UFR 6
Université Paul Valéry, Montpellier III

Mémoire Professionnel S1M1

Challenge Kaggle: Loan Default Prediction

Adjimon VITOFFODJI

Janvier 2025

Remerciement

— Je remercie mon professeur Sophie Lèbre pour avoir valider la thématique du projet.
— Je remercie mon professeur Marine Demangeot pour le cours de Classification supervisée et non supervisée.
— Je remercie Sergei Shirkin Data Scientist at Dentsu Aegis pour avoir ouvert la compétition Loan Default Prediction. Le jeu de donnée mise à disposition qui a été une ressource essentielle pour ce projet.

Résumé

La capacité à anticiper les défauts de paiement permet aux institutions financières non seulement de réduire les risques, mais aussi d'améliorer la rentabilité en ajustant les taux d'intérêt, en optimisant la gestion des créances et en affinant leur stratégie de prêt. En outre, l'intégration des modèles de machine learning dans ces processus promet une automatisation accrue et une prise de décision rapide et efficace.

Dans cette étude, nous avons exploré plusieurs modèles de machine learning pour résoudre un problème de classification, en utilisant la validation croisée pour évaluer leur performance de manière robuste. Les résultats ont révélé que certains modèles se distinguent clairement par leurs performances.

Le modèle de forêts aléatoires a montré les meilleurs résultats en termes d'accuracy, atteignant une moyenne de 0.7793 en validation croisée, et une performance stable sur les différents plis. Cela a confirmé la capacité de ce modèle à gérer des données complexes, à généraliser correctement, et à éviter le surapprentissage. À l'opposé, des modèles comme le SVM ont montré une performance relativement faible, ce qui suggère que le choix des hyperparamètres ou la nature des données pourrait ne pas être adapté à ce type de modèle.

En parallèle, l'analyse du F1-score macro, utilisé pour traiter le problème des classes déséquilibrées, a permis d'identifier le modèle de régression logistique comme le meilleur, avec un score moyen de 0.6462. Cette métrique a été cruciale pour équilibrer les performances sur les classes majoritaires et minoritaires, garantissant une évaluation plus juste des modèles dans un contexte de données déséquilibrées.

La validation croisée a joué un rôle clé en réduisant les risques d'overfitting et en nous permettant de mieux comprendre la stabilité et la fiabilité des différents modèles. Cependant, il est important de noter que même le modèle de forêts aléatoires, bien qu'efficace sur l'ensemble d'entraînement, présente des limites sur le rappel de la classe minoritaire, ce qui pourrait être amélioré par l'ajustement des hyperparamètres ou l'utilisation d'approches spécifiques pour gérer le déséquilibre des classes.

Enfin, bien que la régression logistique ait été sélectionnée comme le meilleur modèle pour le F1-score macro, l'amélioration continue des performances reste possible en ajustant les paramètres et en explorant d'autres techniques de prétraitement et de pondération des classes. Cette étude met en lumière l'importance de choisir les bons modèles et les bonnes métriques en fonction des caractéristiques spécifiques des données et des objectifs de la tâche.

Table des matières

Table des figures

Liste des tableaux

Introduction

La prédiction du défaut de crédit est une tâche importante dans le domaine de la finance, car elle permet aux institutions financières de mieux évaluer le risque associé à chaque emprunt. Une prédiction précise peut aider à prendre des décisions éclairées sur l'octroi de crédits et à minimiser les pertes potentielles.

Dans le cadre de cette compétition Kaggle intitulée "Loan Default Prediction" organisée par Sergei Shirkin, le défi consiste à prédire si un emprunteur fera défaut sur un prêt en utilisant un jeu de données de caractéristiques financières. L'objectif principal de cette compétition est de développer un modèle capable de prédire la probabilité de défaut de crédit à partir des informations disponibles dans un jeu de données d'entraînement. Le jeu de données train.csv contient diverses caractéristiques des emprunteurs ainsi que la variable cible, "Credit Default", tandis que le jeu de données test.csv ne contient que les caractéristiques des emprunteurs, pour lesquels les prédictions doivent être faites.

Ce rapport présente l'approche adoptée pour résoudre ce problème de classification binaire, ainsi que les différentes étapes entreprises, telles que l'exploration des données, le prétraitement, la modélisation et l'évaluation des performances. L'évaluation du modèle est réalisée à l'aide de la métrique F1 score, qui équilibre la précision et le rappel, offrant une évaluation robuste de la performance du modèle sur des classes déséquilibrées.

Le défi propose une opportunité d'améliorer les compétences en machine learning, en particulier dans le domaine de la classification des risques, tout en mettant en pratique des techniques d'analyse de données, de sélection de caractéristiques et de validation de modèles.

Chapitre 1

Nettoyage, Exploration et Analyse descriptive des données

Sommaire

1.1 Nettoyage et Analyse du jeu de donnée train

1.1.1 Statistique descriptive des données train

Le jeu de donnée d'entrainement (train), dispose de 7500 lignes et 18 colonnes. L'analyse descriptive (voir figure 1.1) de ce dernier nous montre que les colonnes Annual Income, Months since last delinquent, Credit Score et Years in current qui contiennent plusieurs valeurs manquantes(Na). Par la suite, nous avons utilisé la médiane pour la gestion des valeurs manquantes.

skimpy summary

Data Summary

dataframe	Values
Number of rows	7500
Number of columns	18

Data Types

Column Type	Count
float64	12
string	4
int64	2

number

column_name	NA	NA %	mean	sd	p0	p25	p50	p75	p100	hist
Id	0	0	3750	2165	0	1875	3750	5624	7499	
Annual Income	1557	20.76	1366000	845300	164600	844300	1168000	1640000	10150000	
Tax Liens	0	0	0.03013	0.2716	0	0	0	0	7	
Number of Open Accounts	0	0	11.13	4.909	2	8	10	14	43	
Years of Credit History	0	0	18.32	7.042	4	13.5	17	21.8	57.7	
Maximum Open Credit	0	0	945200	16030000	0	279200	478200	793500	1305000000	
Number of Credit Problems	0	0	0.17	0.4986	0	0	0	0	7	
Months since last delinquent	4081	54.41	34.69	21.69	0	16	32	50	118	
Bankruptcies	14	0.19	0.1172	0.3472	0	0	0	0	4	
Current Loan Amount	0	0	11870000	31930000	11240	180200	309600	519900	100000000	
Current Credit Balance	0	0	289800	317900	0	114300	209300	360400	6507000	
Monthly Debt	0	0	18310	11930	0	10070	16080	23820	136700	
Credit Score	1557	20.76	1151	1604	585	711	731	743	7510	
Credit Default	0	0	0.2817	0.4499	0	0	0	1	1	

string

column_name	NA	NA %	words per row	total words
Home Ownership	0	0	1.6	11796
Years in current job	371	4.95	2	14821
Purpose	0	0	1.9	14434
Term	0	0	2	15000

End

FIGURE 1.1 – Statistique descriptive train.

1.1.2 Matrice de corrélation

1. Nous allons dans un premier temps représenter la matrice de corrélation avec les colonnes numériques(voir figure 1.2).

   On note une très faible correlation linéaire entre le Credit Default et les autre variables quantitative.

   De même, on note d'une part une faible liaison négative entre Credit Default et

Current Loan Amont et d'autre part une faible liaison postive entre Credit Default et Credit Score

Certaines colonnes ont des corrélations très faibles (proches de 0), ce qui peut indiquer qu'elles n'ont pas d'impact direct sur Credit Default.

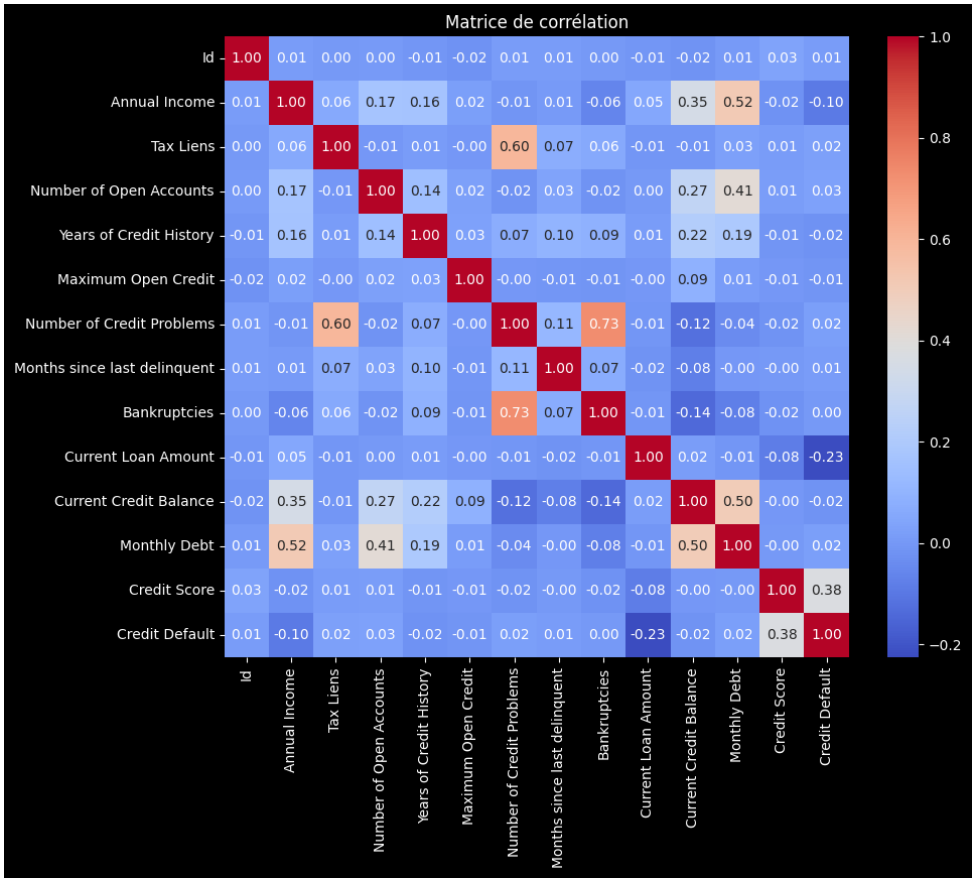


FIGURE 1.2 – Matrice de corrélation.

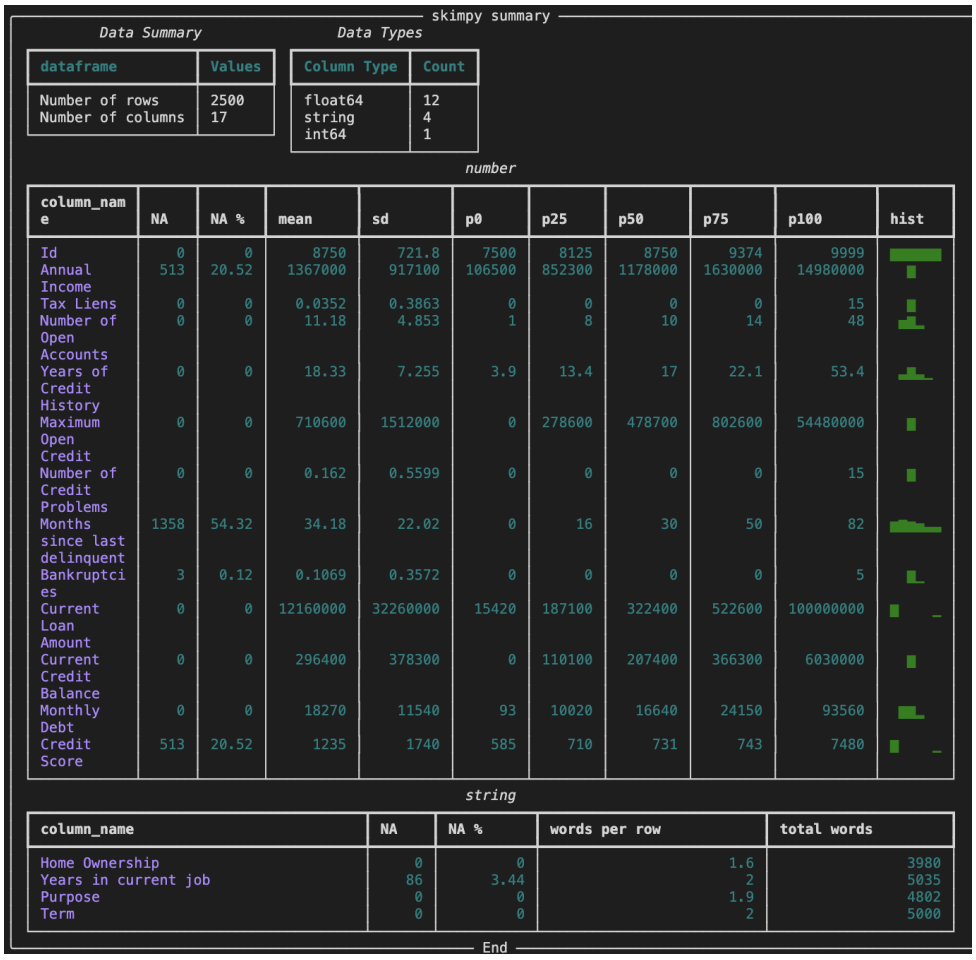
2. Par la suite, nous avons utilisé la technique d'encodage **LabelEncoder** pour encoder la colonne **Years in current job**, puis avons utilisé **One-Hot Encoding** pour encoder les autres colonnes catégorielles.

