiter afin d'analyser leurs liens, leur utilité et leur impact sur le défaut de paiement. Ensuite, nous utiliserons nos connaissances et les outils vus en cours pour créer un modèle de prédiction efficace. Afin d'élaborer ce modèle, nous comparerons plusieurs approches en recourant à des méthodes probabilistes.

2 Pré-traitement des données

2.1 Découverte des données

Nous remarquons lors de l'ouverture du fichier `Data Projet.csv` et grâce au document présentant le sujet que nous disposons de 6000 instances classées (où l'on sait si les clients sont en défaut de paiement). Cependant, en analysant les variables à notre disposition, certaines sont constantes, comme la variable `categorie`, et d'autres contiennent des valeurs manquantes pour certains clients, notamment les variables `age` et `adresse`. Par ailleurs, nous constatons que la variable `education` est représentée sous forme de chaînes de caractères. De plus, nous disposons de données discrètes et continues, ce qui nous indique la manière dont nous devons analyser ces informations.

2.2 Suppression

Dans un premier temps, nous traitons la variable `categorie`, dont le domaine de valeurs est constant. Puisqu'elle n'apporte aucune information déterminante pour la suite, nous ne pouvons pas l'exploiter. Ainsi, nous supprimons la colonne correspondante de notre fichier `Projet Data.csv`.

2.3 Données manquantes

Dans un second temps, nous devons remplacer les valeurs `999` indiquant des données manquantes par une autre valeur afin de ne pas biaiser notre analyse et les résultats de notre modèle par la suite. Nous pouvons soit remplacer les `999` des colonnes par la moyenne de l'ensemble des données respectives aux colonnes `age` et `adresse`, soit omettre ces valeurs en les remplaçant par `NA` afin qu'elles ne soient pas prises en compte.

2.4 Conversions

Enfin, nous devons traiter les variables `education` et `defaut`, dont les domaines de données sont représentés sous forme de chaînes de caractères.

En ce qui concerne la variable `defaut`, qui constitue notre variable cible, son domaine étant composé de chaînes de caractères, nous ne pouvons pas l'exploiter directement. Pour ce faire, nous allons convertir cette variable en facteur, ce qui permettra d'attribuer la valeur 2 pour `Oui` et la valeur 1 pour `Non`.

Pour la variable `education`, nous procédons de la même manière, à un détail près : nous la convertissons en variable ordonnée. Par exemple, la chaîne de caractères `Bac+2` se verra attribuer la valeur 2 et l'ordre 2, tandis que `Bac+5 et plus` se verra attribuer la valeur 5 et l'ordre 5, indiquant un rang supérieur dans la hiérarchie. En somme, "Niveau Bac = 1" < "Bac+2 = 2" < "Bac+3 = 3" < "Bac+4 = 4" < "Bac+5 et plus =5".

Cela facilitera la construction de notre modèle, car nous travaillerons désormais avec des valeurs ayant une signification, plutôt qu'avec des chaînes de caractères, lesquelles sont inexploitées par nos classifieurs.

3 Analyse exploratoire des données

Terminant le pré-traitement de nos données, nous pouvons commencer l'analyse exploratoire de celles-ci. Elle consiste à identifier les problèmes liés aux variables, à découvrir les relations entre les différentes données et à obtenir un premier aperçu des variables les plus déterminantes pour notre modèle.

3.1 Histogramme

Comme nous disposons de données discrètes et continues, nous devons les traiter avec précaution et de manière différente. Dans un premier temps, nous avons créé plusieurs histogrammes de manière naïve et avons remarqué que cette technique de visualisation est inutile pour certaines variables. Il est donc essentiel de bien découper nos domaines en intervalles pertinents. Dans un second

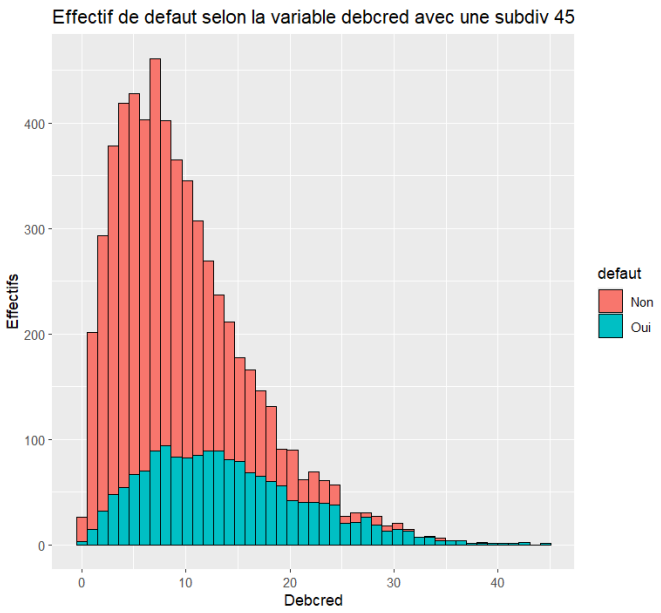
temps, nous avons ajusté le paramétrage des différents histogrammes, notamment en modifiant le partitionnement des domaines de données. En effet, un bin correctement choisi aide à identifier les tendances, les concentrations et les anomalies qui seraient difficiles à observer dans le fichier `.csv`. Enfin, nous avons trié nos histogrammes en fonction de leur pertinence pour l'étude de prédiction des défauts de paiement et avons retenu les graphiques suivants :

- Le nombre de défaut de paiement selon la variable **debcred**
- Le nombre de défaut de paiement selon la variable **emploi** (temps dans l'entreprise)
- Le nombre de défaut de paiement selon la variable **age**
- Le nombre de défaut de paiement selon la variable **education**

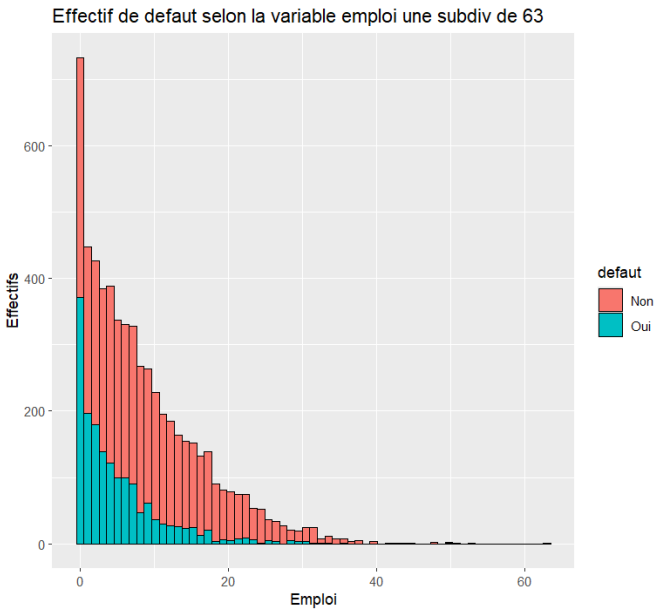
Nous n'avons pas retenu les autres variables pour les raisons suivantes :

- La variable **autre** ne se prête pas à une représentation sous forme d'histogramme, car une grande quantité de clients ont une dette comprise entre 0.009 et 40. Cela rend l'histogramme inutilisable et illisible lorsque l'on partitionne notre domaine en intervalles de plus en plus petits.
- La variable **debcarte** n'est pas exploitable sous forme d'histogramme, car il faudrait attribuer une grande valeur au bin pour obtenir une tendance, ce qui la rend illisible.

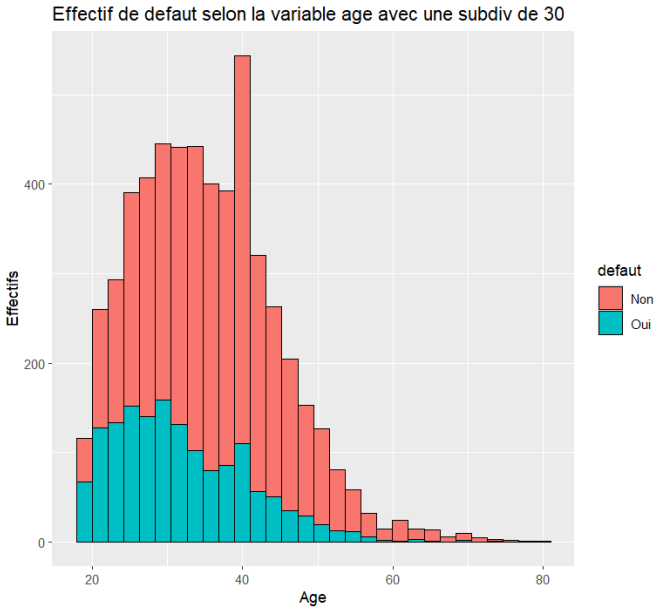
Passons à une brève étude des différents histogrammes.



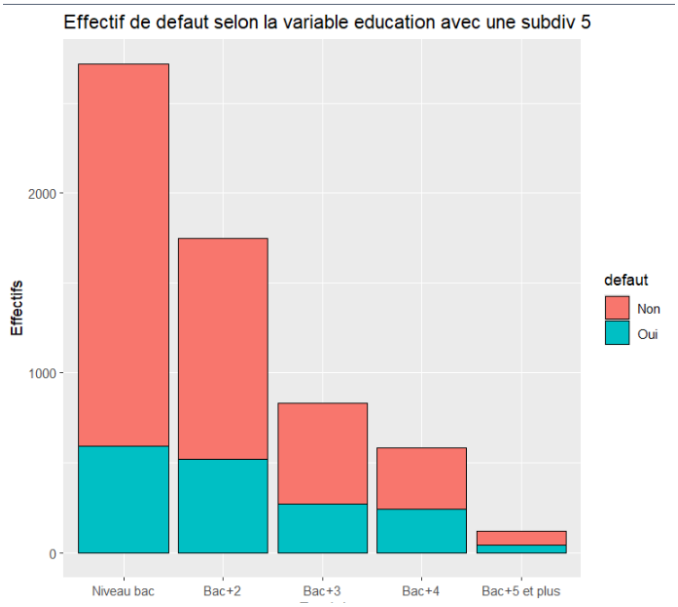
La variable `debcred` semble être un bon indicateur déterminant, car en analysant l'histogramme, on remarque que la majorité des cas de défaut de paiement se retrouvent dans les premières partitions. Le fait que ces cas se concentrent principalement dans les intervalles inférieurs à une certaine valeur de `debcred` suggère une relation potentiellement significative entre cette variable et le risque de défaut. Nous devons approfondir notre analyse en utilisant des outils métriques comme l'indice de Gini afin de confirmer cette intuition.



