Rapport - Projet Techniques d'Apprentissage Artificiel

Détection de fraude sur les transactions de carte bancaire – approche par apprentissage supervisé

Nom : DU
Prenom : Qian

1. Introduction

Dans le cadre du module Techniques d'Apprentissage Artificiel, j'ai choisi d'étudier un problème de classification supervisée appliqué à un cas réel : la détection de fraude sur les transactions de carte bancaire.
Ce choix n'est pas motivé uniquement par l'importance du sujet dans le domaine financier, mais surtout par mon intérêt personnel pour l'analyse de données et par le fait que je travaille en parallèle sur le même dataset dans un autre projet, dans le cours *Recherches en Big Data*, où j'applique un modèle de type CNN.

Ainsi, utiliser le même jeu de données dans deux contextes différents offre une opportunité particulièrement intéressante :

- Comparer les performances et les approches entre des méthodes classiques d'apprentissage supervisé et un modèle de deep learning.

Cette comparaison permet de mieux comprendre les forces et limites de chaque famille de modèles, et d'analyser comment ils se comportent face à un dataset réel et déséquilibré.

L'objectif principal de ce projet est donc :

- d'appliquer plusieurs algorithmes supervisés vus en cours, notamment CART, KNN et Random Forest,
- d'évaluer leurs performances à l'aide de métriques adaptées,
- d'comparer avec le modèle CNN développé dans l'autre projet.

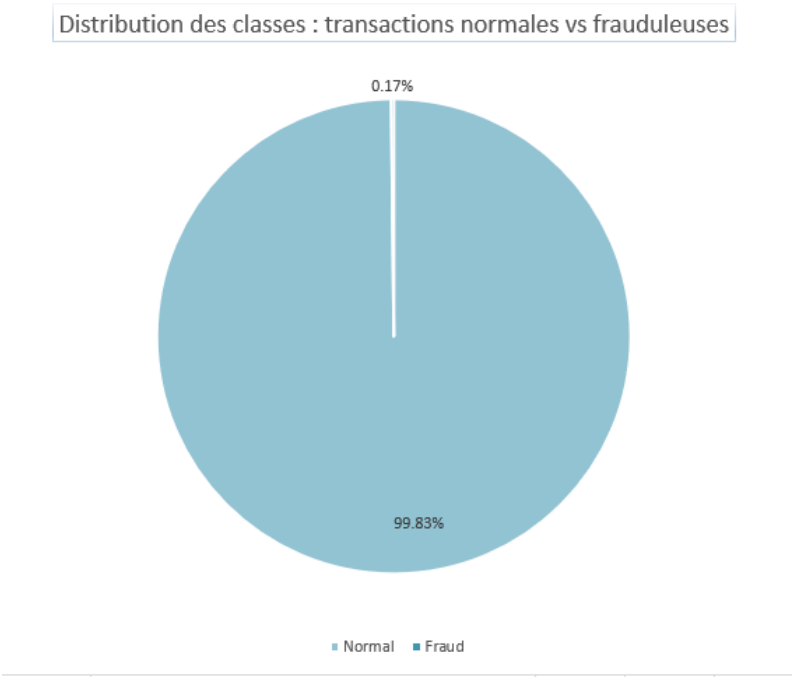
2. Description des données

Le dataset utilisé dans ce projet provient de la plateforme Kaggle, sous le nom Credit Card Fraud Detection(lien: https://www.kaggle.com/datasets/mlg-ulb/creditcardfraud). Il regroupe 284 807 transactions bancaires réalisées par des clients européens sur une période de deux jours en Septembre 2013.
Chaque transaction est décrite par 30 variables numériques, auxquelles s'ajoute une variable cible indiquant si la transaction est normale (Class = 0) ou frauduleuse (Class = 1).

Le dataset ne contient que des variables numériques, résultant d'une transformation par Analyse en Composantes Principales (PCA).
En raison de contraintes de confidentialité, les caractéristiques originales ainsi que des informations plus détaillées sur la nature des données ne sont pas disponibles. Les variables V1, V2, … V28 correspondent ainsi aux composantes principales obtenues après l'application de la PCA.