La corrélation la plus élevée avec Credit Default est pour Credit Score (0.44), ce qui peut indiquer une relation modérée entre ces deux variables. Term_Long Term et Term_Short Term ont des corrélations opposées (0.181 et -0.181), ce qui est logique car elles sont mutuellement exclusives dans un encodage one-hot.
Certaines colonnes ont des corrélations très faibles (proches de 0), ce qui peut indiquer qu'elles n'ont pas d'impact direct sur Credit Default.
Pour soumettre notre jeu de données train aux différents modèles de classification, nous avons besoin de convertir nos colonnes issues du **pd.get_dummies()** en de type **int**, cela facilitera leur traitement dans certains modèles ou calculs. Par la suite nous allons normaliser les colonnes pour éviter les biais dus à des échelles différentes. (Voir le résumé statistique après analyse et traitement des données à l'annexe Annexe 2.3).

1.2 Nettoyage et Analyse du Jeu de donnée test

1.2.1 Statistiques descriptives

Le jeu de donnée d'entrainement (train), dispose de 2500 lignes et 17 colonnes. L'analyse descriptive (voir figure 1.3) de ce dernier nous montre que les colonnes Annual Income, Months since last delinquent, Credit Score et Years in current qui contiennent plusieurs valeurs manquantes(Na). Par la suite, nous avons utilisé la médiane pour la gestion des valeurs manquantes. (voir Annexe 2.3).


skimpy summary

Data Summary

dataframe	Values
Number of rows	2500
Number of columns	17

Data Types

Column Type	Count
float64	12
string	4
int64	1

number

column_name	NA	NA %	mean	sd	p0	p25	p50	p75	p100	hist
Id	0	0	8750	721.8	7500	8125	8750	9374	9999	
Annual Income	513	20.52	1367000	917100	106500	852300	1178000	1630000	14980000	
Tax Liens	0	0	0.0352	0.3863	0	0	0	0	15	
Number of Open Accounts	0	0	11.18	4.853	1	8	10	14	48	
Years of Credit History	0	0	18.33	7.255	3.9	13.4	17	22.1	53.4	
Maximum Open Credit	0	0	710600	1512000	0	278600	478700	802600	54480000	
Number of Credit Problems	0	0	0.162	0.5599	0	0	0	0	15	
Months since last delinquent	1358	54.32	34.18	22.02	0	16	30	50	82	
Bankruptcies	3	0.12	0.1069	0.3572	0	0	0	0	5	
Current Loan Amount	0	0	12160000	32260000	15420	187100	322400	522600	100000000	
Current Credit Balance	0	0	296400	378300	0	110100	207400	366300	6030000	
Monthly Debt	0	0	18270	11540	93	10020	16640	24150	93560	
Credit Score	513	20.52	1235	1740	585	710	731	743	7480	

string

column_name	NA	NA %	words per row	total words
Home Ownership	0	0	1.6	3980
Years in current job	86	3.44	2	5035
Purpose	0	0	1.9	4802
Term	0	0	2	5000

End

FIGURE 1.3 – Statistique descriptive test.

1.2.2 Traitement du jeu de donnée test

Nous avons utilisé les mêmes techniques de nettoyage, de normalisation, de standardisation et d'encodage sur le jeu de donnée test. (voir le résumé statistique après nettoyage à l'annexe Annexe 2.3).

Chapitre 2

Entrainement des modèles et interprétation des résultats

Nous avons divisé les données(train), en données d'entraînement et de validation.

Sommaire

2.1 Entrainement des modèles

2.1.1 Arbre de décision

1. Accuracy (Précision globale) Accuracy = 0.7627

   Cela signifie que 76,27% des prédictions du modèle sont correctes. C'est la proportion des prédictions correctes (que la prédiction soit pour la classe 0 ou 1) parmi l'ensemble des prédictions réalisées. Cependant, l'accuracy peut être trompeuse si les classes sont déséquilibrées. Ce qui est le cas ici.

2. Métriques pour chaque classe

   Classe 0 (la classe majoritaire dans ce cas) Precision = 0.78 : Sur toutes les prédictions faites comme étant de la classe 0, 78% étaient correctes. Autrement dit, si le modèle prédit que l'observation appartient à la classe 0, il a 80% de chances de faire une prédiction correcte.

   Recall = 0.92 : Parmi toutes les vraies instances de la classe 0, 92% ont été correctement identifiées par le modèle. Cela indique que le modèle est assez bon pour identifier la classe 0, mais il y a encore 8% des exemples de la classe 0 qui sont manqués.

   F1-score = 0.85 : Le F1-score est une moyenne harmonique entre la precision et le recall. C'est une métrique utile quand on veut un compromis entre les deux, particulièrement dans des contextes où l'on veut éviter à la fois de fausses alertes (fausses positives) et des faux négatifs. Ici, le modèle montre un bon équilibre pour la classe 0.

   Classe 1 (la classe minoritaire)

   Precision = 0.67 : Sur toutes les prédictions faites comme étant de la classe 1, 67 % étaient correctes. Le modèle semble avoir plus de mal à prédire correctement la classe 1 que la classe 0.

   Recall = 0.39 : Parmi toutes les vraies instances de la classe 1, 39% ont été correctement identifiées. Cela suggère que le modèle a des difficultés à reconnaître la classe 1, et il manque 61 % des exemples de cette classe.

   F1-score = 0.49 : L'F1-score pour la classe 1 est relativement faible, ce qui reflète une combinaison de précision et de rappel qui n'est pas idéale. Cela montre que le modèle a un équilibre défavorable pour cette classe.

3. Métriques moyennes (macro et weighted avg) :

   Macro avg :

   Precision = 0.72, Recall = 0.65, F1-score = 0.67 : Ces valeurs sont les moyennes des métriques pour les deux classes, calculées sans tenir compte du déséquilibre des classes. La macro moyenne donne une idée générale de la performance du modèle sur toutes les classes, indépendamment de leur fréquence dans les données.

Weighted avg :

Precision = 0.75, Recall = 0.76, F1-score = 0.74 : La weighted moyenne prend en compte la fréquence des classes dans le dataset (plus de 5000 instances de la classe 0 contre environ 2100 pour la classe 1). Ces valeurs montrent qu'en pondérant par la taille des classes, le modèle est un peu plus performant sur la classe 0, mais reste assez équilibré.

En conclusion :

Le modèle est globalement performant, mais il a un déséquilibre de performance entre les deux classes.

Il fait un très bon travail pour prédire la classe 0, avec une précision élevée (78%) et un rappel élevé (92%).

Cependant, il a plus de mal avec la classe 1, avec un rappel faible (39%) et un F1-score assez bas (49%), ce qui suggère que le modèle manque beaucoup de cas positifs pour cette classe.

Cela pourrait être dû à un déséquilibre de classes (plus d'exemples de la classe 0 que de la classe 1)

TABLE 2.1 – Résultats de performance sur les données de validation-Arbre de décision

Classe	**Precision**	**Recall**	**F1-Score**	**Support**
0	0.78	0.92	0.85	1059
1	0.67	0.39	0.49	441
Accuracy			0.76	1500
Macro Avg	0.72	0.65	0.67	1500
Weighted Avg	0.75	0.76	0.74	1500

Accuracy sur les données de validation : 0.7626666666666667

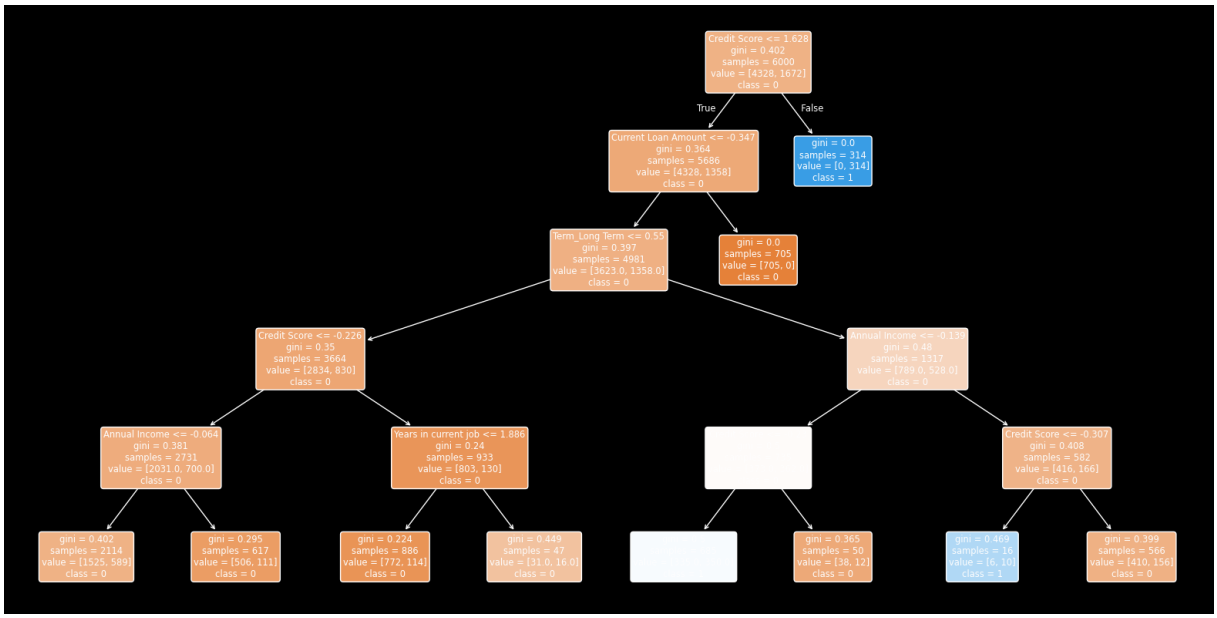


FIGURE 2.1 – arbre de décision.

La matrice de confusion et la courbe de ROC (voir Annexe 2.3) indiquent que le modèle a une bonne capacité à distinguer entre les classes, mais il n'est pas parfait.

2.1.2 Régression logistique

1. Accuracy : 76.6%

   L'accuracy mesure la proportion globale des prédictions correctes. Ici, le modèle prédit correctement 76,6 % des cas. Bien que ce soit un bon résultat, ce chiffre seul ne suffit pas à évaluer la performance du modèle, surtout si les classes sont déséquilibrées. Ce qui est le cas ici.

2. Precision, Recall et F1-Score pour la classe 0 (Non-défaillant) :

   Precision = 0.76 : Cela signifie que parmi toutes les prédictions faites par le modèle comme étant "0" (non défaut), 76% étaient correctes.

   Recall = 0.98 : Cela signifie que parmi tous les vrais cas "0", le modèle a réussi à en prédire 98%. Cela indique que le modèle est très bon pour détecter les non-défaillants.

   F1-Score = 0.86 : Le F1-score est une moyenne harmonique entre la précision et le rappel, donc ici, il est relativement élevé, ce qui signifie que le modèle a une bonne capacité à détecter les cas de "0" tout en minimisant les faux positifs.

3. Precision, Recall et F1-Score pour la classe 1 (Défaillant) :

   Precision = 0.85 : Cela signifie que parmi toutes les prédictions faites par le modèle comme étant "1" (défaut), 85% étaient correctes.

   Recall = 0.25 : Ici, seulement 25% des vrais cas "1" ont été détectés. C'est un faible score, ce qui signifie que le modèle a du mal à identifier les clients en défaut.

F1-Score = 0.38 : Ce score est également faible, ce qui reflète une mauvaise performance du modèle pour prédire les défauts

4. Macro et Weighted Averages :

   Macro avg : Cela donne une moyenne simple des scores de chaque classe (sans tenir compte de la proportion des classes), ce qui donne un score de 0.80 pour la précision, 0.62 pour le rappel et 0.63 pour le F1-score.

   Weighted avg : Cette moyenne prend en compte la distribution des classes. Ici, la précision et le rappel sont équilibrés à 0.79 et 0.77 respectivement, avec un F1-score global de 0.72.

   En conclusion : Le modèle est globalement performant, mais faible déséquilibre de performance entre les deux classes.

TABLE 2.2 – Résultats de performance sur les données de validation-Régression Logistique

Classe	**Precision**	**Recall**	**F1-Score**	**Support**
0	0.76	0.98	0.86	1059
1	0.85	0.25	0.38	441
Accuracy			0.77	1500
Macro Avg	0.80	0.61	0.62	1500
Weighted Avg	0.79	0.77	0.72	1500

Accuracy sur les données de validation : 0.766

La matrice de confusion et la courbe de ROC associées (voir Annexe 2.3) indiquent que le modèle a une performance acceptable à distinguer entre les deux classes.