Les clients récemment embauchés semblent présenter un risque accru de défaut de paiement en raison de leur faible stabilité financière. Leur manque d'ancienneté les rend plus vulnérables aux imprévus économiques, comme la perte d'emploi ou une baisse de revenus. Cependant nous devons mettre en lien l'ancienneté dans l'entreprise avec l'âge des clients pour confirmer notre analyse.



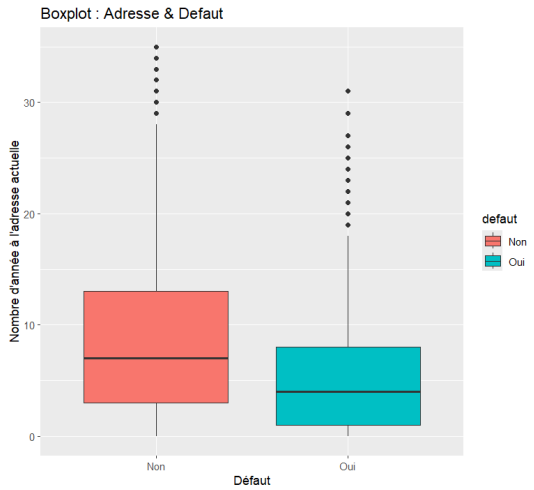
Cet histogramme révèle que la tranche d'âge jeune, particulièrement entre 18 et 40 ans avec un pic aux alentours de 30 ans, présente une plus grande concentration de défauts de paiement. Cette tendance suggère que les jeunes emprunteurs peuvent être en défaut de paiement dans un emprunt.



On constate que la plupart des défauts de paiement concernent des personnes ayant uniquement un niveau Bac. À l'inverse, pour ceux qui ont atteint un niveau plus élevé, le nombre de défauts diminue, ce qui suggère qu'un meilleur niveau d'éducation pourrait être lié à une plus grande stabilité financière et donc à un risque de défaut réduit. Cependant nous devons prendre en compte l'effectif élevé des clients ayant que le niveau Bac comparé aux clients ayant un Bac+4 et plus.

3.2 Box-plot

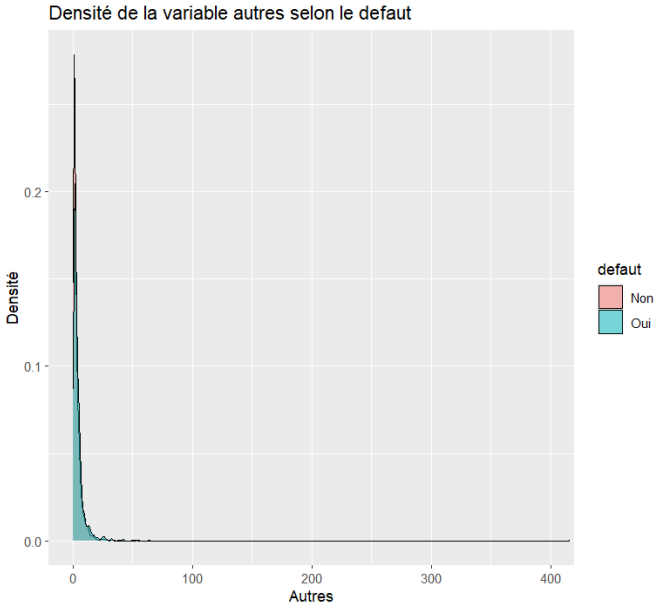
Après avoir exploré la distribution des défauts à l'aide d'histogrammes, nous allons maintenant nous concentrer sur les variables **autres, adresse et debcarte** en utilisant des boxplots. Cette approche nous permettra de mettre en évidence des tendances et des liens avec le défaut de paiement qui n'étaient pas apparents dans les histogrammes. Cependant, en créant les boxplots pour les variables **autres** et **debcarte**, nous remarquons qu'ils ne sont pas exploitables en raison de leur distribution. Ainsi, nous nous concentrerons uniquement sur l'analyse du boxplot suivant.



Nous remarquons que pour les clients sans défaut, la médiane se situe aux alentours de 10 ans et que la plupart des valeurs se regroupent entre environ 5 et 15 ans. En revanche, pour les clients en défaut de paiement, la médiane est plus basse, autour de 7 ans, avec un intervalle interquartile allant de 3 à 12 ans, ce qui suggère une faible stabilité. On remarque également quelques valeurs aberrantes dans chaque groupe. Ainsi, ce graphique indique qu'une durée plus longue à la même adresse est potentiellement associée à un risque réduit de défaut de paiement. Après avoir exploré la dispersion, nous devons étudier les deux variables restantes avec d'autres méthodes, mais aussi confirmer les tendances reconnues lors de nos analyses des histogrammes.

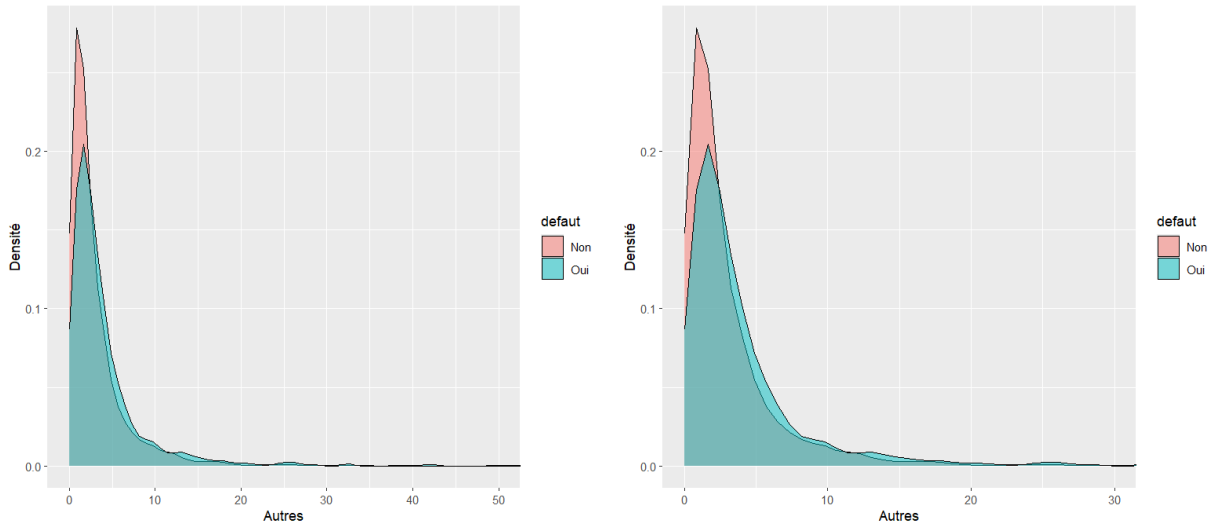
3.3 Density plots

Cette nouvelle approche graphique nous permettra d'observer des tendances non révélées par les histogrammes, car nous disposerons de graphiques qui ne dépendent pas du paramétrage du partitionnement et qui sont sous la forme de courbes. Intéressons-nous, dans un premier temps, au graphique représentant l'ensemble du domaine de données de la variable **autres**.

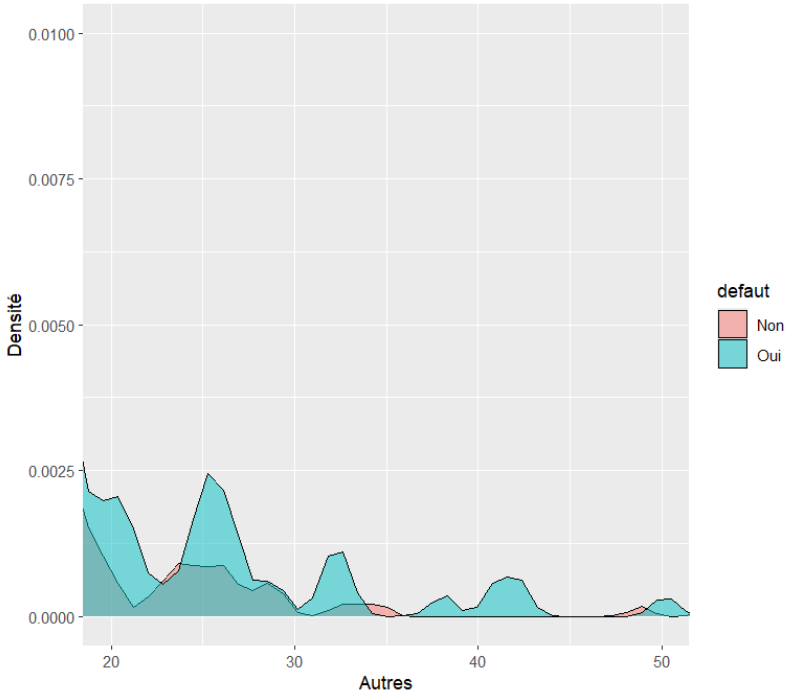


À ce stade, nous ne pouvons pas exploiter pleinement le graphique. Nous devons zoomer sur nos courbes, ce qui revient à limiter notre plage de domaine à un intervalle bien choisi où la majorité des clients se trouvent, correspondant ainsi à notre pic.

Limitons-nous aux intervalles [0, 55] et [0, 35] afin d'obtenir une meilleure visualisation des tendances.

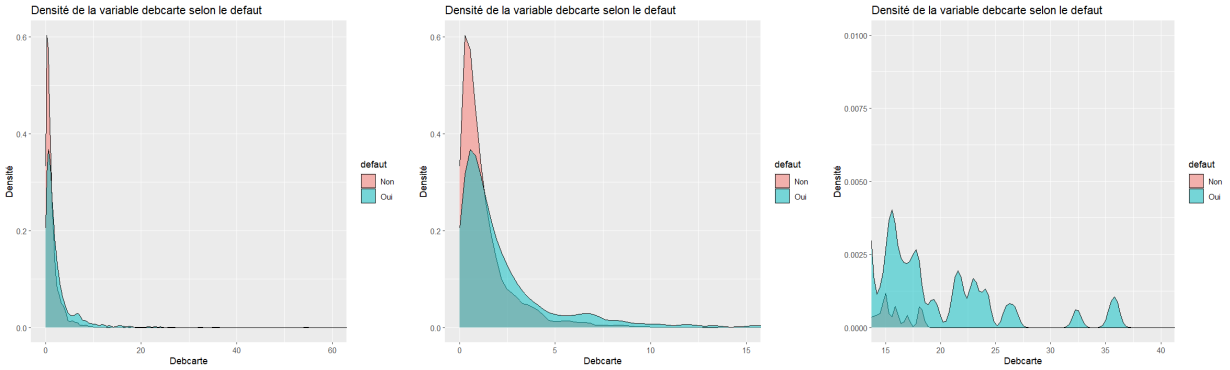


On remarque que pour des niveaux de dette faibles à modérés, la courbe des clients en défaut est plus haute, ce qui signifie qu'il y a davantage de clients en défaut dans ces plages de dette. Cela met en évidence le rôle des dettes dans les défauts de paiement. Ainsi, la variable **autres** pourrait être un facteur déterminant dans l'analyse du risque de défaut.