Apres observation, la variable cible Class est fortement déséquilibrée :

- 0 (transaction normale) : 284 315 observations (~99,83 %)
- 1 (transaction frauduleuse) : 492 observations (~0,17 %)



Ce déséquilibre extrême constitue un défi majeur pour les modèles d'apprentissage supervisé. Pour éviter que le modèle n'apprenne simplement à prédire la classe majoritaire, il est nécessaire d'appliquer des stratégies d'équilibrage et, dans certains cas, d'augmenter le poids attribué à la classe minoritaire (Class 1). Cela permet au modèle de ne pas perdre en précision sur les cas de fraude et de réellement apprendre à distinguer une transaction frauduleuse d'une transaction normale.

Par ailleurs, l'anonymisation par PCA due aux contraintes de confidentialité entraîne plusieurs limitations :

- l'impossibilité d'interpréter directement les variables V1–V28,
- une structure de corrélation modifiée, qui ne reflète plus nécessairement les relations entre les caractéristiques originales.

Ces transformations rendent le dataset très utile pour l'expérimentation technique, mais soulèvent également une question importante : la performance observée sur ce jeu de données peut-elle être directement transposée à un environnement réel ?
En l'absence des caractéristiques d'origine, il reste difficile d'évaluer avec certitude la capacité du modèle à généraliser sur des données opérationnelles contenant des variables non anonymisées.

3. Pretraitement et Methodologie

Dans ce projet, trois modèles supervisés — CART, K-Nearest Neighbors (KNN) et Random Forest (RF) — ont été sélectionnés afin de mener une approche expérimentale comparative entre des méthodes d'apprentissage aux logiques différentes.
Ces trois modèles ont été choisis car ils représentent des familles complémentaires :

- CART offre une structure simple et très interprétable, permettant de comprendre clairement les règles qui mènent à la détection d'une fraude.
- KNN, en tant que méthode non paramétrique, repose uniquement sur la similarité entre observations et permet d'évaluer si les transactions frauduleuses présentent des comportements proches dans l'espace des features.
- Random Forest combine plusieurs arbres via l'agrégation, apportant une robustesse élevée, une meilleure généralisation et une performance supérieure sur des données déséquilibrées.

La combinaison de ces trois approches permet ainsi d'obtenir une vision expérimentale complète et de comparer l'impact de modèles simples, non paramétriques et ensemble learning dans la détection de la fraude.

Afin de mieux évaluer l'efficacité des trois modèles d'apprentissage supervisé — CART, KNN et Random Forest — j'ai choisi d'appliquer différentes stratégies de préparation des données sur le même jeu de données.
L'objectif est d'observer dans quelle mesure les performances des modèles varient lorsque les données sont normalisées, transformées ou conservées dans leur forme initiale.

Ainsi, quatre versions du dataset ont été construites :

Données originales (Original)
→ aucune normalisation appliquée, permettant de mesurer les performances "brutes" des modèles.

Standardisation (StandardScaler)
→ centrage et réduction des variables, particulièrement utile pour les modèles sensibles à l'échelle, comme KNN.

Normalisation MinMax (MinMaxScaler)
→ transformation des valeurs dans l'intervalle [0, 1], afin d'observer l'effet d'une échelle uniformisée.

PCA réduite à 10 composantes (PCA_10)
→ application d'une PCA supplémentaire après standardisation, dans le but d'analyser l'impact d'une réduction de dimension supplémentaire, malgré le fait que les données initiales proviennent déjà d'une transformation PCA.

Grâce à ces quatre configurations, il devient possible d'évaluer de manière systématique la sensibilité de chaque modèle aux différentes formes de préparation

des données, et d'identifier les conditions dans lesquelles leurs performances sont les plus optimales.

L'ensemble de ces transformations est implémenté dans le fichier prepare.py, qui définit un pipeline clair et reproductible pour la préparation des données.
Ce pipeline comprend :

Lecture des données (Read) : chargement du fichier CSV et séparation entre les variables explicatives (X) et la variable cible (Y).
Division du dataset (split_data) : découpage en jeux d'entraînement et de test (85 % / 15 %) avec stratification afin de conserver la proportion très faible de la classe frauduleuse.
Normalisation Standard (standard_scaler) : application du StandardScaler pour normaliser les données;
Normalisation MinMax (minmax_scaler) : application du MinMaxScaler pour normaliser les données;
PCA à 10 composantes (apply_pca) : réduction supplémentaire de dimension pour analyser son effet sur les performances, malgré la présence d'une PCA initiale dans le dataset.