2.1.3 Forêt aléatoire

1. Accuracy (Précision globale) : L'accuracy est de 76.33%, ce qui signifie que 76.33% des prédictions du modèle sont correctes sur l'ensemble de validation. Cependant, cette métrique seule peut être trompeuse si les classes sont déséquilibrées.

2. Métriques pour chaque classe :

   Classe 0 (support : 1059 instances) :

   Précision (Precision) = 0.76 : Cela signifie que, parmi les prédictions de la classe 0, 76% étaient correctes.

   Rappel (Recall) = 0.96 : Cela signifie que le modèle a identifié 96% des instances appartenant réellement à la classe 0.

   F1-score = 0.85 : Le modèle a bien équilibré précision et rappel pour la classe 0, montrant une bonne performance.

Classe 1 (support : 441 instances) :

Précision (Precision) = 0.76 : Cela signifie que, parmi les prédictions de la classe 1, 76% étaient correctes.

Rappel (Recall) = 0.29 : Cela signifie que le modèle n'a identifié que 29% des instances appartenant réellement à la classe 1.

Problème notable : Le modèle a de grandes difficultés à détecter la classe 1.

F1-score = 0.42 : Un faible F1-score montre que le modèle est déséquilibré dans sa capacité à prédire correctement la classe 1.

3. Macro Avg (Moyenne non pondérée) :

   Précision, Rappel, F1-score : Ces moyennes prennent en compte les métriques de chaque classe avec un poids égal.

   Précision moyenne = 0.76 ; Rappel moyen = 0.62 ; F1-score moyen = 0.63. Cela montre que, en moyenne, les performances du modèle sont limitées par sa faible capacité à gérer la classe 1.

4. Weighted Avg (Moyenne pondérée par le support) : Ces moyennes tiennent compte de l'importance relative des classes (plus de poids pour la classe 0, qui a plus d'exemples).

   Précision pondérée = 0.76 ; Rappel pondéré = 0.76 ; F1-score pondéré = 0.72. Ces scores montrent que les performances globales semblent correctes, mais elles sont biaisées par la bonne performance sur la classe majoritaire (classe 0).

   En résumé :

   Le modèle Random Forest montre une bonne capacité à identifier la classe majoritaire (classe 0), mais il a des difficultés à prédire la classe minoritaire (classe 1).

TABLE 2.3 – Résultats de performance sur les données de validation Forêt aléatoire

Classe	**Precision**	**Recall**	**F1-Score**	**Support**
0	0.76	0.96	0.85	1059
1	0.76	0.29	0.42	441
Accuracy			0.76	1500
Macro Avg	0.76	0.62	0.63	1500
Weighted Avg	0.76	0.76	0.72	1500

Accuracy sur les données de validation : 0.763

La matrice de confusion et la courbe de ROC associées (voir Annexe 2.3) indiquent que (l'AUC = 0.76) le modèle a performance acceptable à distinguer entre les deux classes.

2.1.4 Support Vector Machines (SVM)

1. Accuracy (Précision globale) : 0.452 L'accuracy est de 45,2%, ce qui signifie que le modèle fait des prédictions correctes environ 45% du temps.
2. Métriques pour chaque classe

   Pour la classe 0 :

   La précision de la classe 0 est assez bonne, ce qui signifie que, parmi toutes les prédictions où le modèle a dit classe 0, 71% étaient correctes.

   Le rappel pour la classe 0 est assez faible, ce qui signifie que parmi toutes les instances réelles de la classe 0, le modèle n'a réussi à en identifier que 38%.

   Le F1-score(0.50) pour la classe 0 est relativement moyen, ce qui reflète un compromis entre la précision et le rappel.

   Pour la classe 1 :

   Précision de 0.29 (29%) : La précision de la classe 1 est faible, ce qui signifie qu'une grande proportion des prédictions de la classe 1 étaient incorrectes.

   Recall de 0.62 (62%) : Le rappel pour la classe 1 est meilleur, ce qui signifie que le modèle a identifié 62% des instances réelles de cette classe.

   F1-score de 0.40 : Le F1-score pour la classe 1 est plus faible, ce qui indique que le modèle lutte davantage pour classer correctement les instances de cette classe.

   Déséquilibre des classes :

   Le déséquilibre des classes explique en grande partie cette performance biaisée. Le modèle est plus performant pour prédire la classe majoritaire (classe 0), ce qui conduit à des performances médiocres pour la classe minoritaire (classe 1). Cela se traduit par un recall faible pour la classe 1 et un biais vers la prédiction de la classe 0.

   En résumé :

   Le modèle SVM ne semble pas très performant, particulièrement pour la classe minoritaire (1). Il fait mieux sur la classe majoritaire (0) mais il est relativement mauvais pour détecter les instances de la classe 1.

TABLE 2.4 – Résultats de performance sur les données de validation - SVM

Classe	**Precision**	**Recall**	**F1-Score**	**Support**
0	0.71	0.38	0.50	1059
1	0.29	0.62	0.40	441
Accuracy			0.45	1500
Macro Avg	0.50	0.50	0.45	1500
Weighted Avg	0.59	0.45	0.47	1500

Accuracy sur les données de validation : 0.452

La matrice de confusion et la courbe de ROC associées (voir Annexe 2.3) indiquent que (l'AUC=0.51) le modèle ne parvient pas à mieux prédire que de simples prédictions aléatoires.

2.1.5 K Nearest Neighbors (KNN)

1. Accuracy (Précision globale) : 0.699 (7%)

   L'accuracy est de 0.699 (70%), indique que le modèle prédit correctement 70% des échantillons de validation. Cependant, l'accuracy seule peut être trompeuse, en particulier si les classes sont déséquilibrées (ce qui semble être le cas ici)

2. Métrique pour les classes

   Pour la classe 0 :

   Précision (Precision) = 0.70. Cela signifie que, parmi toutes les prédictions faites comme étant de la classe 0, 70% étaient correctes.

   Rappel (Recall) = 0.99. Cela signifie que 99% des instances de la classe 0 ont été correctement identifiées par le modèle. Le modèle est donc très performant pour la classe 0.

   F1-Score = 0.82. Une bonne combinaison de la précision et du rappel pour la classe 0. Le modèle fait bien pour la classe majoritaire.

   Pour la classe 1 :

   Précision (Precision) = 0.14. Parmi les prédictions faites comme étant de la classe 1,seulement 14% étaient correctes.

   Rappel (Recall) = 0.00. Le modèle n'a pratiquement pas réussi à identifier les instances réelles de la classe 1 (seules quelques-unes, voire aucune, ont été correctement classées)

   F1-Score = 0.01. La combinaison de la précision et du rappel est faible, ce qui indique que la classe 1 est mal prédite.

3. Moyennes(sans pondération des effectifs) Macro avg :

   La macro-moyenne montre des résultats faibles (0.42 pour F1-score), en raison des mauvaises performances sur la classe 1.

4. Weightedtenant compte de la proportion des instances dans chaque classe() avg :

   Avec une pondération pour la classe majoritaire (classe 0), les scores sont légèrement meilleurs (0.58 pour F1-score), mais toujours médiocres.

   Déséquilibre des classes :

   Le déséquilibre des classes explique en grande partie cette performance biaisée. Le modèle est plus performant pour prédire la classe majoritaire (classe 0), ce qui conduit à des performances médiocres pour la classe minoritaire (classe 1). Cela se traduit par un recall faible pour la classe 1 et un biais vers la prédiction de la classe 0.

En résumé :

En résumé, le modèle KNN avec K=20 montre de bonnes performances pour la classe 0 (majoritaire), mais échoue à prédire la classe 1 (minoritaire).

TABLE 2.5 – Résultats de performance pour KNN avec K=20 sur les données de validation

Classe	**Precision**	**Recall**	**F1-Score**	**Support**
0	0.70	0.99	0.82	1059
1	0.14	0.00	0.01	441
Accuracy			0.70	1500
Macro Avg	0.42	0.50	0.42	1500
Weighted Avg	0.54	0.70	0.58	1500

Meilleur nombre de voisins (**K**) : 20
Accuracy sur les données de validation : 0.699.

La matrice de confusion et la courbe de ROC associées (voir Annexe 2.3) indiquent : Bien que le modèle ait un AUC relativement supérieur à 0.5, cela suggère qu'il n'est pas très bon pour séparer les classes, en particulier dans un cas de classe déséquilibrée. le modèle ne parvient pas à mieux prédire que de simples prédictions aléatoires.

2.2 Comparaison et Validation croisée

2.2.1 Comparaison et Validation croisée avec Accuracy

Les résultats de la validation croisée montrent la moyenne d'accuracy pour chaque modèle sur les 5 plis. Ces scores aident à évaluer la stabilité et la performance générale des modèles, tout en réduisant les risques de sur-apprentissage (overfitting).

1. Précision globale
   — Arbre de décision (Accuracy moyenne = 0.6908) : L'arbre de décision a une performance correcte. Performance moyenne, avec une forte variation entre les folds (de 0.6791 à 0.6991). Cela indique que l'arbre de décision est sensible aux variations dans les données d'entraînement, probablement dû à un surapprentissage (overfitting) dans certains cas.
   — Régression logistique (Accuracy moyenne = 0.6872) : La régression logistique montre une performance stable, mais légèrement inférieure à celle des forêts aléatoires. C'est un modèle linéaire, donc il pourrait être limité si les relations entre les variables ne sont pas linéaires.
   — Forêts aléatoires (Accuracy moyenne = 0.7793) : C'est clairement le modèle le plus performant en validation croisée. Il combine plusieurs arbres de décision

pour améliorer la généralisation et réduire le surapprentissage. La faible variance entre les plis confirme qu'il s'adapte bien aux données.
   — SVM (Accuracy moyenne = 0.5130) : Le SVM affiche des performances très faibles. Cela pourrait être dû à des hyperparamètres non optimaux ou à un déséquilibre dans les classes.
   — KNN (Accuracy moyenne = 0.6768) : KNN a des performances modestes, probablement à cause de la nature des données ou d'un choix sous-optimal du nombre de voisins. Cela peut indiquer que KNN n'est pas adapté à vos données ou qu'il est sensible aux déséquilibres ou à des caractéristiques spécifiques.
2. Comparaison des scores en validation croisée :
   — Forêts aléatoires se démarque avec la meilleure moyenne (0.7793), suivie par l'arbre de décision et la régression logistique.
   — SVM et KNN montrent des scores faibles, ce qui les rend moins fiables pour ce problème.

     Cela indique que les modèles non linéaires comme les forêts aléatoires ou les arbres de décision peuvent mieux capturer les patterns complexes des données.
3. **Évaluation sur le test set pour le meilleur modèle** :
   — **Le modèle Forêts aléatoires, choisi comme meilleur modèle**, montre une accuracy de 0.7613 sur le jeu de test. Cependant, une analyse plus fine des métriques (précision, rappel, F1-score) révèle des points importants :
   — Classe 0 (majoritaire) : Très bien prédite avec une précision de 76% et un rappel de 97%.
   — Classe 1 (minoritaire) : Faible rappel (26%), ce qui indique que de nombreuses instances de cette classe ne sont pas correctement identifiées.

     Ce problème de rappel faible pour la classe minoritaire peut être dû à un déséquilibre des classes ou au choix des hyperparamètres. (Voir Annexe 2.3).

2.2.2 Comparaison et Validation croisée F1-score macro

Pour améliorer la pondération des classes, nous allons utiliser l'argument class_weight='balanced' dans les modèles comme Arbre de décision, RandomForestClassifier, LogisticRegression et SVM.
Les classes étant déséquilibrées, pour accorder donc la même importance à chaque classe, indépendamment de sa taille, nous allons utiliser le F1-score macro.
Le F1-score macro est une bonne métrique pour évaluer un modèle, surtout lorsqu'on a des classes déséquilibrées, car il donne une importance égale à toutes les classes, indépendamment de leur fréquence.

1. Précision globale

— **Régression Logistique a la meilleure moyenne F1-score macro (0.6462)**, ce qui indique qu'elle a bien équilibré la précision et le rappel pour toutes les classes.
— Forêts Aléatoires suit avec un F1-score macro moyen de 0.6379, et Arbre de décision est un peu derrière à 0.6177.
— SVM et KNN sont en bas du classement, avec des moyennes de 0.4737 et 0.4912 respectivement, ce qui montre qu'ils ont de moins bons résultats sur les classes déséquilibrées.

2. **Meilleur modèle : Régression Logistique**

   Après avoir effectué la validation croisée, le modèle de Régression Logistique a été sélectionné comme le meilleur, car il a obtenu la meilleure moyenne F1-score macro.

3. **Performance sur le jeu de test :**
   — **Accuracy : 0.686**, ce qui signifie que le modèle prédit correctement 68.6% des exemples du jeu de test.
   — La précision pour la classe 0 est relativement bonne (0.83), mais pour la classe 1, elle est plus faible (0.48), ce qui indique que le modèle fait moins bien pour la classe minoritaire.
   — Le rappel pour la classe 1 (0.65) est supérieur à sa précision, ce qui montre que le modèle détecte une partie significative des exemples de cette classe, mais encore une fois pas aussi bien que la classe 0.
   — Le f1-score pour la classe 0 est de 0.76, tandis que pour la classe 1, il est de 0.55, ce qui reflète l'impact du déséquilibre des classes.