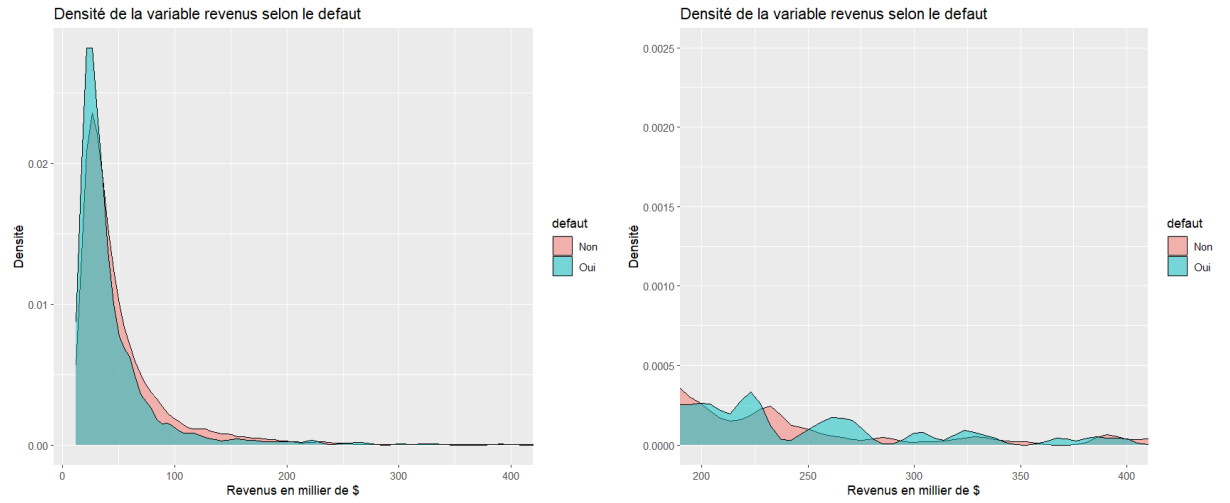
En revanche, aux très hauts niveaux de dette, les deux courbes sont très basses, indiquant que le nombre de clients avec une dette très élevée est faible dans les deux groupes. Cependant, nous remarquons également un nombre significatif de défauts de paiement, mis en évidence par l'aire sous la courbe bleue, ce qui renforce notre analyse sur l'importance de la variable **autres**.

Maintenant que nous avons détaillé notre analyse pour la variable **autres**, nous appliquons les mêmes manipulations à la variable **debcarte**.



On observe que pour des niveaux faibles à moyens de la variable **debcarte**, il y a une concentration plus élevée de clients en défaut. Pour des montants très élevés, les deux groupes sont rares, mais nous remarquons que pour un niveau élevé de **debcarte**, les défauts de paiement sont plus fréquents. Cela indique que la variable **debcarte** est particulièrement utile pour prédire le défaut lorsque les dépenses restent dans une plage faible à moyenne. Cependant, en ce qui concerne les plages plus élevées, nous devons approfondir notre analyse et mettre en relation la variable **debcarte** avec d'autres variables comme **autres**.

A présent, nous sommes en mesure de d'avoir une idée des variables déterminante. Cependant il nous reste une variable à analyser, **revenus**. Pour cela nous étudions de la même manière :

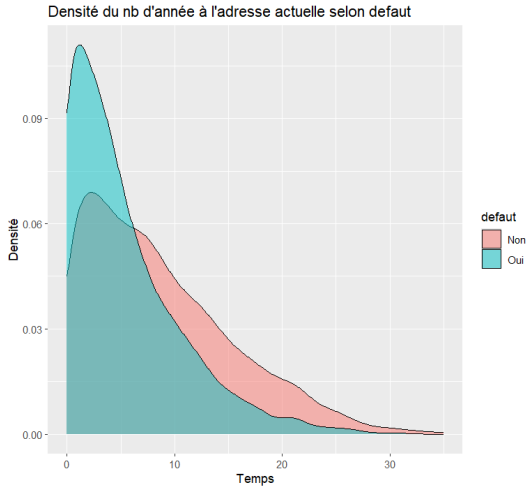


Les ménages aux faibles revenus sont plus susceptibles d'être en défaut de paiement, contrairement aux ménages aux revenus élevés.

3.4 Vérification

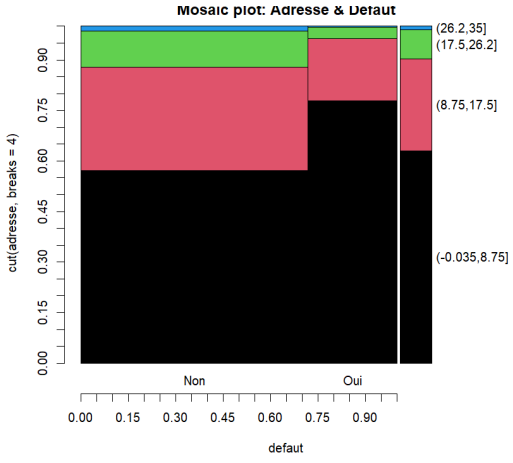
Nous vérifions avec d'autres techniques graphiques si nos analyses sont convaincantes. Dans le rapport, nous nous limiterons à la vérification de la variable **adresse**, mais nous pouvons également appliquer cette démarche aux autres variables. C'est précisément ce que nous avons fait afin de rédiger la partie **3.6) Déduction**.

3.4.1 Density Plot



Nous observons que les ménages habitant depuis peu d'années à leur adresse actuelle sont plus susceptibles d'être en défaut de paiement, comparés aux ménages résidant à leur adresse depuis plus de 10 ans.

3.4.2 Mosaïque



On observe que, dans les tranches de durée les plus faibles, la part des clients en défaut est plus élevée. En revanche, lorsque le nombre d'années passées à la même adresse augmente, la proportion de clients sans défaut devient majoritaire. Ainsi, ce graphique met en évidence qu'une durée de résidence plus longue à la même adresse est associée à un risque moindre de défaut de paiement.

3.4.3 Tableau de contingence

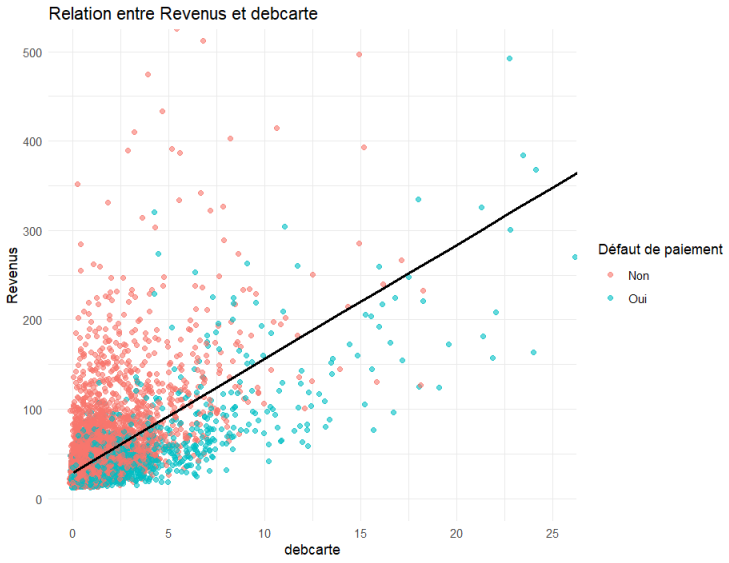
Adresse	Défaut Non	Défaut Oui
$[0, 7]$	37.080	20.918
$(7, 14]$	20.973	5.387
$(14, 21]$	9.846	1.523
$(21, 28]$	3.307	0.334
$(28, 35]$	0.594	0.037

Table 1: Probabilité de défauts de paiement selon le nombre d'années à l'adresse actuelle.

Ce tableau constitue un complément statistique au Density plot. Maintenant que nous avons en tête plusieurs caractéristiques de nos données, nous pouvons mettre en lien certaines variables afin de mieux comprendre les défauts de paiement.

3.5 Liens

À mesure que l'âge augmente, la durée d'emploi tend à être plus longue, ce qui est confirmé par la ligne de régression tracée dans le graphique. Cette tendance suggère une corrélation entre l'expérience professionnelle et la stabilité de l'emploi, renforçant l'idée que l'ancienneté peut être un facteur déterminant dans l'analyse du risque de défaut de paiement.Un autre lien notable est le suivant :



On constate que la tendance représentée par la ligne de régression est positive : plus le ratio **debcarte** augmente, plus les revenus ont tendance à être élevés. Cette corrélation suggère que les clients ayant un ratio **debcarte** plus important disposent généralement de revenus plus conséquents, ce qui pourrait être un indicateur pertinent dans l'analyse du risque de défaut de paiement.

3.6 Déduction

Grâce à nos graphiques et à nos analyses, nous pouvons déduire que les variables suivantes seront les plus utiles pour la prédiction de la classe **défaut** :

- **debcarte** : Cette variable reflète l'utilisation de la carte de crédit et le "mode de vie" de la personne. On remarque, grâce à l'analyse de densité, que les individus ayant un faible débit de carte de crédit sont les plus susceptibles d'être en défaut de paiement.
- **revenus** : Cette variable joue un rôle essentiel dans le remboursement des emprunts. De plus, les différents graphiques montrent que les personnes aux faibles revenus sont les plus susceptibles de contracter des défauts de paiement.
- **autres** : Les density plots ont mis en évidence une forte corrélation entre le défaut de paiement et les autres dettes.
- **debcred** : La majorité des cas de défaut de paiement se retrouvent dans les premières partitions de cette variable, ce qui suggère son importance dans l'analyse du risque de défaut.

4 Définition de la méthode d'évaluation des classifieurs

Après avoir exploré en détail les données, les tendances et les liens entre les variables clés, nous passons maintenant à la définition de la méthode d'évaluation des classifieurs. Cette étape va nous permettre de comprendre les outils qui nous serviront à comparer différents modèles. Nous utiliserons et étudierons des indicateurs de performance afin d'évaluer et de comparer nos différents modèles.