Ce pipeline permet de générer de manière cohérente les quatre versions du dataset utilisées pour la comparaison des modèles, tout en garantissant une reproductibilité totale des expériences.

4. Evaluation des modèles

4.1 Construction des modèles

Chaque modèle est évalué sur les quatre versions du dataset :

- Original
- StandardScaler
- MinMaxScaler
- PCA_10

Pour garantir une comparaison équitable :

- même découpage train/test
- mêmes métriques,
- même protocole d'entraînement,
- mêmes visualisations générées automatiquement.

Cette stratégie permet d'identifier :

- l'impact réel du prétraitement,
- la sensibilité de chaque modèle à l'échelle des données,
- et la robustesse globale de chaque approche.

Dans le contexte d'un dataset fortement déséquilibré, plusieurs techniques ont été appliquées afin d'améliorer l'apprentissage et d'éviter que les modèles ne se concentrent uniquement sur la classe majoritaire.

- CART et Random Forest :
  Ces deux modèles utilisent l'option class_weight='balanced', qui compense automatiquement le déséquilibre du dataset en ajustant les poids des classes lors de l'entraînement.
- KNN :
  Étant très sensible au déséquilibre des classes, ce modèle nécessite un suréchantillonnage de la classe minoritaire.
  Ainsi, la méthode SMOTE est appliquée via la fonction apply_smote() avant l'entraînement, ce qui permet de construire un ensemble d'apprentissage plus équilibré et d'améliorer la capacité du modèle à détecter les fraudes..

4.2 Métriques de performance

L'évaluation des performances constitue une étape essentielle du projet, notamment dans le contexte d'un dataset fortement déséquilibré comme celui utilisé pour la détection de fraude.
Afin de mesurer précisément la capacité des modèles à identifier la classe frauduleuse, plusieurs métriques complémentaires ont été utilisées.
L'ensemble de la procédure d'évaluation est implémenté dans le fichier evaluation.py, via la fonction compute_metrics().

- Accuracy - Proportion globale de prédictions correctes.
  → Peu pertinente ici, car un modèle qui prédit toujours 0 atteindrait déjà presque 99,8 %.

- Precision (Class 1 => Fraude)
  Proportion de transactions réellement frauduleuses parmi celles prédites comme fraude.
  → Utile pour limiter les faux positifs.

- Recall (Class 1 => Fraude)
  Proportion de fraudes correctement identifiées parmi l'ensemble des fraudes réelles.
  → Indicateur crucial dans notre contexte, car manquer une fraude peut être très coûteux.

- F1-score
  Moyenne harmonique entre précision et rappel.
  → Bien adapté lorsque les classes sont déséquilibrées.

- ROC AUC
  Aire sous la courbe ROC, qui mesure la capacité du modèle à séparer les deux classes.

- PR AUC (Average Precision Score)
  Performance sur la courbe Precision–Recall.
  → Beaucoup plus informative que ROC AUC dans un dataset hautement déséquilibré.

4.3 Visualisation

Pour chaque modèle des visualisations sont générées automatiquement :

- Courbes ROC, via la fonction *plot_roc_pr_curves()*
  → Affichent le compromis entre taux de vrais positifs (TPR) et taux de faux positifs (FPR).

- Courbes Precision–Recall (PR), via la fonction *plot_roc_pr_curves()*
  → Particulièrement pertinentes pour la détection de fraude, car elles montrent directement l'équilibre entre rappel et précision sur la classe minoritaire.

- Matrices de confusion, via la fonction *plot_confusion_matrix()*
  TP (True Positives) : fraudes correctement détectées
  FN (False Negatives) : fraudes manquées
  FP (False Positives) : transactions normales classées à tort comme fraude
  TN (True Negatives) : prédictions correctes de transactions normales
  → Mesure est essentielle pour comprendre si un modèle détecte peu de fraudes (faible recall) / génère trop de faux positifs (faible précision) / atteint un équilibre satisfaisant entre les deux.