En conclusion :
Le modèle de régression logistique semble être le meilleur choix basé sur **F1-score macro**, car il maintient un bon équilibre entre précision et rappel pour toutes les classes, en particulier la classe majoritaire.
Toutefois, même si la précision de la classe 0 est bonne, la classe minoritaire (classe 1) a une précision relativement faible, ce qui peut suggérer un manque de puissance dans la prédiction des cas de cette classe.

La matrice de confusion associées (voir Annexe 2.3) indique que ce modèle semble être le meilleur choix basé sur F1-score macro.

2.2.3 Recommandation

Suggestions pour améliorer la performance sur la classe minoritaire :

1. Ré-échantillonnage des données :

TABLE 2.6 – Résultats de performance sur les données test

Classe	Precision	Recall	F1-Score	Support
0	0.83	0.70	0.76	1059
1	0.48	0.65	0.55	441
Accuracy			0.69	1500
Macro Avg	0.65	0.68	0.65	1500
Weighted Avg	0.72	0.69	0.70	1500

Accuracy sur les données test : 0.686

On peut utiliser l'oversampling (par exemple, SMOTE) pour augmenter le nombre d'exemples de la classe minoritaire, ou l'undersampling pour réduire la classe majoritaire.

2. Ajustement des seuils de décision :

   Pour les modèles comme la régression logistique, on peux ajuster le seuil de décision pour favoriser la prédiction de la classe minoritaire.

3. Utilisation de modèles plus complexes :

   Par exemple, des modèles comme XGBoost ou LightGBM, qui sont souvent plus efficaces dans la gestion des classes déséquilibrées.

2.3 Prédire Credit Default avec le meilleur modèle

Nous avons sélectionner le meilleur modèle par validation croisé en utilisant le F1score Macro. Le meilleur modèle retenu est celui de la logistique. Par la suite, nous avons utiliser ce meilleur modèle d'entrainement, sur le jeu de donné test pour prédire les nouvelles valeurs de la variable Credit Default. Voici une sortie des cinq premières lignes de cette valeur :

TABLE 2.7 – Données de crédit défaut prédite

Id	Credit Default
7500	0
7501	0
7502	1
7503	0
7504	1

Conclusion

Ce projet s'inscrit dans le cadre de la prédiction du défaut de paiement des prêts (Loan Default Prediction), un domaine d'une importance capitale pour le secteur financier. En permettant aux institutions bancaires et financières de prédire avec précision les risques de défaut, ce projet peut contribuer à une meilleure gestion des risques, à la réduction des pertes et à une prise de décision plus éclairée. L'approche par machine learning utilisée dans ce projet s'appuie sur des techniques avancées telles que l'analyse des données clients, leur historique de crédit et d'autres facteurs économiques et sociaux pour créer des modèles de prédiction robustes.

Cette étude a permis de comparer plusieurs modèles de machine learning dans le cadre d'un problème de classification, en évaluant leurs performances à travers des métriques adaptées telles que l'accuracy et le F1-score macro. Les résultats ont montré que les modèles basés sur les forêts aléatoires et la régression logistique sont particulièrement efficaces pour gérer les défis posés par les données déséquilibrées, chaque modèle ayant ses propres forces et limites.

L'approche de validation croisée a été essentielle pour garantir une évaluation rigoureuse des modèles et éviter les biais de sur-apprentissage, tout en assurant une meilleure généralisation des résultats sur de nouveaux ensembles de données. Malgré les performances encourageantes de certains modèles, il est apparu que des améliorations étaient possibles, notamment en ajustant les hyperparamètres ou en explorant des techniques plus sophistiquées pour gérer le déséquilibre des classes.

Cependant, certaines limitations demeurent dans cette étude, notamment la gestion du recall pour les classes minoritaires, un aspect qui mérite une attention particulière pour une optimisation complète des modèles. De plus, l'expérimentation d'autres algorithmes, tels que des méthodes d'ensemble plus avancées ou des réseaux de neurones, pourrait offrir des performances améliorées.

Pour des travaux futurs, il serait intéressant d'élargir cette analyse en intégrant davantage de techniques de pré-traitement des données, ainsi que l'optimisation des modèles à travers des approches comme la recherche bayésienne des hyperparamètres. Une meilleure gestion de l'équilibre entre les classes pourrait aussi constituer un axe d'amélioration important pour maximiser l'efficacité des prédictions sur les classes minoritaires.

En conclusion, cette étude met en lumière l'importance de choisir judicieusement les modèles et les métriques, en fonction des spécificités des données et des objectifs de la tâche. Les résultats obtenus ouvrent la voie à de nouvelles explorations et à l'amélioration continue des modèles pour des applications réelles.

Annexes

Annexe 3

skimpy summary

Data Summary

dataframe	Values
Number of rows	7500
Number of columns	36

Data Types

Column Type	Count
float64	34
int64	2

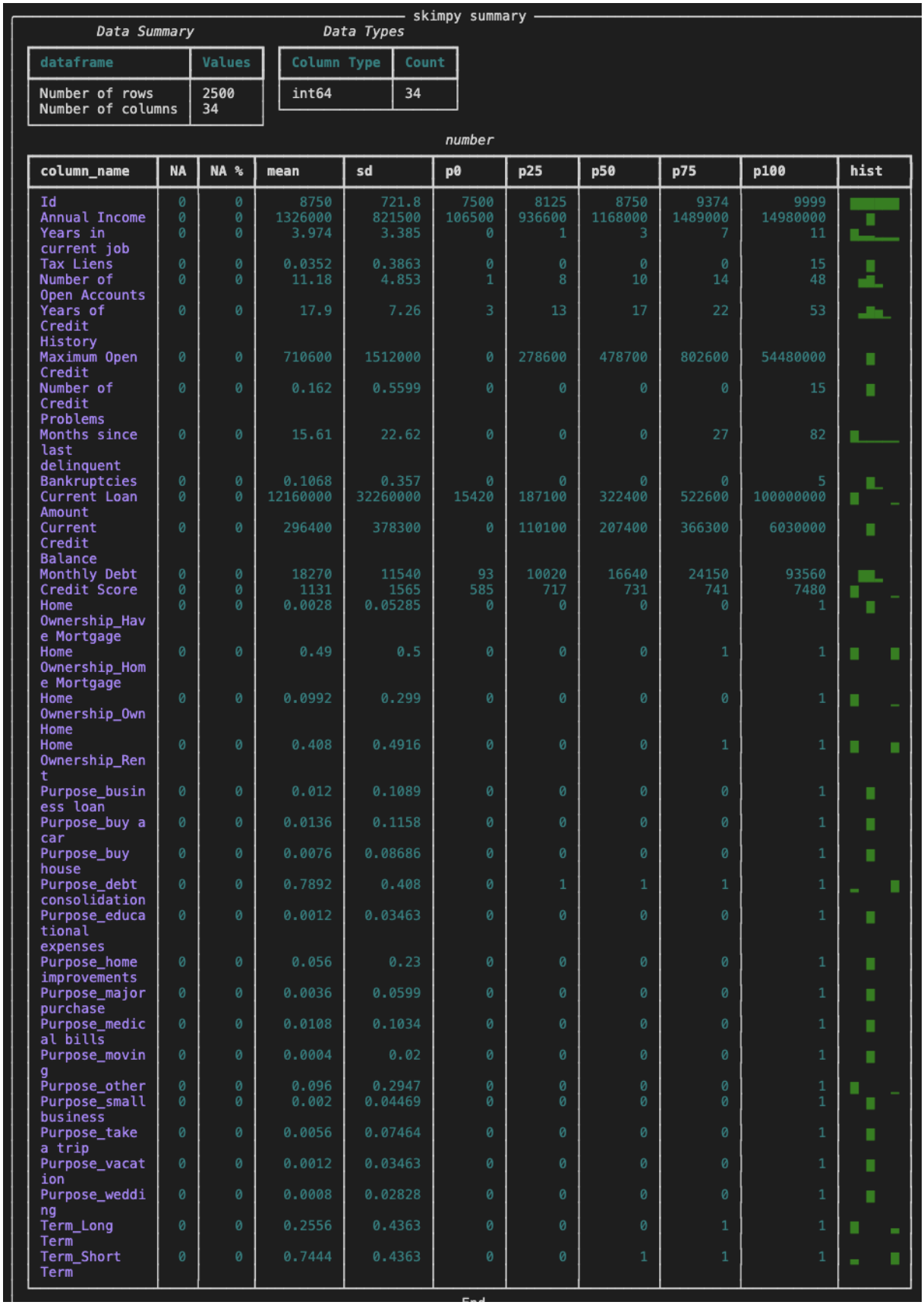
number

column_name	NA	NA %	mean	sd	p0	p25	p50	p75	p100	hist
Id	0	0	3750	2165	0	1875	3750	5624	7499	▇▇▇▇▇
Annual Income	0	0	5.969e-17	1	-1.534	-0.5209	-0.2073	0.2309	11.66	▇▁
Years in current job	0	0	6.253e-17	1	-1.169	-0.8782	-0.2963	0.8673	2.031	▃▂▃▅▇
Tax Liens	0	0	2.274e-17	1	-0.111	-0.111	-0.111	-0.111	25.66	▇
Number of Open Accounts	0	0	3.647e-17	1	-1.86	-0.6378	-0.2304	0.5845	6.493	▇▃▂
Years of Credit History	0	0	2.937e-17	1	-1.97	-0.6935	-0.1261	0.4412	5.547	▂▇▃▂
Maximum Open Credit	0	0	-4.737e-18	1	-0.05898	-0.04155	-0.02914	-0.009463	81.36	▇
Number of Credit Problems	0	0	-7.011e-17	1	-0.341	-0.341	-0.341	-0.341	13.7	▇
Months since last delinquent	0	0	-9.142e-17	1	-0.6983	-0.6983	-0.6983	0.5822	4.512	▇▂▁▁
Bankruptcies	0	0	6.348e-17	1	-0.3371	-0.3371	-0.3371	-0.3371	11.19	▇▁
Current Loan Amount	0	0	4.832e-17	1	-0.3716	-0.3663	-0.3622	-0.3556	2.761	▇▁▁▁▁
Current Credit Balance	0	0	1.326e-17	1	-0.9119	-0.5524	-0.2533	0.222	19.56	▇
Monthly Debt	0	0	4.547e-17	1	-1.536	-0.6915	-0.1877	0.4615	9.925	▇▂
Credit Score	0	0	-4.169e-17	1	-0.333	-0.2405	-0.2314	-0.2252	4.482	▇
Credit Default	0	0	0.2817	0.4499	0	0	0	1	1	▇▁▁▂
Home Ownership_Have Mortgage	0	0	-9.474e-18	1	-0.04003	-0.04003	-0.04003	-0.04003	24.98	▇
Home Ownership_Home Mortgage	0	0	-1.895e-18	1	-0.9703	-0.9703	-0.9703	1.031	1.031	▇▁▁▁▇
Home Ownership_Own Home	0	0	-1.895e-17	1	-0.3073	-0.3073	-0.3073	-0.3073	3.255	▇▁▁▁▁
Home Ownership_Rent	0	0	9.474e-18	1	-0.8636	-0.8636	-0.8636	1.158	1.158	▇▁▁▁▇
Purpose_business loan	0	0	1.611e-17	1	-0.1323	-0.1323	-0.1323	-0.1323	7.559	▇
Purpose_buy a car	0	0	-2.653e-17	1	-0.1139	-0.1139	-0.1139	-0.1139	8.782	▇
Purpose_buy house	0	0	0	1	-0.06748	-0.06748	-0.06748	-0.06748	14.82	▇
Purpose_debt consolidation	0	0	6.916e-17	1	-1.954	0.5116	0.5116	0.5116	0.5116	▁▁▁▁▇
Purpose_educational expenses	0	0	2.368e-18	1	-0.03654	-0.03654	-0.03654	-0.03654	27.37	▇
Purpose_home improvements	0	0	-2.463e-17	1	-0.2411	-0.2411	-0.2411	-0.2411	4.148	▇
Purpose_major purchase	0	0	-2.274e-17	1	-0.07323	-0.07323	-0.07323	-0.07323	13.66	▇
Purpose_medical bills	0	0	-7.105e-17	1	-0.09776	-0.09776	-0.09776	-0.09776	10.23	▇
Purpose_moving	0	0	2.937e-17	1	-0.03833	-0.03833	-0.03833	-0.03833	26.09	▇
Purpose_other	0	0	6.632e-18	1	-0.3119	-0.3119	-0.3119	-0.3119	3.206	▇▁▁▁▁
Purpose_renewable energy	0	0	-2.842e-18	1	-0.01633	-0.01633	-0.01633	-0.01633	61.23	▇
Purpose_small business	0	0	-2.179e-17	1	-0.05898	-0.05898	-0.05898	-0.05898	16.95	▇
Purpose_take a trip	0	0	1.99e-17	1	-0.07041	-0.07041	-0.07041	-0.07041	14.2	▇
Purpose_vacation	0	0	-4.263e-18	1	-0.03268	-0.03268	-0.03268	-0.03268	30.6	▇
Purpose_wedding	0	0	1.658e-17	1	-0.04477	-0.04477	-0.04477	-0.04477	22.34	▇
Term_Long Term	0	0	5.921e-17	1	-0.5915	-0.5915	-0.5915	1.691	1.691	▇▁▁▁▇
Term_Short Term	0	0	-5.921e-17	1	-1.691	-1.691	0.5915	0.5915	0.5915	▇▁▁▁▇

End

FIGURE 2.2 – Résumé Statistique descriptive train.