4.1 Taux et indices

4.1.1 Taux de succès

Le premier indicateur de performance que l'on peut qualifier de naturel est le calcul de la probabilité de l'événement suivant :

$$A = \{\text{La prédiction de notre modèle est juste}\}$$

Pour trouver $\mathbb{P}(A)$, nous effectuons la prédiction sur l'ensemble de test, ajoutons les résultats à un tableau, comptons le nombre de fois où nous obtenons la même valeur et calculons la moyenne arithmétique caractérisée par la formule suivante :

$$\bar{x} = \frac{1}{n} \sum_{i=1}^{\eta} x_i$$

où $x_i \in A, \forall i \in \{1, \ldots, \eta\}$, n correspond aux 6000 clients, et $\eta \in \{1, \ldots, 6000\}$ représente le nombre de bonnes prédictions. Nous traduisons cela par le code suivant :

```
# On predit sur notre ensemble de test
prediction <- predict(tree_rp_info, data_projet_ET, type = "class")

#On rajoute au tableau les resultats
data_projet_ET$Prediction <- prediction

nbr_succes <- sum(data_projet_ET$defaut == prediction)

#Notre moyenne arithmetique
taux_succes <- nbr_succes / nrow(data_projet_ET)

#On transforme notre taux en pourcentage, on arrondis au centieme, c'est P(A)
proba_A <-round(taux_succes * 100,2)
cat(proba_A)
```

Ainsi, nous obtenons une probabilité de succès et nous pouvons comparer les différents résultats afin de choisir le modèle avec le plus haut pourcentage de précision. Nous pouvons également calculer le taux d'échec et la probabilité de l'événement :

$$\bar{A} = \{\text{La prédiction de notre modèle est fausse}\}$$

4.1.2 Mesures de sélection d'attribut - Les différents indices

A) Gain d'informations

Avant d'introduire la notion de **gain d'information**, nous devons clarifier quelques concepts comme l'entropie et l'information requise. Dans notre cadre, l'entropie quantifie le niveau d'incertitude associé à la répartition des différentes classes ; elle mesure donc le désordre présent dans les données des clients. Elle est donnée par la formule suivante :

$$\text{Entropie}(E) = -\sum_{i=1}^{M} P(C_i) \log_2 P(C_i)$$

où M représente le nombre de classes et $P(C_i)$ la proportion d'exemples de E appartenant à la classe $C_i \in \{C_1, \ldots, C_M\}$.

L'information requise correspond à la quantité minimale nécessaire pour réduire l'incertitude dans les données. Elle est utilisée pour calculer le gain d'information, qui mesure l'efficacité d'un attribut à séparer ces données :

$$\text{Info}_V(EA) = \sum_{j=1}^{N} \frac{|EA_{V_j}|}{|EA|} \times \text{Entropie}(EA_{V_j})$$

où V est une variable prenant N valeurs, soit $V = \{V_1, \ldots, V_N\}$, et EA_{V_j} représente les exemples de EA pour lesquels la variable V prend la valeur V_j. Ainsi, si V segmente efficacement EA en sous-ensembles avec une entropie faible, la valeur de $\text{Info}_V(EA)$ sera faible.

Cela nous amène à définir le **gain d'information** :

$$\text{Gain}_V(EA) = \text{Entropie}(E) - \text{Info}_V(EA)$$

Un gain d'information élevé signifie que la variable (attribut) réduit fortement l'incertitude, ce qui entraîne une entropie moyenne faible des sous-ensembles. Les groupes deviennent homogènes et la classification est facilitée. À l'inverse, un gain faible indique que l'attribut ne diminue que peu l'entropie, laissant des sous-ensembles encore mélangés et peu efficaces pour distinguer les classes.

B) Indice de Gini

Passons maintenant à l'indice de Gini, qui mesure le degré de mélange des classes présentes. Un nœud pur, c'est-à-dire contenant principalement des exemples d'une seule classe, a une impureté faible, ce qui est essentiel pour une bonne performance du modèle. Cet indice se définit comme suit :

$$Gini_V(EA) = \sum_{j=1}^{N} \frac{|EA_{V_j}|}{|EA|} \times \left(1 - \sum_{i=1}^{M} P(C_i|EA_{V_j})^2\right)$$

Lorsque l'indice se situe près de zéro, cela signifie que le nœud est très homogène, tandis qu'une valeur élevée indique que le nœud présente une diversité marquée dans la répartition des classes.

4.2 Matrice de confusion

Intéressons-nous maintenant aux matrices de confusion, qui offrent une vue détaillée des prédictions correctes et des erreurs des classifieurs. Cet outil détaille le nombre de prédictions correctes et incorrectes pour chaque classe prédite et chaque classe réelle. Il quantifie ainsi l'importance des différents types de succès et d'erreurs. La matrice de confusion se présente sous la forme suivante :

Structure d'une matrice de confusion

	Prédite	
Réelle	**Positif**	**Négatif**
Positif	VP	FN
Négatif	FP	VN

Avec :

- **Vrais Positifs (VP)** : Exemples réels positifs correctement prédits comme positifs.
- **Faux Négatifs (FN)** : Exemples réels positifs incorrectement prédits comme négatifs.
- **Faux Positifs (FP)** : Exemples réels négatifs prédits à tort comme positifs.
- **Vrais Négatifs (VN)** : Exemples réels négatifs correctement prédits comme négatifs.

```r
# On predit sur notre ensemble de test
prediction <- predict(tree_rp_info, data_projet_ET, type = "class")

#On cree un tableau contenant nos informations
matrice_confusion <- table(data_projet_ET$defaut, prediction)

#On affiche le tableau
print(matrice_confusion)
```

4.3 Matrice de coût

Lorsqu'on construit la matrice de confusion de notre modèle, nous l'utilisons comme outil pour quantifier l'impact des erreurs en élaborant une matrice de coûts qui précise la pénalité associée à chaque mauvaise classification. Pour effectuer cette estimation, nous devons associer une valeur aux quatre catégories définies précédemment. Pour les vrais positifs, le coût de l'erreur est nul, tout comme pour les vrais négatifs. Étant donné que nous sommes dans un contexte de prédiction de défaut de paiement des clients, il est primordial de bien identifier les futurs clients qui ne paieront pas. Ainsi, le coût d'une erreur de prédiction pour un faux négatif (client prédit comme **défaut = Non** alors qu'il est en réalité en défaut de paiement) est fixé arbitrairement à 100. Enfin, pour les faux positifs, qui sont des clients catégorisés comme potentiellement en défaut de paiement alors qu'ils ne le sont pas réellement, le coût est moindre et fixé à 10. Résumons :

- **Coût VP** : Client réellement en défaut de paiement - Coût = 0
- **Coût FN** : Client prédit comme **défaut = Non** alors qu'il est en défaut de paiement - Coût = 100
- **Coût FP** : Client qui n'est pas en défaut de paiement mais qui est marqué comme en défaut de paiement - Coût = 10
- **Coût VN** : Client qui n'est pas en défaut de paiement - Coût = 0

Matrice de coût

	Prédite	
Réelle	**Positif**	**Négatif**
Positif	0	100
Négatif	10	0

Ainsi, en obtenant la matrice de confusion, notée $\mathbf{MC_{2x2}}$, nous pouvons multiplier le coefficient en $(1, 2)$ par 100 et le coefficient en $(2, 1)$ par 10 afin d'obtenir la matrice de coût de notre classifieur. Soit **card** le nombre d'éléments, nous pouvons le résumer de la manière suivante :

Matrice de coût d'un classifieur

	Prédite	
Réelle	**Positif**	**Négatif**
Positif	card(VP) * 0	card(FN) * 100
Négatif	card(FP) * 10	card(VN) * 0

De cette manière, nous pouvons comparer les coûts des différents classifieurs afin de limiter le risque de défaut de paiement. Cependant, il faut prêter attention aux coefficients. En effet, dans le code en R, la matrice est présentée sous la forme suivante :

Matrice de coût d'un classifieur

	Prédite	
Réelle	**Négatif**	**Positif**
Négatif	card(VN) * 0	card(FP) * 10
Positif	card(FN) * 100	card(VP) * 0

4.4 Les mesures

Ces matrices de confusion et de coût permettent de calculer des indicateurs clés comme le rappel, la spécificité, la précision et le taux de vrais négatifs. Ces mesures permettent de sélectionner le classifieur en fonction des besoins applicatifs. Par exemple, dans notre cas, il est préférable d'avoir des faux positifs plutôt que des faux négatifs, car ces derniers peuvent avoir des répercussions financières en cas de défaut de paiement. Introduisons ces mesures :

- **Sensibilité - Taux de vrais positifs (TVP)** : Indique la proportion d'exemples positifs correctement identifiés. Elle se mesure de la manière suivante :

$$\frac{VP}{VP + FN}$$

- **Spécificité** : Mesure la proportion d'exemples négatifs correctement mis de côté. Elle se mesure de la manière suivante :

$$\frac{VN}{VN + FP}$$

- **Précision** : Mesure la proportion de prédictions positives qui sont correctes. Elle se mesure de la manière suivante :

$$\frac{VP}{VP + FP}$$

- **Taux de vrais négatifs (TVN)** : Quantifie la qualité des prédictions négatives. Elle se mesure de la manière suivante :

$$\frac{VN}{VN + FN}$$

- **Taux de faux positifs** : Proportion des exemples négatifs qui ont été incorrectement classés comme positifs. Elle se mesure de la manière suivante :

$$\frac{FP}{FP + VN}$$

ou encore :

$$1 - \frac{VP}{VP + FN}$$

Dans notre contexte, il est crucial de détecter le maximum de clients susceptibles d'être en défaut de paiement. Ainsi, les mesures les plus importante à comparer entre les classifieurs sont le taux de faux négatifs et la sensibilité , car une sensibilité élevée minimise le nombre de clients en défaut de paiement et un faible taux de faux négatifs diminiue le coût.

4.5 Courbe ROC et AUC

Ces mesures sont un atout pour comparer les différents classifieurs. En effet, elles sont utilisées pour construire la courbe ROC.

4.5.1 Courbe ROC

La courbe ROC est un outil de visualisation graphique permettant d'évaluer et de comparer les performances des classifieurs dans la distinction de la classe positive. Elle trace, pour divers seuils de décision, le TVP en ordonnée et le TFP en abscisse. Elle se distingue par sa robustesse face aux déséquilibres de classes et son indépendance par rapport aux coûts, permettant ainsi une comparaison visuelle efficace des classifieurs.