Annexe 4



skimpy summary

Data Summary

dataframe	Values
Number of rows	2500
Number of columns	34

Data Types

Column Type	Count
int64	34

number

column_name	NA	NA %	mean	sd	p0	p25	p50	p75	p100	hist
Id	0	0	8750	721.8	7500	8125	8750	9374	9999	
Annual Income	0	0	1326000	821500	106500	936600	1168000	1489000	14980000	
Years in current job	0	0	3.974	3.385	0	1	3	7	11	
Tax Liens	0	0	0.0352	0.3863	0	0	0	0	15	
Number of Open Accounts	0	0	11.18	4.853	1	8	10	14	48	
Years of Credit History	0	0	17.9	7.26	3	13	17	22	53	
Maximum Open Credit	0	0	710600	1512000	0	278600	478700	802600	54480000	
Number of Credit Problems	0	0	0.162	0.5599	0	0	0	0	15	
Months since last delinquent	0	0	15.61	22.62	0	0	0	27	82	
Bankruptcies	0	0	0.1068	0.357	0	0	0	0	5	
Current Loan Amount	0	0	12160000	32260000	15420	187100	322400	522600	100000000	
Current Credit Balance	0	0	296400	378300	0	110100	207400	366300	6030000	
Monthly Debt	0	0	18270	11540	93	10020	16640	24150	93560	
Credit Score	0	0	1131	1565	585	717	731	741	7480	
Home Ownership_Have Mortgage	0	0	0.0028	0.05285	0	0	0	0	1	
Home Ownership_Home Mortgage	0	0	0.49	0.5	0	0	0	1	1	
Home Ownership_Own Home	0	0	0.0992	0.299	0	0	0	0	1	
Home Ownership_Rent	0	0	0.408	0.4916	0	0	0	1	1	
Purpose_business loan	0	0	0.012	0.1089	0	0	0	0	1	
Purpose_buy a car	0	0	0.0136	0.1158	0	0	0	0	1	
Purpose_buy house	0	0	0.0076	0.08686	0	0	0	0	1	
Purpose_debt consolidation	0	0	0.7892	0.408	0	1	1	1	1	
Purpose_educational expenses	0	0	0.0012	0.03463	0	0	0	0	1	
Purpose_home improvements	0	0	0.056	0.23	0	0	0	0	1	
Purpose_major purchase	0	0	0.0036	0.0599	0	0	0	0	1	
Purpose_medical bills	0	0	0.0108	0.1034	0	0	0	0	1	
Purpose_moving	0	0	0.0004	0.02	0	0	0	0	1	
Purpose_other	0	0	0.096	0.2947	0	0	0	0	1	
Purpose_small business	0	0	0.002	0.04469	0	0	0	0	1	
Purpose_take a trip	0	0	0.0056	0.07464	0	0	0	0	1	
Purpose_vacation	0	0	0.0012	0.03463	0	0	0	0	1	
Purpose_wedding	0	0	0.0008	0.02828	0	0	0	0	1	
Term_Long Term	0	0	0.2556	0.4363	0	0	0	1	1	
Term_Short Term	0	0	0.7444	0.4363	0	0	1	1	1	

End

FIGURE 2.3 – Résumé Statistique descriptive jeu test.

Annexe 5

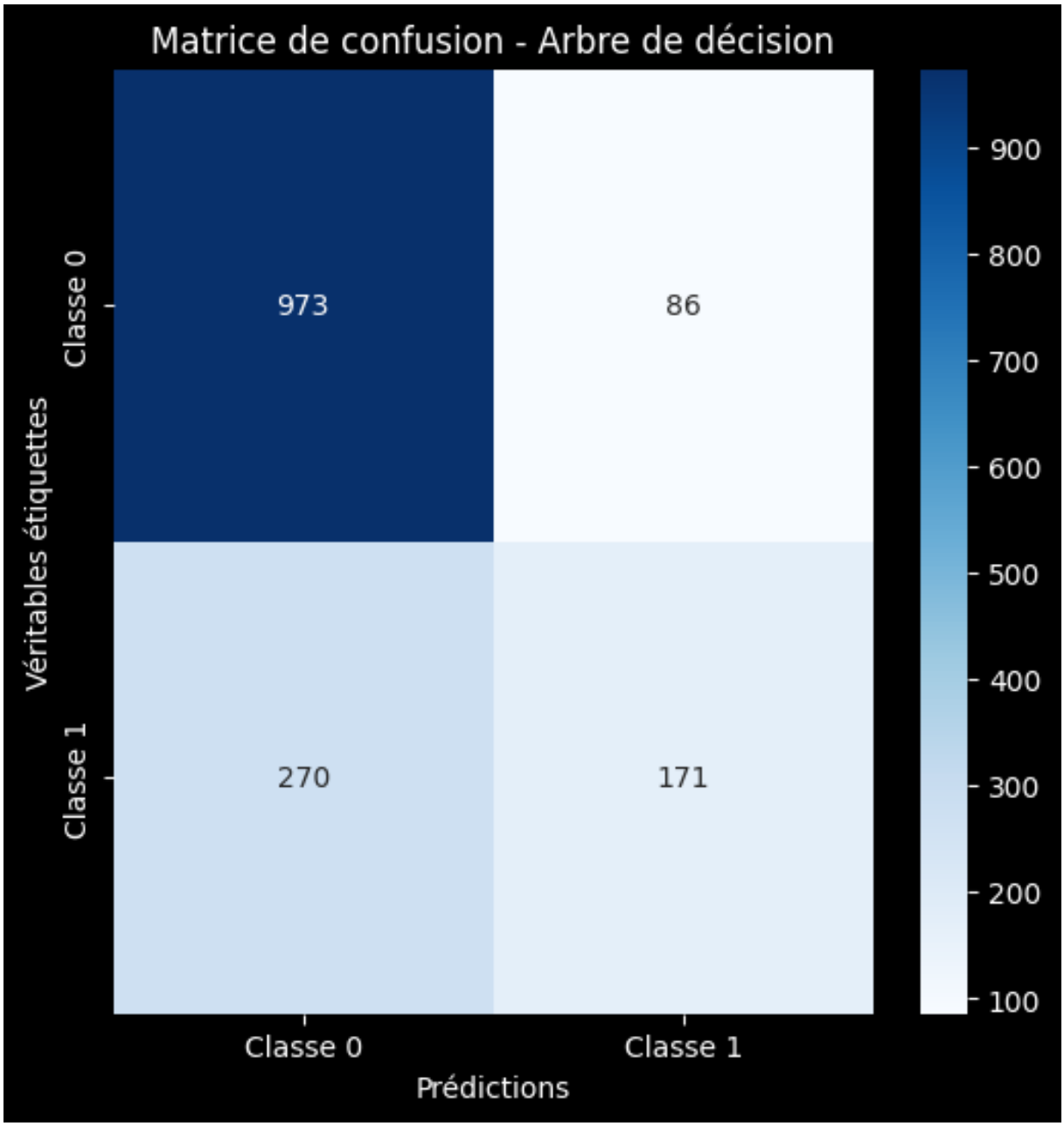


FIGURE 2.4 – Matrice de confusion Arbre de décision.

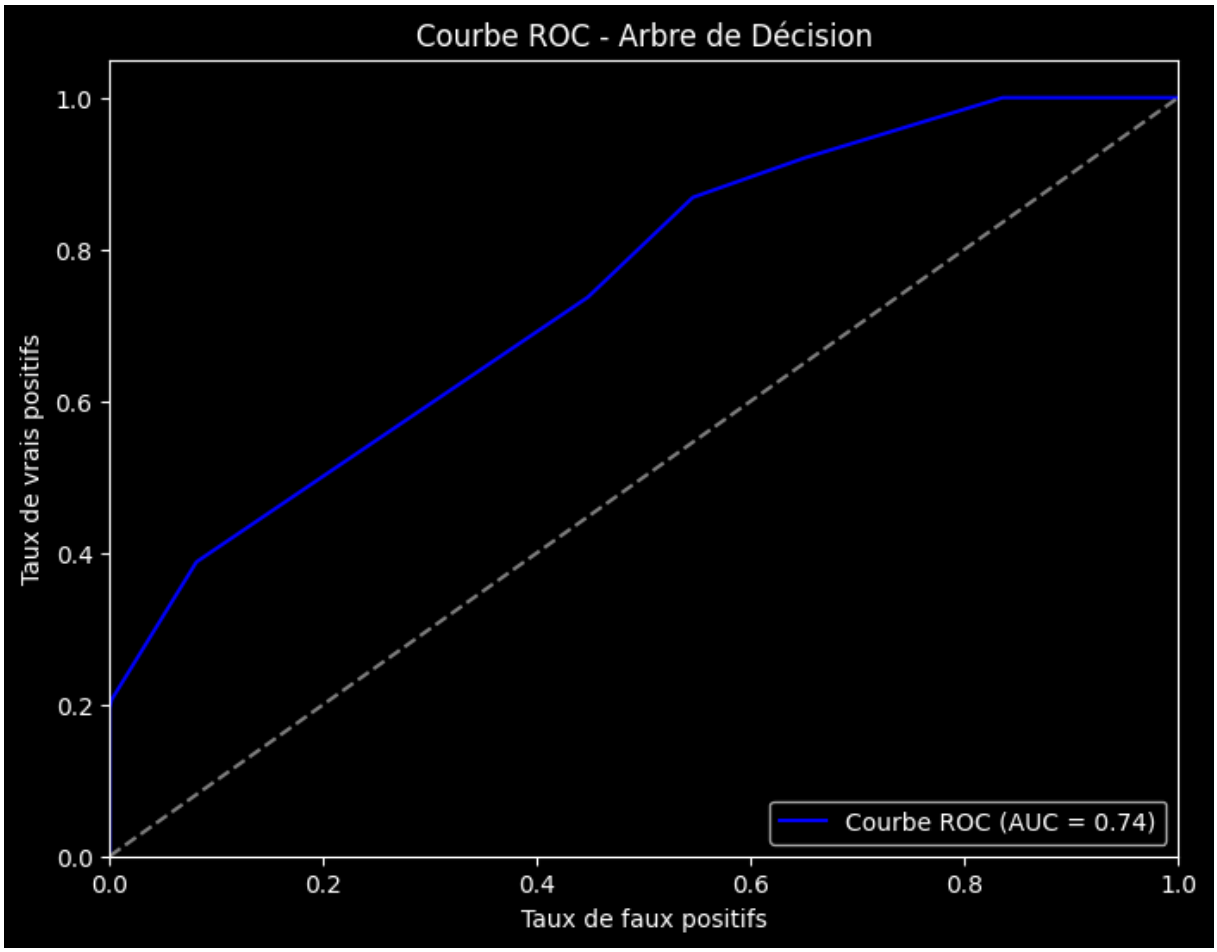


FIGURE 2.5 – Courbe de ROC Arbre de décision.

Annexe 6

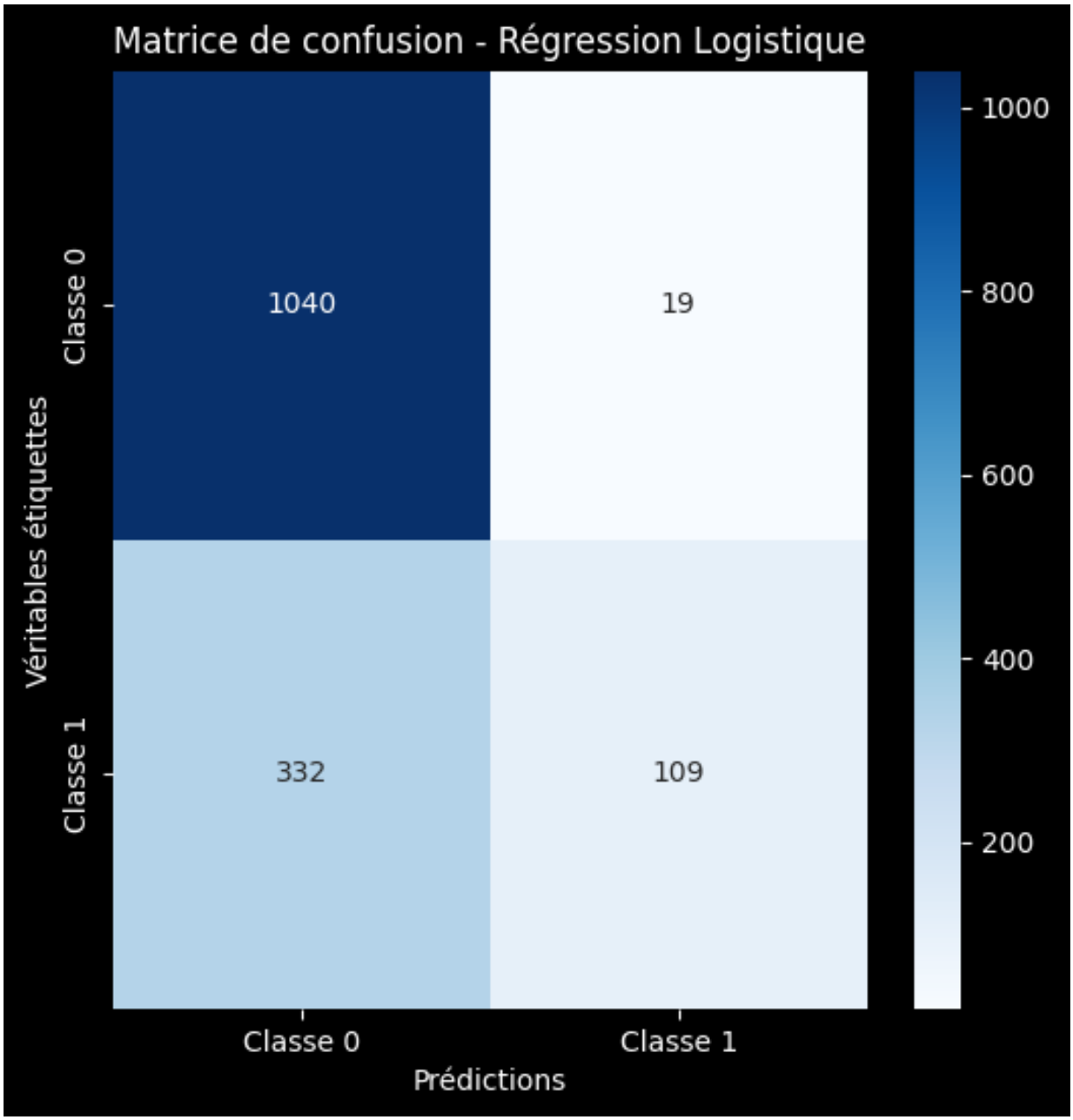


FIGURE 2.6 – Matrice de confusion Régression logistique.

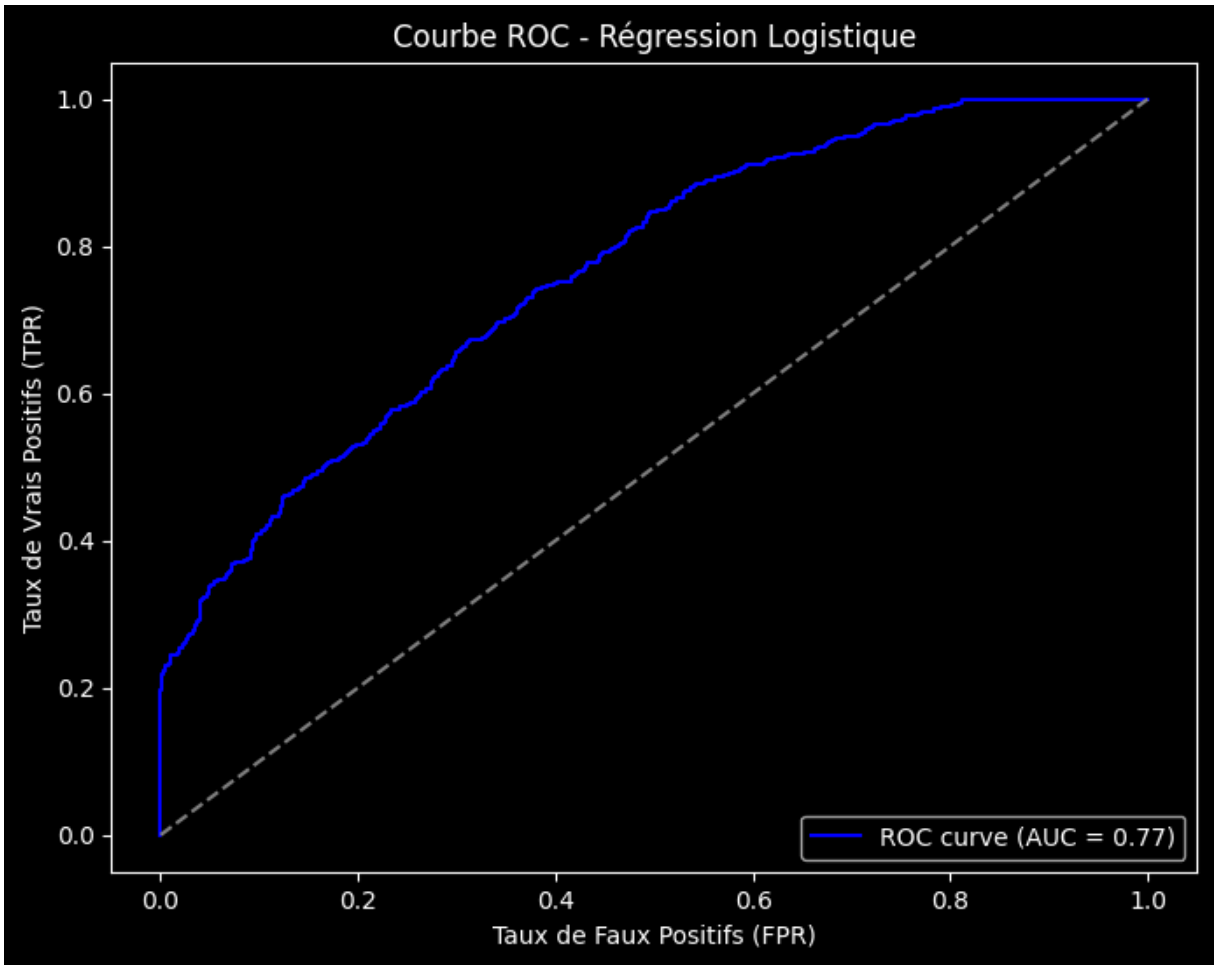


FIGURE 2.7 – Courbe de ROC Régression Logistique.

Annexe 7

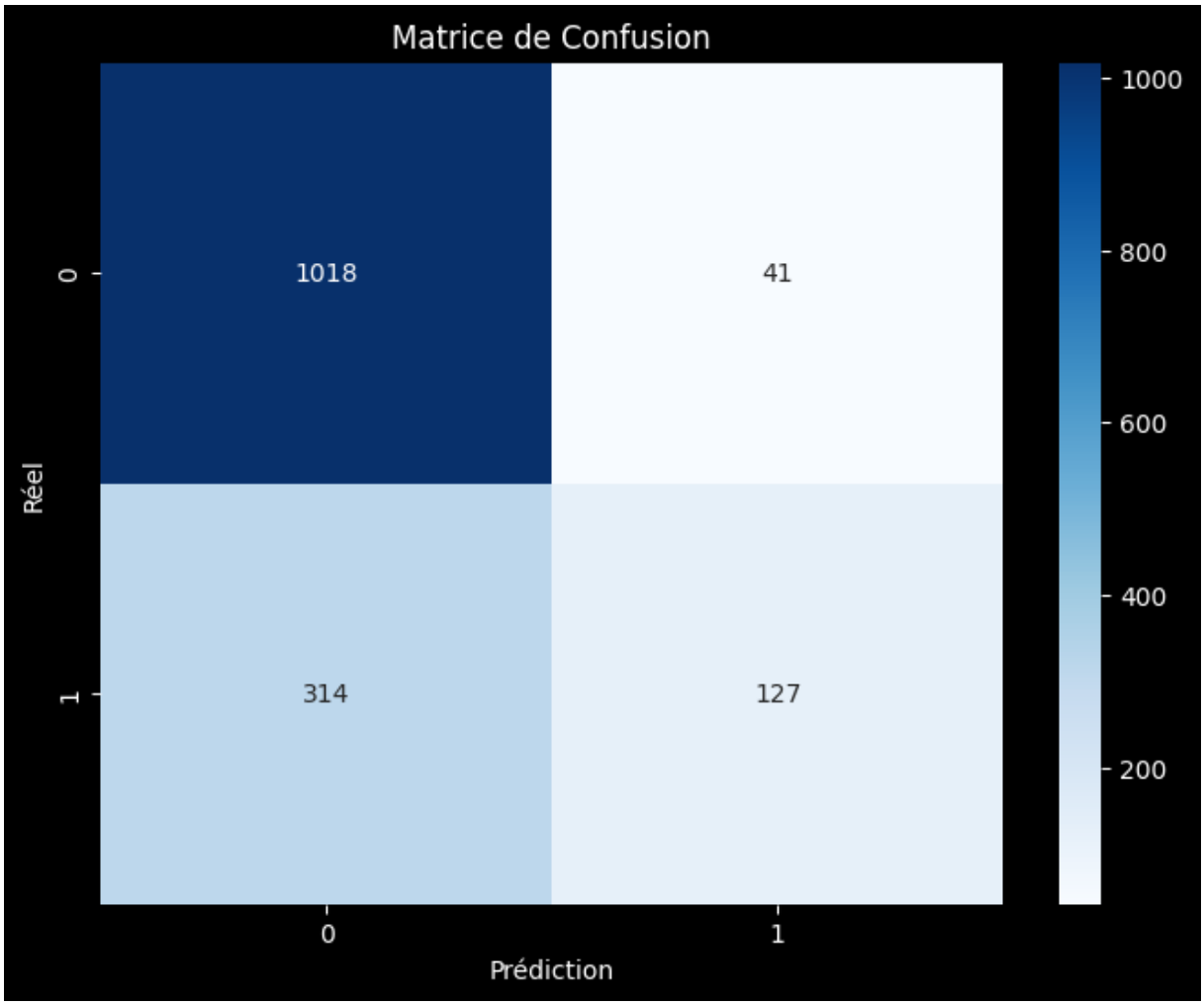


FIGURE 2.8 – Matrice de confusion Forêt Aléatoire.