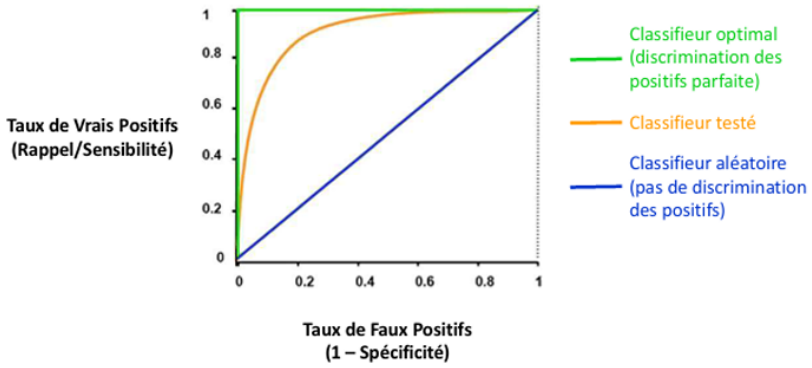


Figure 1: Schéma récapitulant le concept de courbe ROC
Source : PASQUIER Nicolas

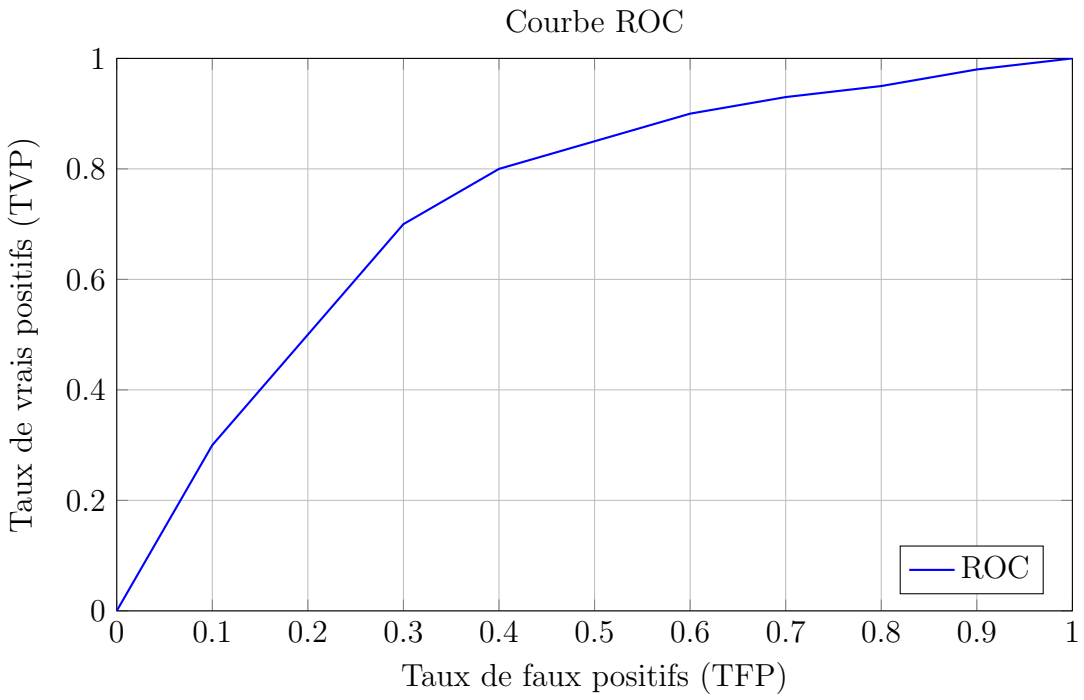


Figure 2: Exemple de courbe ROC avec TVP en ordonnée et TFP en abscisse

4.5.2 L'air sous la courbe - AUC

L'aire sous la courbe (AUC) synthétise la performance globale du classifieur. Une valeur proche de 1 indique une très bonne discrimination, tandis qu'une AUC de 0,5 correspond à un modèle équivalent à un tirage au sort.

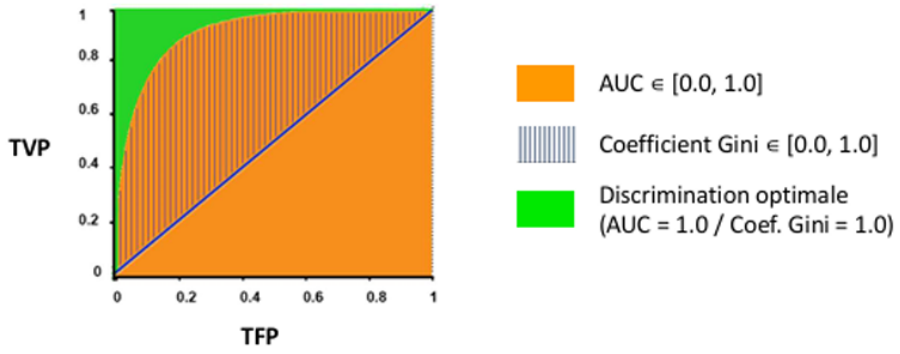


Figure 3: Schéma récapitulant le concept d'AUC
Source : PASQUIER Nicolas

L'avantage de ces données est d'avoir une mesure indépendante des coûts pour pouvoir mieux comparer les différents résultats.

En combinant toutes ces méthodes d'évaluation, nous pouvons nous assurer de choisir le classifieur le plus adapté à notre cas. Avant de présenter les différents modèles de classifieurs utilisés, nous devons nous assurer de bien définir nos données d'apprentissage et de test.

5 Définition des données d'apprentissage et de test

5.1 Avec des valeurs inconnus

Nous pouvons créer plusieurs échantillons. Nous allons sélectionner aléatoirement deux tiers des données du fichier `Projet Data.csv` pour l'apprentissage de nos classifieurs, et un tiers des données pour effectuer les différents tests et comparaisons de résultats.

5.2 Traitement des valeurs inconnues

Cependant, parmi ces observations, certaines lignes contiennent la valeur NA (pour les colonnes **adresse** et **âge**), ce qui pose problème pour certains classifieurs comme les modèles utilisant **Random Forests** et **K-nearest neighbors**. Nous devons, dans un premier temps, quantifier le nombre de valeurs NA dans notre base de données afin de déterminer si la suppression de ces lignes aura un impact significatif sur la prédiction. Pour ce faire, nous exécutons la commande suivante dans notre console :

```
print(sapply(data_projet, function(x) sum(is.na(x))))
```

Nous remarquons que nous obtenons 530 valeurs manquantes (NA) pour la variable **âge** et 617 valeurs manquantes pour la variable **adresse**. En pourcentage, cela représente 9% de données manquantes pour **âge** et 10% pour **adresse**. Ainsi, si nous supprimons un pourcentage modéré de données, nous pouvons envisager de remplacer les valeurs NA par la moyenne arithmétique correspondante aux colonnes concernées ou de réduire la taille de notre échantillon.

5.2.1 La moyenne

Nous devons remplacer les valeurs NA dans notre base de données par une valeur appropriée. Cette valeur n'est pas choisie au hasard : elle correspond à la moyenne arithmétique des valeurs des colonnes **âge** et **adresse**. En calculant la moyenne de chacune de ces deux colonnes, nous pouvons remplacer les valeurs manquantes.

Cependant, nous devons nous assurer que cette moyenne n'est pas excessive. Pour cela, nous nous appuyons sur les graphiques présentés ultérieurement. Nous obtenons une moyenne de **35 ans** pour l'âge et une moyenne de **7.7** pour la variable adresse. En vérifiant nos graphiques afin de mieux comprendre la distribution des données, nous constatons, grâce à l'histogramme représentant l'effectif des défauts de paiement selon l'âge, qu'il y a une forte concentration entre **20 et 40 ans**, avec un pic autour de **35 ans**, ce qui correspond à notre moyenne. Nous pouvons donc l'utiliser.

Concernant la variable **adresse**, l'analyse du **Box Plot** montre que la moyenne du nombre d'années à l'adresse actuelle est d'environ **4 ans** pour les clients en défaut de paiement et de **8 ans** pour ceux qui ne le sont pas. Notre moyenne de **7.7** correspond donc bien aux données et n'est pas excessive, nous pouvons l'utiliser.

Ainsi, nous pouvons constituer un **échantillon d'apprentissage** de **4000 observations** et un **échantillon de test** de **2000 observations**, en remplaçant les valeurs NA par la moyenne correspondante à chaque colonne.

```r
data_projet_moy <- data_projet

#On remplace NA par la moyenne en question
data_projet_moy$age[is.na(data_projet_moy$age)] <- mean(data_projet_moy$age, na.rm = TRUE)
data_projet_moy$adresse[is.na(data_projet_moy$adresse)] <- mean(data_projet_moy$adresse, na.rm = TRUE)
```

Cependant nous devons pas oublier d'appliquer le pré-traitement (suppression de la variable catégorie, defaut devient un facteur, l'aléatoire...) de nos données aux nouvelles partitions.

5.2.2 Réduction de la base de données

Néanmoins, nous devons garder à l'esprit que remplacer les valeurs manquantes par la moyenne peut fausser notre prédiction. Pour éviter ce biais, nous pouvons adopter une approche différente en restreignant notre base de données.

Dans un premier temps, nous appliquons le prétraitement, puis nous supprimons toutes les lignes contenant des valeurs NA. Cela nous donne une base de données de **4887 observations**, après suppression des **530 valeurs NA** pour la variable **âge** et des **617 valeurs NA** pour la variable **adresse**. Ainsi, nous obtenons une base de données sans valeurs manquantes et sans biais. Enfin, nous la partitionnons en **deux tiers** pour l'apprentissage et **un tiers** pour le test.

```r
#Melange aleatoire des donnees

set.seed(123)

data_projet_pur <- data_projet_pur[sample(nrow(data_projet_pur)), ]

#3258 observations pour l'apprentissage et 1629 pour le test
data_projet_EA_pur <- data_projet_pur[1:3258, ]
data_projet_ET_pur <- data_projet_pur[3259:4887, ]
```

L'inconvénient de cette méthode est qu'elle réduit la taille de notre ensemble d'apprentissage, ce qui peut avoir des répercussions sur la précision de nos modèles de prédiction.

6 Construction et évaluation des classifieurs

Maintenant que nous savons ce que nous souhaitons évaluer, de quelle manière et avec quel échantillon, nous devons créer nos classifieurs. Au cours de notre recherche, nous avons construit plusieurs classifieurs, exploré divers paramétrages et affiché différentes illustrations. Examinons plus en détail chaque modèle testé et étudié.

6.1 Rpart

Dans un premier temps, nous avons utilisé la fonction `rpart`, qui implémente l'algorithme CART. Ce dernier partitionne de manière récursive l'ensemble des données en sous-groupes homogènes, en choisissant à chaque nœud l'attribut qui maximise la réduction d'impureté. Cet attribut peut être sélectionné grâce à la fonction `rpart` et, selon notre cours, nous disposons de deux critères pour le choix de cet attribut : le **gain d'information** et l'**indice de Gini**, présentés précédemment.