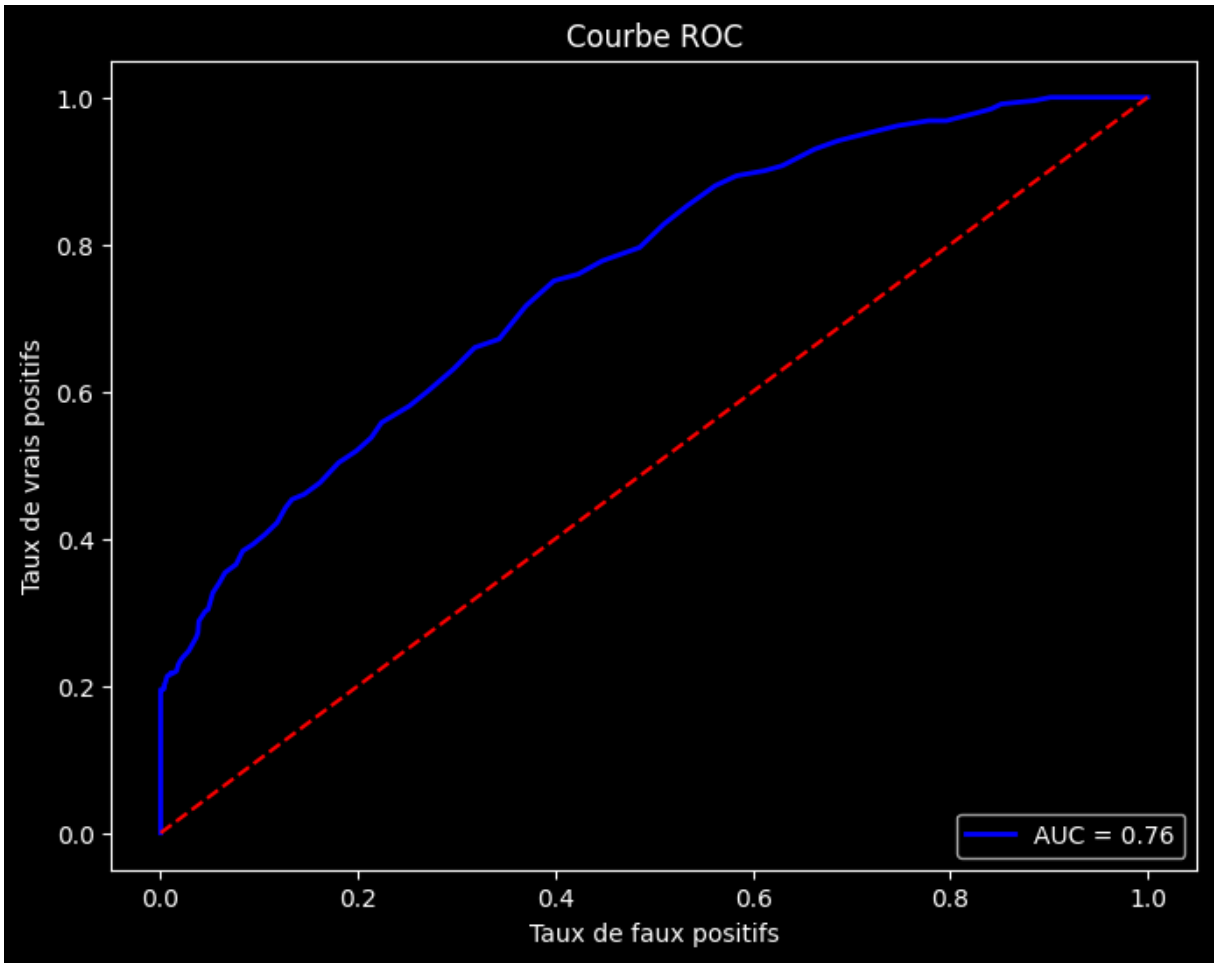


FIGURE 2.9 – Courbe de ROC Forêt Aléatoire

Annexe 8

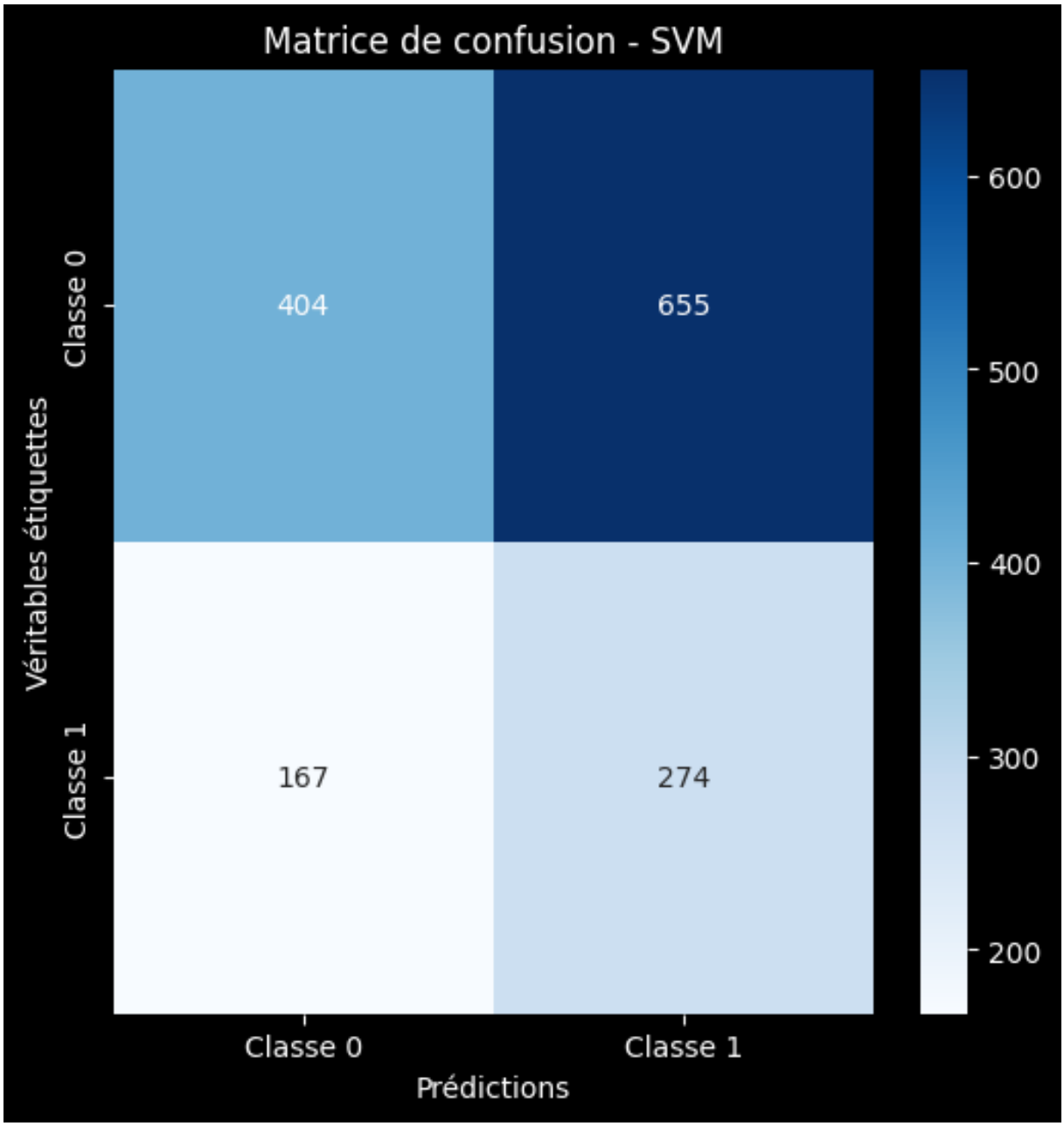


FIGURE 2.10 – Matrice de confusion SVM.

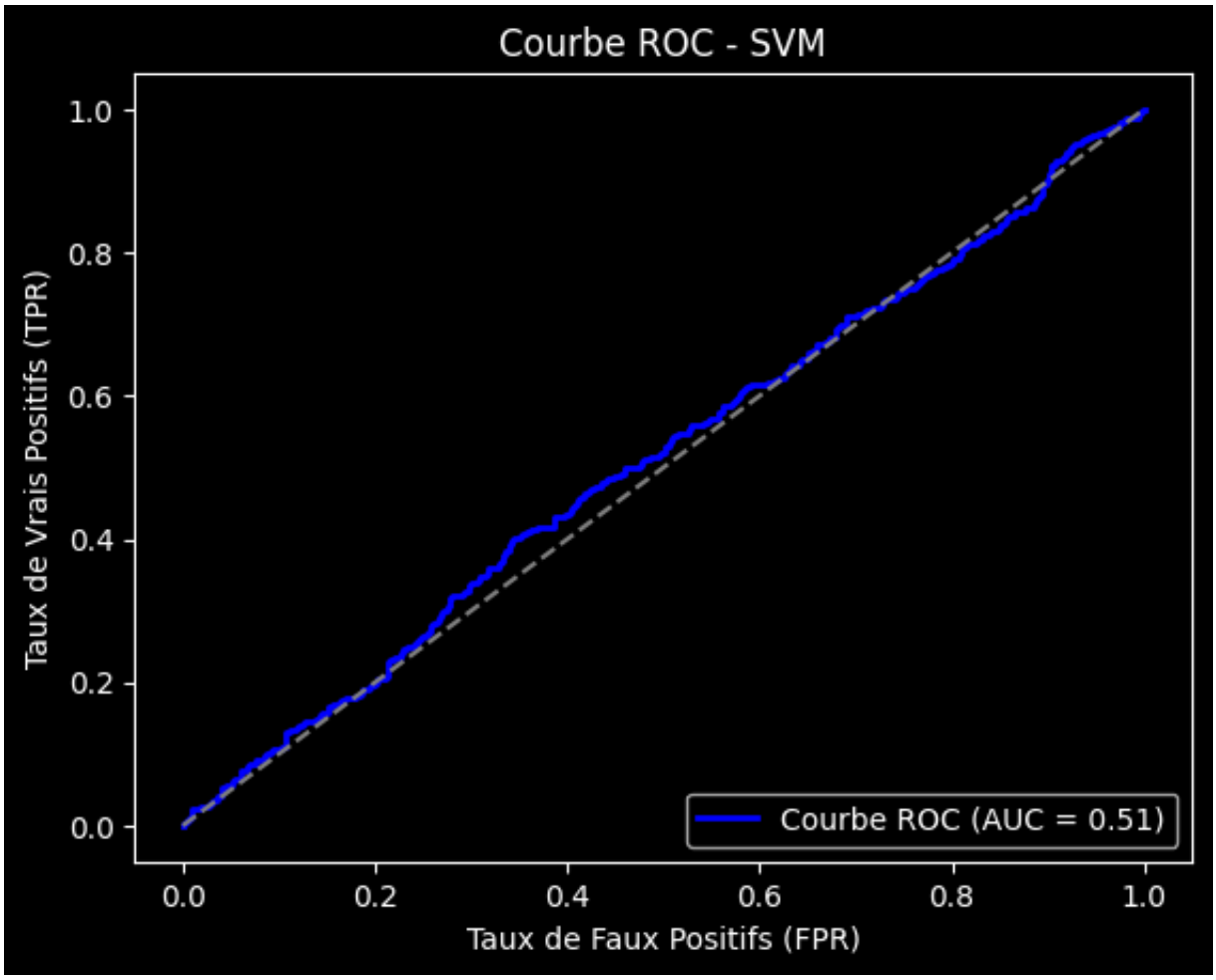


FIGURE 2.11 – Courbe de ROC SVM.

Annexe 9

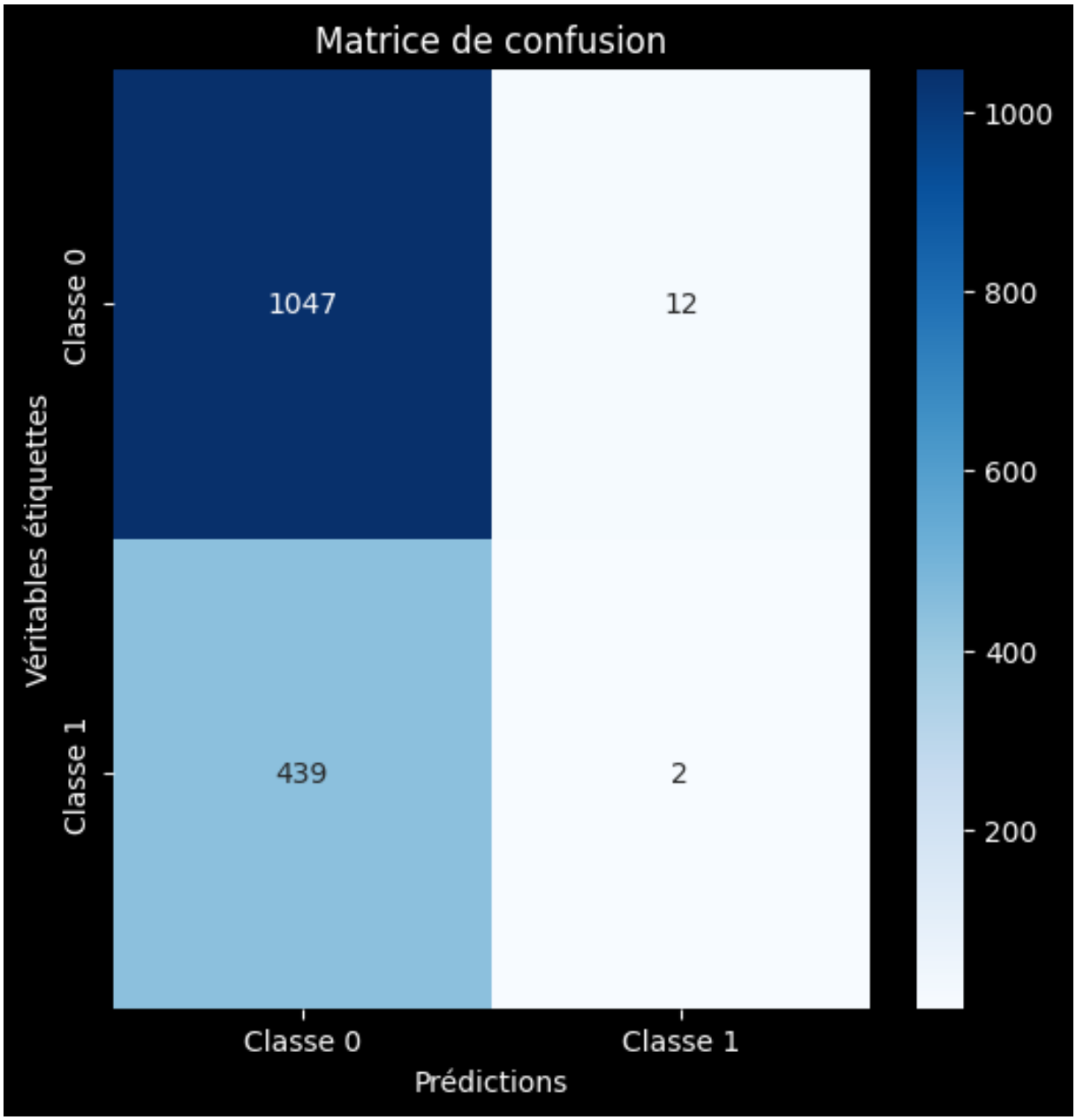


FIGURE 2.12 – Matrice de confusion KNN.

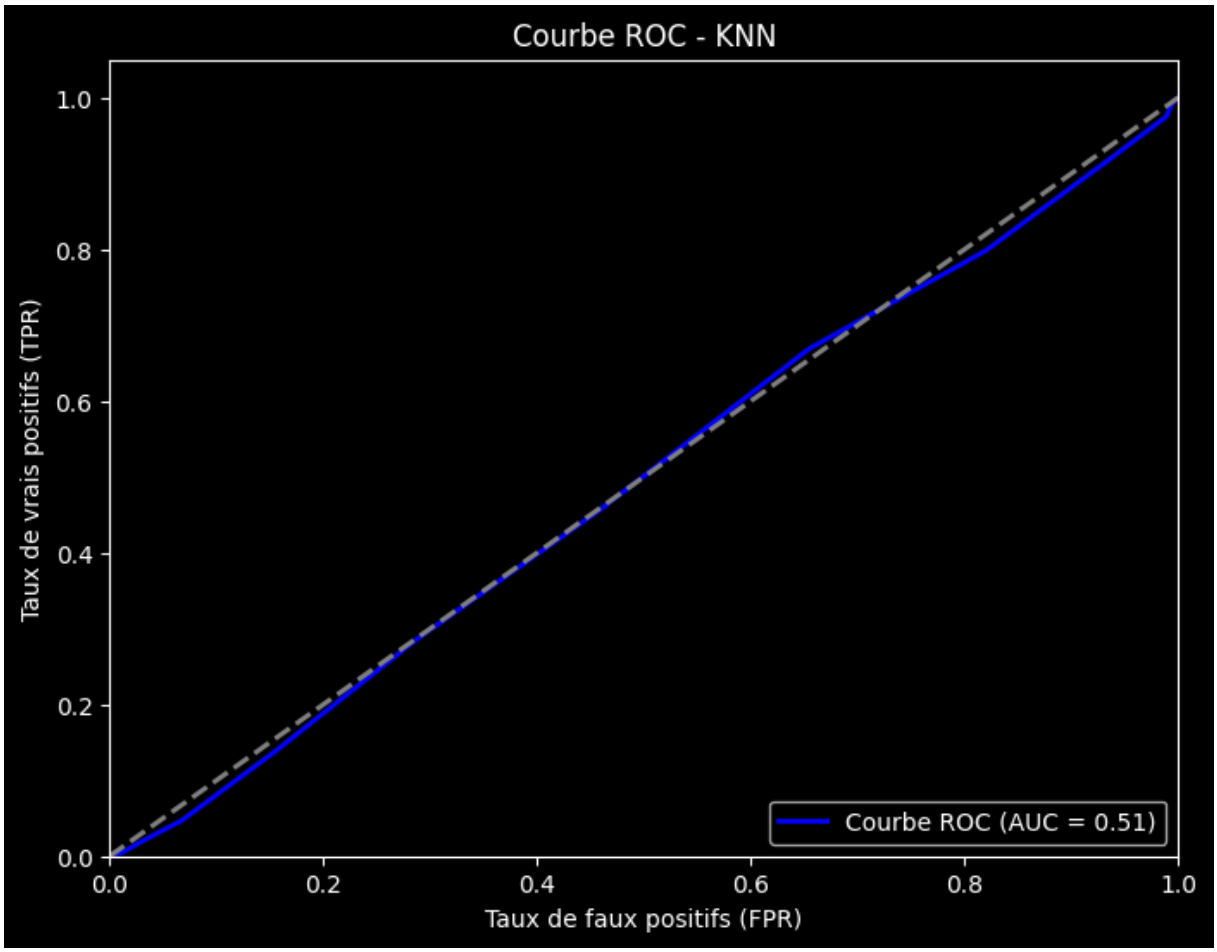


FIGURE 2.13 – Courbe de ROC KNN.

Annexe 10

```
Validation croisée - Arbre de décision: Accuracy moyenne = 0.6908
Validation croisée - Régression Logistique: Accuracy moyenne = 0.6872
Validation croisée - Forêts Aléatoires: Accuracy moyenne = 0.7793
Validation croisée - SVM: Accuracy moyenne = 0.5130
Validation croisée - KNN: Accuracy moyenne = 0.6768

Comparaison des scores de validation croisée:
Arbre de décision: [0.67916667 0.69916667 0.69166667 0.6925     0.69166667] -> Moyenne: 0.6908
Régression Logistique: [0.6825     0.6875     0.68416667 0.68083333 0.70083333] -> Moyenne: 0.6872
Forêts Aléatoires: [0.77833333 0.78083333 0.775      0.78166667 0.78083333] -> Moyenne: 0.7793
SVM: [0.585      0.43083333 0.53583333 0.46083333 0.5525    ] -> Moyenne: 0.5130
KNN: [0.67416667 0.67       0.68666667 0.68666667 0.66666667] -> Moyenne: 0.6768

Évaluation sur le test set pour le meilleur modèle:
Meilleur modèle: Forêts Aléatoires
Accuracy: 0.7613333333333333
              precision    recall  f1-score   support

           0       0.76      0.97      0.85      1059
           1       0.78      0.26      0.39       441

    accuracy                           0.76      1500
   macro avg       0.77      0.62      0.62      1500
weighted avg       0.77      0.76      0.72      1500
```

FIGURE 2.14 – Métrique Validation Forêt Aléatoire.

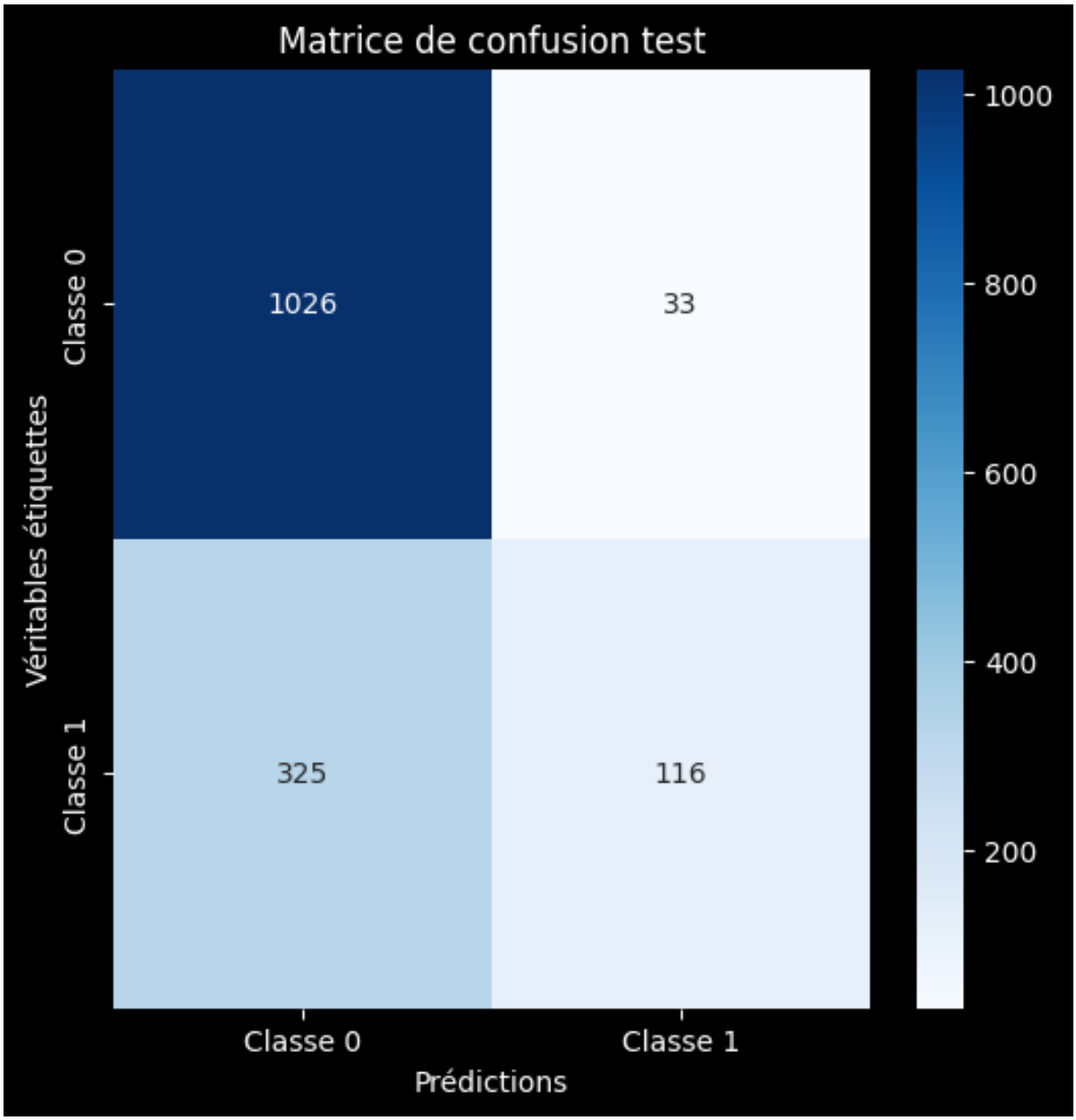


FIGURE 2.15 – Matrice de validation croisée Forêt Aléatoire.

Annexe 11

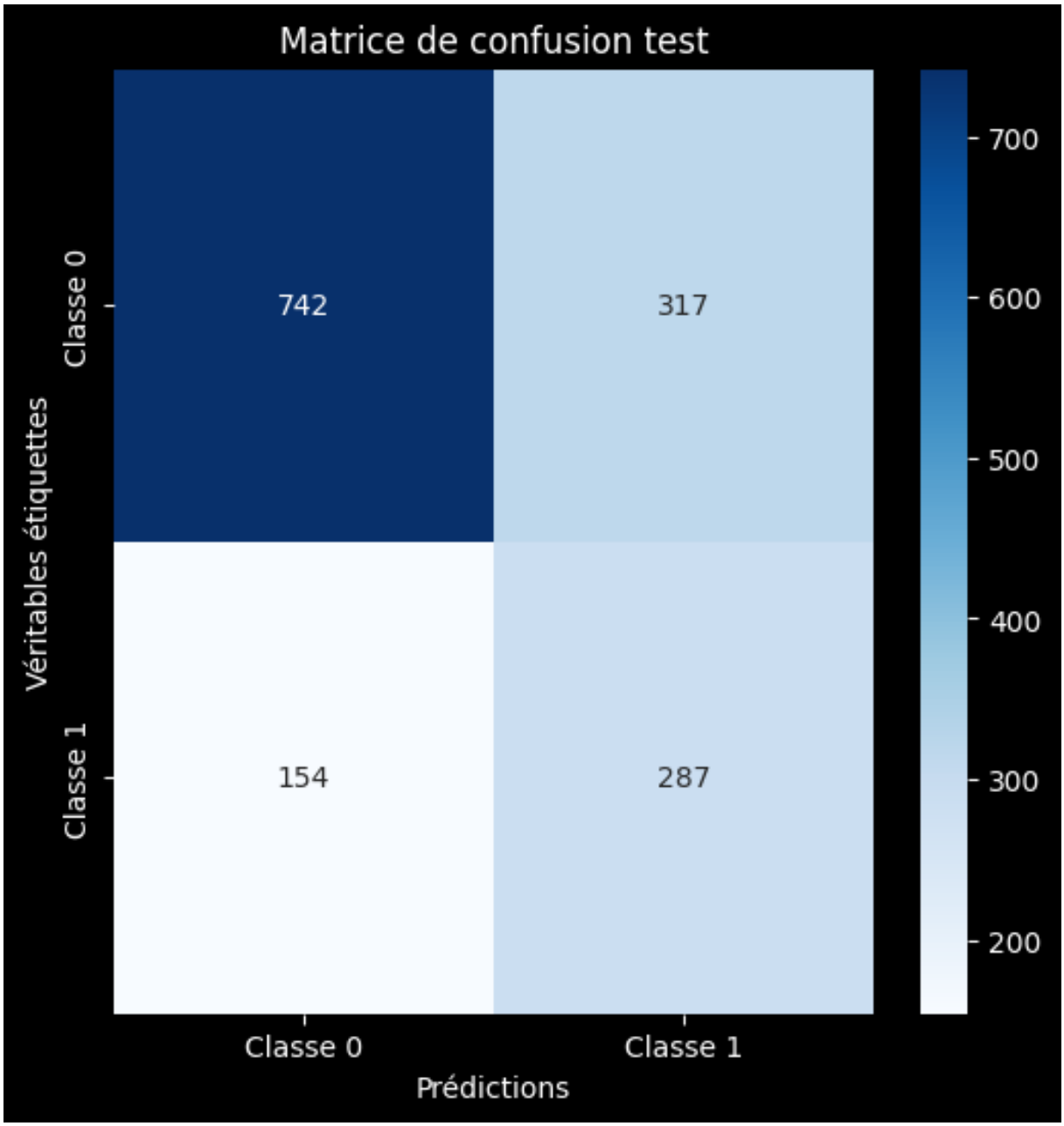


FIGURE 2.16 – Matrice de validation croisée Régression Logistique.

Bibliographie

[1] Chloé-Agathe Azencott. *Introduction au Machine Learning*. Dunod, 2020.

[2] Marine Demangeot. *Classification supervisée et non supervisée*. Ellipses, 2022.

[3] VITOFFODJI Adjimon Jerome. Loan default prediction, 2025. URL : `https://github.com/JeromeVitoff/Loan-Default-Prediction`.

[4] SERGEI SHIRKIN. Loan-default-prediction compétition, 2019. URL : `https://www.kaggle.com/competitions/credit-default-prediction-ai-big-data/overview`.