De plus, nous disposons d'un autre paramètre, le **minbucket**, qui correspond à l'effectif minimal imposé dans chaque nœud. En le modifiant, nous pouvons obtenir des prédictions différentes et identifier des variables déterminantes distinctes.

A) Configurations

Nous avons expérimenté plusieurs configurations du modèle `rpart` afin d'optimiser la prédiction des défauts de paiement. Dans un premier temps, nous avons fixé le paramètre **minbucket** à 5 et choisi comme critère de sélection d'attribut le **gain d'information** pour construire notre premier arbre de décision. Ensuite, nous avons fait varier le paramètre **minbucket** à 7, 10, 15 et enfin à 20. Dans un second temps, nous avons procédé de la même manière en utilisant l'**indice de Gini** comme critère de sélection d'attribut.

B) Évaluations

Tout d'abord, pour chaque configuration, nous avons généré un arbre de décision que nous avons visualisé avec la fonction `prp`. Ensuite, nous avons appliqué différentes méthodes d'évaluation (matrice de confusion, matrice de coût, courbe ROC...) sur divers échantillons afin d'obtenir une analyse complète.

RPART w/ NA	Taux de succès	FN	FP	Coût Total	Coût Moyen	AUC	Taux Vrai Né
Split = Information Minbucket = 5	76%	384	100	39400	20	0.69	0.77
Split = Information Minbucket = 10	76%	384	100	39400	20	0.69	0.77
Split = Information Minbucket = 15	7%	384	100	39400	20	0.69	0.77
Split =Gini Minbucket = 5	76%	403	86	41160	21	0.69	0.77
Split =Gini Minbucket = 10	76%	403	86	41160	21	0.69	0.77
Split =Gini Minbucket = 15	76%	403	86	41160	21	0.69	0.77

Figure 4: Résultats du modèle RPart sur l'échantillon avec NA

De plus grâce à la prochaine ligne de commande :

```
print(tree_rp_info$variable.importance)

print(tree_rp_gini$variable.importance)
```

Nous obtenons la liste des variables les plus déterminantes pour la prédiction. Nous avons remarqué que pour le modèle RPart utilisant l'indice de Gini les variables les plus importantes sont les suivantes : debcred - autres - debcarte. En revanche pour le modèle utilisant le gain d'information nous remarquons qu'à la place de debcarte nous avons emploi. Cela rejoint la conclusion de notre analyse exploratoire des données.

6.2 C50

Nous avons testé une deuxième méthode de classification basée sur l'algorithme **C5.0**. Contrairement à `rpart`, qui met en œuvre l'algorithme **CART**, **C5.0** intègre des mécanismes avancés tels que l'élagage global (**Global Pruning**) afin de réduire le sur-apprentissage et d'améliorer la robustesse du modèle. Cette approche permet également de mesurer de manière fine l'importance des variables pour la prédiction d'un défaut de paiement.

A) Configurations

Nous avons expérimenté plusieurs configurations du modèle **C5.0** en jouant particulièrement sur deux paramètres clés : **mincases** et **GlobalPruning**. Le premier paramètre fixe le nombre minimal d'observations dans un nœud pour qu'il puisse être scindé, comme pour `rpart`. En revanche, le deuxième paramètre permet de nettoyer l'arbre en supprimant les branches non significatives après la croissance initiale, si nous activons le **Global Pruning**. Nous avons jugé intéressant de jouer avec ces deux paramètres afin d'évaluer ces modèles dans notre cas.

B) Évaluations

Pour évaluer les modèles **C5.0**, nous avons d'abord créé des tableaux de contingence afin d'obtenir une idée de la répartition des classes prédites et du taux de succès. Nous avons également extrait les variables les plus déterminantes dans la prédiction du défaut de paiement. Ensuite, nous avons effectué une analyse complète, comme pour la méthode `rpart`.

C5.0 w/ NA	Taux de succès	FN	FP	Coût Total	Coût Moyen	AUC	Taux Vrai Né
GlobalPuring = Avec Minbucket = 5	76%	364	103	37430	19	0.69	0.78
GlobalPuring = Avec Minbucket = 10	77%	358	114	36940	18	0.69	0.77
GlobalPuring = Avec Minbucket = 15	76%	380	100	39000	20	0.68	0.78
GlobalPuring = Sans Minbucket = 5	76%	384	87	39270	20	0.69	0.78
GlobalPuring = Sans Minbucket = 10	77%	368	97	37770	19	0.7	0.78
GlobalPuring = Sans Minbucket = 15	76%	373	102	38320	19	0.69	0.77

Figure 5: Résultats du modèle C5.0 sur l'échantillon avec NA

En ce qui concerne les variables déterminantes donnée par le modèle, nous remarquons qu'elles sont identique pour les différents paramétrages. Voici les variables déterminantes pour le modèle C5.0 : debcred - emploi- debcarte - adresse

6.3 TREE

Nous n'allons pas nous arrêter en si bon chemin nous avons implémenté la méthode **TREE**. Cette technique, basée sur l'algorithme `tree`, permet de construire des arbres de décision en utilisant des critères de division différents de ceux utilisés par les méthodes **C5.0** et **Rpart**. Elle offre ainsi une vision complémentaire pour extraire et analyser les variables déterminantes.

A) Configurations

Pour expérimenter la méthode **TREE**, nous avons construit deux modèles en faisant varier le paramètre **split** ainsi que le seuil minimal d'observations (**mincut**) requis pour effectuer une division. Dans un premier temps, comme pour les modèles précédents, nous avons fixé le paramètre de scission. Ici, nous avons eu le choix entre l'**indice de Gini** et la **déviance**, qui évalue l'efficacité d'un attribut à rendre un nœud plus homogène en comparant la distribution de la variable cible avant et après la division. Dans un second temps, nous avons fixé notre seuil minimal d'observations.

B) Évaluations

Comme pour les autres classifieurs, nous avons réalisé des prédictions sur l'ensemble de test contenant des observations **NA** pour chacun des modèles **TREE**. Nous avons ensuite établi des tableaux de contingence afin de comparer la répartition des classes prédites aux classes réelles et calculé le taux de succès. Pour solidifier notre analyse, nous avons poursuivi nos études en examinant la **matrice de coût** et la **courbe ROC**, puis nous avons résumé l'ensemble des résultats dans un tableau.

TREE w/ NA	Taux de succès	FN	FP	Coût Total	Coût Moyen	AUC	Taux Vrai Né
Split = deviance Mincut = 5	76%	300	175	31750	16	0.77	0.81
Split = deviance Mincut = 10	76%	358	114	31750	16	0.77	0.81
Split = deviance Mincut = 15	76%	358	114	31750	16	0.77	0.81
Split = Gini Mincut = 5	73%	286	257	31170	16	0.73	0.8
Split = Gini Mincut = 10	74%	292	238	31580	16	0.76	0.8
Split = Gini Mincut = 15	75%	292	212	31320	16	0.77	0.81

Figure 6: Résultats du modèle TREE sur l'échantillon avec NA

Nous remarquons grâce aux indicateurs de taux de succès, AUC, et coût moyen et FN, que TREE prédit mieux les défauts de paiements que les méthodes C5.0 et RPart. De ce fait, pour pouvoir comparer les différentes méthodes sur le même échantillon, nous allons implémenté la méthode TREE sur l'échantillon réduit **data projet ET pur**.

TREE réduit	Taux de succès	FN	FP	Coût Total	Coût Moyen	AUC	Taux VN
Split = deviance Mincut = 5	76%	258	137	27170	17	0.76	0.8
Split = deviance Mincut = 10	76%	233	132	27170	17	0.76	0.81
Split = deviance Mincut = 15	76%	211	151	27170	17	0.76	0.81
Split = Gini Mincut = 5	78%	254	106	26460	16	0.84	0.8

Figure 7: Résultats du modèle RF sur l'échantillon réduit

6.4 Random Forest

Passons maintenant à un modèle plus complet : la méthode **Random Forest**. La méthode **TREE** nous offre une vue simple des décisions et des variables déterminantes pour la prédiction, tandis que **Random Forest** combine plusieurs arbres issus d'échantillons aléatoires et de variables afin d'améliorer la robustesse et la précision des prédictions.

A) Configurations

Pour configurer notre modèle **Random Forest**, nous avons d'abord tenté d'appliquer le classifieur à l'échantillon comportant des données manquantes (**NA**), mais nous avons rencontré plusieurs erreurs. En consultant la documentation, nous avons appris que cette méthode doit être utilisée sur une base de données ne contenant aucune valeur manquante. Ainsi, nous avons appliqué notre classifieur à deux échantillons : l'un où les valeurs **NA** ont été remplacées par la moyenne correspondante et l'autre, de taille réduite, ne comportant aucune valeur **NA**. Ensuite, nous avons ajusté deux paramètres essentiels, **ntree** qui détermine le nombre total d'arbres construits dans la forêt et **mtry** qui spécifie le nombre de variables sélectionnées aléatoirement à chaque division pour construire l'arbre. En augmentant le paramètre **ntree**, par exemple de 300 à 500, nous améliorons la robustesse de notre prédiction. En augmentant **mtry** de 3 à 5, nous améliorons la séparation entre les classes.

B) Évaluations

En sélectionnant attentivement les informations utiles ainsi que les échantillons d'apprentissage et de test, nous avons résumé les différents résultats dans le tableau suivant :

RF réduit	Taux de succès	FN	FP	Coût Total	Coût Moyen	AUC	Taux Vrai Né
ntree = 300 Mtry = 3	77%	264	117	27570	17	0.82	0.80
ntree = 300 Mtry = 5	77%	263	114	27440	17	0.81	0.8
ntree = 500 Mtry = 3	76%	268	118	27980	17	0.82	0.79
ntree = 500 Mtry = 5	77%	263	117	27470	17	0.81	0.8

Figure 8: Résultats du modèle RF sur l'échantillon réduit

RF mean	Taux de succès	FN	FP	Coût Total	Coût Moyen	AUC	Taux Vrai Né
ntree = 300 Mtry = 3	79%	282	136	29560	15	0.83	0.82
ntree = 300 Mtry = 5	79%	280	140	29400	15	0.83	0.83
ntree = 500 Mtry = 3	79%	279	138	29280	15	0.83	0.83
ntree = 500 Mtry = 5	79%	283	138	29680	15	0.83	0.82

Figure 9: Résultats du modèle RF sur l'échantillon avec les moyennes

Nous remarquons que la méthode Random Forest sur l'échantillon avec les moyenne à la place de NA obtient un meilleur taux de succès, un coût inférieur et une AUC supérieure aux autres méthodes. En revanche comme nous avons appliqué notre algorithme sur des données modifiés, cela ne rélfète pas la réalité, un biais existe ce qui fausse notre prédiction.

6.5 K-Nearest Neighbors

Après avoir implémenté la méthode **Random Forest**, nous nous penchons désormais sur la méthode **K-Nearest Neighbors (KNN)**. Cette dernière prédit la classe d'un client en s'appuyant sur ses voisins les plus proches en termes de données.

A) Configurations

Dans notre implémentation, le paramètre **k** représente le nombre de clients voisins à considérer dans l'analyse, tandis que le paramètre **distance** correspond à la distance euclidienne utilisée pour

mesurer la proximité entre les observations. Nous avons fait varier ces paramètres afin d'observer leurs impacts respectifs sur la prédiction. Cependant, nous ne pouvons pas appliquer cette méthode à notre échantillon contenant des valeurs manquantes, car cet algorithme nécessite de connaître toutes les valeurs des différentes variables.

B) Évaluations

Ainsi, nous avons évalué le modèle sur l'échantillon où les valeurs **NA** ont été remplacées par la moyenne, ainsi que sur l'échantillon réduit.

KNN mean	Taux de succès	FN	FP	Coût Total	Coût Moyen	AUC	Taux Vrai Né
k = 10 Distance = 1	78%	267	173	28430	15	0.79	0.83
k = 10 Distance = 2	78%	267	140	28430	15	0.83	0.83
k = 20 Distance = 1	80%	284	124	29640	15	0.81	0.83
k = 20 Distance = 2	79%	286	135	29950	15	0.80	0.82

Figure 10: Résultats du modèle KNN sur l'échantillon avec les moyennes

KKN réduit	Taux de Succès	FN	FP	Coût Total	Coût Moyen	AUC	Taux VN
K = 10 Distance = 1	75%	270	131	29310	17	0.80	0.79
K = 10 Distance = 2	76%	258	126	27060	17	0.78	0.8
K = 20 Distance = 1	77%	275	102	28520	18	0.80	0.79
K = 20 Distance = 2	77%	273	102	28320	17	0.80	0.79

Figure 11: Résultats du modèle KNN sur l'échantillon réduit

6.6 Support Vector Machine

Après avoir exploré les approches basées sur des arbres, nous abordons l'implémentation du **SVM** (Support Vector Machine). Cette méthode sépare les différentes classes en créant un hyperplan optimal et en maximisant la marge entre elles.

A) Configurations

Dans un premier temps, nous effectuons la séparation avec une frontière linéaire. Ensuite, nous modifions le paramétrage du **kernel** en choisissant différentes séparations, telles que la polynomiale ou la sigmoïde. Chacune de ces séparations transforme l'espace des variables de manière différente afin de capturer certaines relations omises par les séparations linéaires. Comme pour les deux méthodes précédentes, nous devons nous assurer que nous travaillons sur un échantillon dont les valeurs sont connues. Ainsi, nous implémentons cette méthode sur les échantillons où les valeurs **NA** ont été remplacées par la moyenne, ainsi que sur l'échantillon réduit.

B) Évaluations

SVM mean	Taux de succès	FN	FP	Coût Total	Coût Moyen	AUC	Taux VN
Probability = True Kernel = Linear	63%	493	255	51850	26	0.15	0.7
Probability = True Kernel = Polynomial	63%	534	124	54640	27	0.18	0.71
Probability = True Kernel = Radial	63%	501	244	52540	26	0.18	0.7
Probability = True Kernel = Sigmoid	73%	267	249	29190	15	0.26	0.82

Figure 12: Résultats du modèle SVM sur l'échantillon avec les moyennes

SVM réduit	Taux de succès	FN	FP	Coût Total	Coût Moyen	AUC	Taux VN
Probability = True Kernel = Linear	78%	269	87	27770	17	0.84	0.8
Probability = True Kernel = Polynomial	76%	357	26	35960	22	0.82	0.71
Probability = True Kernel = Radial	78%	294	72	30120	18	0.83	0.79
Probability = True Kernel = Sigmoid	73%	264	170	28100	17	0.75	0.79

Figure 13: Résultats du modèle SVM sur l'échantillon réduit

6.7 Naïve Bayes

Passons maitenant à une méthode probabiliste avec la classification Naive Bayes. Cette méthode utilise l'indépendance des variables qui permet d'estimer les probabilités de classe à partir des différentres fréquences.

A) Configurations

Dans notre implémentation, nous configurons le modèle grâce aux paramètres **Laplace**, qui permet un lissage en ajoutant une constante à chaque fréquence afin d'éviter une probabilité nulle, et **userkernel**, qui permet d'estimer la densité des variables. Nous avons appliqué cette méthode à l'échantillon comportant des valeurs inconnues et avons fait varier ces deux paramètres afin d'évaluer leur impact sur la prédiction des défauts de paiement des clients.

B) Évaluations

NB w/ NA	Taux de succès	FN	FP	Coût Total	Coût Moyen	AUC	Taux VN
Laplace = 0 Usekernel = TRUE	77%	270	183	28830	14	0.81	0.82
Laplace = 20 Usekernel = TRUE	78%	268	182	28620	14	0.81	0.82
Laplace = 0 Usekernel = FALSE	77%	340	123	35230	18	0.81	0.79
Laplace = 20 Usekernel = FALSE	77%	339	124	35140	18	0.81	0.79

Figure 14: Résultats du modèle NB sur l'échantillon avec NA

NB réduit	Taux de succès	FN	FP	Coût Total	Coût Moyen	AUC	Taux VN
Laplace = 0 Usekernel = TRUE	78%	233	127	24570	15	0.82	0.81
Laplace = 20 Usekernel = TRUE	78%	235	127	24770	15	0.82	0.81
Laplace = 0 Usekernel = FALSE	74%	381	45	38550	24	0.80	0.74
Laplace = 20 Usekernel = FALSE	74%	267	249	38540	24	0.80	0.82

Figure 15: Résultats du modèle NB sur l'échantillon réduit

6.8 Réseau de neurones

Enfin, nous implémentons notre dernier classifieur, les réseaux de neurones.

A) Configurations

Dans notre configuration, nous utilisons plusieurs paramètres. Le premier, **size**, détermine le nombre de neurones dans la couche cachée, ce qui aide le modèle à comprendre les liens et les motifs entre

les variables. Le deuxième, **decay**, permet au modèle d'éviter le sur-apprentissage ; il correspond au taux d'apprentissage. Enfin, le dernier paramètre, **maxit**, représente le nombre maximal d'itérations nécessaires pour assurer une convergence satisfaisante. Dans nos tests, nous avons remarqué que **100 itérations** étaient insuffisantes. Par conséquent, nous avons décidé de travailler avec **300 itérations**, comme recommandé dans le cours. Enfin, nous avons également choisi d'implémenter la méthode sur l'échantillon réduit afin de comparer les résultats des méthodes précédentes.

B) Évaluations

RN réduit	Taux de succès	FN	FP	Coût Total	Coût Moyen	AUC	Taux VN
Size = 25 Decay = 0,01 Maxit = 300	77%	245	122	25720	16	0.83	0.81
Size = 50 Decay = 0,01 Maxit = 300	78%	233	132	24620	15	0.82	0.81
Size = 25 Decay = 0,001 Maxit = 300	78%	211	151	22610	14	0.82	0.83
Size = 50 Decay = 0,001 Maxit = 300	78%	254	106	26460	16	0.84	0.8

Figure 16: Résultats du modèle RN sur l'échantillon réduit

7 Choix du classifieur le plus performant

Maintenant que nous disposons de toutes les données nécessaires, nous pouvons nommer le classifieur le plus performant pour prédire les défauts de paiement des clients. Comparons les différents résultats.

7.1 Comparaison générale

Dans un premier temps, nous devons comparer les classifieurs. Cependant, comme nous avons travaillé sur plusieurs échantillons, les résultats sont différents. Les prédictions sur l'ensemble **data projet ET moy** (i.e. échantillon avec les moyennes) sont bien plus précises que sur les autres échantillons, car nous disposons de toutes les valeurs. Toutefois, comme nous avons utilisé la moyenne, ces prédictions sont biaisées et ne représentent pas la véritable performance des classifieurs. De ce fait, nous allons nous attarder sur les prédictions des échantillons **data projet ET** et **data projet ET pur** (i.e. échantillon réduit). Nous remarquons que plusieurs classifieurs se démarquent malgré le manque de valeurs et un échantillon avec un nombre d'observations réduit :

- **TREE** - avec : **split = gini** et **mincut = 5**
- **Random Forest** - avec : **ntree = 300** et **mtry = 5**
- **K-Nearest Neighbors** - avec : **k = 20** et **distance = 2**

- **Support Vector Machine** - avec : **probability = True** et **kernel = linear**
- **Naïve Bayes** - avec : **laplace = 20** et **usekernel = True**
- **Réseau de neurones** - avec : **size = 25**, **decay = 0.001 maxit = 300**

Nous pouvons résumer leurs performances dans le tableau suivant, et nous allons comparer les différents résultats.

	Taux de succès	FN	FP	Coût Total	Coût Moyen	AUC	Taux VN	Echantillon
TREE	78%	254	106	26460	16	0.84	0.8	réduit
Random Forest	77%	263	117	27470	17	0.81	0.8	réduit
K- Nearest Neighbors	76%	258	126	27060	17	0.78	0.8	réduit
Support Vector Machine	78%	269	87	27770	17	0.84	0.8	réduit
Naïve Bayes	78%	268	182	28620	14	0.81	0.82	NA
Naïve Bayes	78%	233	127	24570	15	0.82	0.81	réduit
Réseau de neurones	78%	211	151	22610	14	0.82	0.83	réduit

Figure 17: Résultats des meilleurs classifieurs sur plusieurs échantillons

Comme nous souhaitons minimiser le risque qu'un client choisi soit en défaut de paiement, nous devons nous assurer que notre modèle possède un coût moindre, un taux de succès de prédiction élevé et des variables déterminantes pertinentes données par l'AUC.

Nous devons nous attarder sur la mesure **FN**, qui représente les **faux négatifs**, c'est-à-dire les fausses prédictions de clients en non-défaut de paiement alors qu'ils sont en réalité en défaut de paiement. Si cette valeur est élevée dans les résultats des différents classifieurs, c'est un mauvais signe et signifie que ce classifieur n'est pas adapté à notre cas.

En analysant le tableau, nous remarquons que la méthode **Naïve Bayes** possède le coût total le plus faible : **24 570** sur l'échantillon **data projet ET pur** et **28 620** sur l'échantillon **data projet ET**. Même si cette valeur peut sembler élevée, il ne faut pas oublier qu'elle est obtenue sur **2 000 observations** contre **1 628** pour l'échantillon réduit. Ainsi, en comparant les coûts totaux des autres classifieurs, le coût total de **Naïve Bayes** est parmi les meilleurs, tout comme son coût moyen.

Un autre candidat est le classifieur reposant sur les **réseaux de neurones**. Ses performances et prédictions sont remarquables, mais nous avons travaillé sur un échantillon réduit. En effet, en implémentant cette méthode sur l'échantillon comportant des valeurs **NA**, nous n'obtenons pas de résultats, car elle ne peut pas apprendre à partir de données incomplètes et ne peut pas prédire sur des observations avec des valeurs manquantes. Nous observons le même phénomène pour les autres méthodes.

Ainsi, en comparant les modèles, les **AUC**, les coûts moyens ainsi que le taux de succès, nous pouvons conclure que le meilleur classifieur est **Naïve Bayes**.

7.2 Structure

Le modèle **Naïve Bayes** est un classifieur probabiliste qui repose sur l'indépendance des variables (ici **âge**, **éducation**, etc.). En cours de théorie des probabilités, nous avons vu que, étant donné un événement $A \in \mathscr{A}$ de probabilité positive, on appelle *probabilité conditionnelle sachant* A la mesure de probabilité sur $(\Omega, \mathscr{A})$, notée $\mathbb{P}(\cdot \mid A)$, et définie par

$$\mathbb{P}(B \mid A) = \frac{\mathbb{P}(A \cap B)}{\mathbb{P}(A)}.$$

Grâce à ce concept, nous obtenons des estimations efficaces et rapides en termes de calcul computationnel. De plus, contrairement aux approches basées sur des arbres ou des réseaux de neurones, notre méthode génère une structure contenant des formules et des calculs de probabilités pour chaque attribut. Ainsi, elle offre une légèreté computationnelle et une robustesse face à un grand nombre de variables comportant des valeurs manquantes.

En ce qui concerne le paramétrage utilisé, nous avons configuré notre modèle de la manière suivante

```
nb <- naive_bayes(defaut~., data_projet_EA, laplace = 20, usekernel = TRUE)
nb_class <- predict(nb, data_projet_ET, type="class")
```

Le paramètre **laplace = 20** introduit un lissage des fréquences afin d'éviter les problèmes de probabilités nulles. De plus ce lissage permet d'éviter le biais dû aux observations contenant un faible nombre de valeurs ou valeurs manquantes. Enfin, le paramètre **usekernel = TRUE** permet d'ajuster les distributions des variables continues.

7.3 Performances

	Taux de succès	FN	FP	Coût Total	Coût Moyen	AUC	Taux VN	Rappel	Spécificité	Echantillon
Naïve Bayes	78%	268	182	28620	14	0.81	0.82	0.54	0.87	NA

Figure 18: Performances du modèle Naïve Bayes

En comparant les différents résultats, le taux de succès se situe en moyenne autour de **78 %**, ce qui rejoint les performances des autres classifieurs. Cependant, ce taux est obtenu tout en maintenant un **coût moyen** et un **coût total** faibles, ce qui est primordial.

De plus, en examinant plus en détail les coûts, nous remarquons que notre modèle minimise le nombre de **faux négatifs** (i.e. prédire à tort qu'un client ne fera pas défaut alors qu'il en fera effectivement défaut).

En nous attardant sur l'**aire sous la courbe ROC**, qui permet de discriminer les variables et d'identifier les plus pertinentes, nous constatons que **AUC = 0.81**, ce qui est une valeur acceptable et satisfaisante, comme expliqué précédemment.

Enfin, de manière globale, bien que les performances des autres classifieurs surpassent légèrement celles du nôtre, il faut garder à l'esprit que nos prédictions ont été réalisées avec des valeurs manquantes, sans tri ni sélection des observations, ce qui introduit un biais. Cependant, grâce aux paramétrages de notre méthode **Naïve Bayes**, nous parvenons à fournir une prédiction globale fiable, avec un bon compromis entre toutes les mesures d'évaluation.

8 Résultats de l'application du classifieur

8.1 Pré-traitement des données

De la même manière, on supprime la variable **categorie**, on rend la variable **education** en facteur, et on remplace les valeurs absurdes par NA.

```
#Charger le fichier
new_data <- read.csv("Projet Data New.csv", header = TRUE, sep = ",")

#Nettoyer la variable categorie
new_data$categorie <- NULL

#Convertion en facteur
new_data$education <- factor(new_data$education,
                             levels = c("Niveau bac", "Bac+2", "Bac+3", "Bac+4",
    "Bac+5 et plus"),
                             ordered = TRUE)

#Remplacement des valeurs absurdes
new_data$age[new_data$age == 999] <- NA
new_data$adresse[new_data$adresse == 999] <- NA

#Verification
View(new_data)
```

8.2 Prédiction

Nous pouvons appliquer le classifieur à notre prédiction en supposant que le modèle a déjà été entraîné de la manière suivante :

```
#Apprentissage
nb <- naive_bayes(defaut~., data_projet_EA, laplace = 20, usekernel = TRUE)

#Prediction
nb_class_new_data <- predict(nb, new_data, type = "class")
```

8.3 Prédiction des probabilités

Nous devons calculer la probabilité associée à la prédiction de la classe. Pour ce faire nous allons appliquer notre méthode de la manière suivante :

```
#Prediction proba
nb_proba_new_data <- predict(nb, new_data, type = "prob")
```

Maintenant nous devons récupérer les valeurs pour pouvoir les mettre dans notre fichier csv.

```r
#Recuperer la classe predite pour la ligne i, elle contient la classe predite pour l'observation i

predicte_prob <- sapply(1:nrow(nb_prob_new_data), function(i) {

#Mais nous devons convetir cette valeur en chaine de caractere pour pouvoir l'ecrire dans notre fichier donc :

  class_i <- as.character(nb_class_new_data[i])


  #Comme nb_prob_new_data est une matrice ou chaque ligne correspond a une observation et chaque colonne a une classe, nous voulons acceder a la probabilite dans la ligne i et au client

  prob_i <- nb_prob_new_data[i, class_i]

  #On retourne la probabilite de l'indice i

  return(prob_i)
})
```

Ensuite nous devons créer le tableau final des résultats :

```r
resultat <- data.frame(
    client = new_data$client, #Le numero de colonne
    prediction = nb_class_new_data, #Notre prediction avec le classifieur NB
    probabilite = round(predicte_proba,2)  #la proba associe a la prediction
)
```

8.4 Les résultats

Prédiction	Min	Max	Mean
Non	0.5	1	0.831
Oui	0.5	1	0.759

Figure 19: Résumé des probabilités par classe

Notre classifieur a prédit que parmi les **500 clients**, **311** ne seront pas en défaut de paiement, contre **189** clients susceptibles d'être en défaut de paiement. En résumé, notre modèle estime qu'environ **62,2 %** des clients ne présenteront pas de défaut de paiement, tandis qu'environ **37,8 %** présentent un risque de défaut.

Analysons maintenant notre tableau de probabilités. Pour la classe **"Non"**, la plus faible probabilité associée à une prédiction "Non" est de **0.5** ($min = 0.5$), tandis que pour certains clients, le modèle était absolument certain qu'ils n'auraient pas de défaut de paiement ($max = 1$). En calculant la moyenne pour chaque client, la probabilité associée aux prédictions "Non" est de **83,31 %** ($mean = 0.831$), ce qui montre une forte confiance du modèle pour ces cas.

En ce qui concerne la classe **"Oui"**, nous obtenons les mêmes valeurs minimales et maximales. Néanmoins, en moyenne, les prédictions "Oui" sont associées à une probabilité de **75,9 %**.

En conclusion, notre modèle est relativement confiant dans ses prédictions, ce qui est primordial pour anticiper un défaut de paiement. En effet, une fausse prédiction peut entraîner une perte financière.

9 Conclusion

Au cours de la réalisation de ce projet, nous avons dû effectuer plusieurs tests et observations. Nous avons passé de longues heures à comprendre les liens entre les données, à partitionner les ensembles et à trouver des solutions pour adapter chaque échantillon aux différents modèles.

Trouver le bon compromis a été difficile, notamment lorsqu'il fallait choisir entre des échantillons avec valeurs manquantes et des ensembles réduits afin de limiter les biais. Tester divers modèles (**RPart, C5.0, SVM, Random Forest, Naïve Bayes**, etc.) nous a montré que chaque méthode apporte ses avantages et ses limites selon les critères (**coût, taux de succès, AUC**). Finalement, **Naïve Bayes** s'est révélé particulièrement pertinent grâce à sa simplicité, sa rapidité et sa capacité à fournir des probabilités interprétables.

Nous aurions pu affiner notre analyse en construisant une base de données plus précise, notamment en variant la valeur de la moyenne des **NA** en comparant l'âge des autres clients ayant des caractéristiques similaires sur les autres variables. Cela aurait permis d'obtenir des résultats plus précis et moins biaisés. Par manque de temps, nous n'avons pas pu réaliser tout ce que nous aurions souhaité, notamment la comparaison sur tous les échantillons, l'utilisation d'outils mathématiques plus avancés (distances, fonctions d'activation), l'implémentation de l'algorithme **XORB**, et bien d'autres éléments qui auraient permis d'obtenir un projet plus complet et un meilleur résultat.

Nous sommes tout de même satisfait de notre travaille malgré les contre-temps et obligations personelle. A noter que nous avons utilisé l'intelligence artificielle pour corriger les erreurs d'orthographes du rapport et pour repéter les lignes de codes redondante comme pour les différentes mesures pour chaque classifieur, le code qui résume les probabilités a été pris du site Stack Overflow et de R-Bloggers. Le reste des lignes de codes ont été faite par nous même en prenant exemple sur les différents tutoriels et cours du site du Professeur PASQUIER Nicolas.

10 Bibliographie et sources

- Nicolas PASQUIER, *Cours et tutoriels de Data Mining*, Université Côte d'Azur
- Reda CHHAIBI *Cours de Théorie des probabilités*, Université Côte d'Azur
- Olivier PANTZ *Cours de Machine Learning*, Université Côte d'Azur
- Documentation des outils R utilisés : https://cran.r-project.org
- Documentation Naïve Bayes : https://www.r-bloggers.com/2021/04/naive-bayes-classification-in-r/
- Stack Overflow : https://stackoverflow.com/questions