5. Resultat

5.1 Tableau global des scores

Scaler	Model	Accuracy	Precision	Recall	F1	ROC_AUC	PR_AUC
Original	CART	99.90%	71.62%	71.62%	71.62%	85.79%	51.35%
Original	KNN	94.95%	1.81%	52.70%	3.49%	76.90%	3.60%
Original	RF	99.95%	93.22%	74.32%	82.71%	93.09%	79.97%
Standard	CART	99.89%	69.33%	70.27%	69.80%	85.11%	48.77%
Standard	KNN	99.78%	43.15%	85.14%	57.27%	93.18%	55.88%
Standard	RF	99.95%	94.83%	74.32%	83.33%	93.09%	79.20%
MinMax	CART	99.62%	20.41%	40.54%	27.15%	70.13%	8.38%
MinMax	KNN	99.87%	61.45%	68.92%	64.97%	85.78%	58.38%
MinMax	RF	99.83%	100.00%	2.70%	5.26%	87.48%	54.48%
PCA_10	CART	99.76%	27.87%	22.97%	25.19%	61.43%	6.54%
PCA_10	KNN	99.31%	15.24%	64.86%	24.68%	84.19%	20.04%
PCA_10	RF	99.87%	95.00%	25.68%	40.43%	91.16%	51.74%

Ce tableau présente l'ensemble des performances pour les 12 combinaisons.

5.2 Impact du prétraitement : tendances générales

L'analyse comparative révèle plusieurs régularités importantes :

1. Les modèles CART, RF sont peu sensibles à l'échelle des variables
   Très petites variations entre Original / Standard
   → Ce comportement confirme que les arbres reposent sur des seuils, non sur des distances.

2. La normalisation influence fortement les modèles basés sur la distance
   KNN devient performant uniquement après StandardScaler
   MinMax améliore mais reste insuffisant
   → La cohérence des distances est essentielle pour ce modèle.

3. La PCA_10 dégrade systématiquement les performances
   Perte d'information lors d'une réduction supplémentaire sur un dataset déjà PCA
   F1-score fortement diminué pour tous les modèles
   → PCA_10 n'est pas adaptée dans ce contexte.

5.3 Analyse des modèles

1. Random Forest :
   Le modèle le plus performant et le plus robuste
   Présente les résultats les plus élevés et les plus stables dans la majorité des configurations.
   Les versions Original et Standard offrent de meilleurs niveaux de recall, F1-score et AUC, avec très peu de variation entre les deux prétraitements.
   Le modèle se montre globalement peu sensible à l'échelle des données, ce qui confirme la robustesse des méthodes ensemblistes basées sur des arbres.
   La version MinMax entraîne une baisse notable du recall, tandis que PCA_10 réduit davantage les performances en raison de la perte d'information.
   Malgré cela, Random Forest reste le modèle globalement le plus fiable et le plus performant.

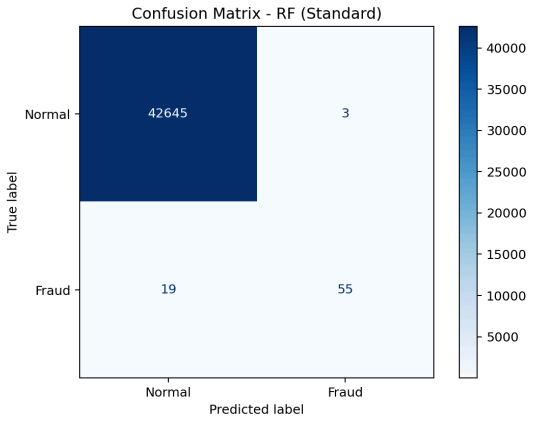
2. CART :
   Un modèle stable mais moins performant
   Le modèle CART présente des performances cohérentes, notamment dans les versions Original et Standard, où les scores restent proches.
   Sa sensibilité aux différents scalers est relativement faible, ce qui reflète la nature des modèles arborescents.
   Cependant, les résultats restent inférieurs à ceux de Random Forest, et les versions MinMax et PCA_10 entraînent une dégradation significative du F1-score et du rappel.
   CART constitue ainsi une base stable, mais ses capacités restent limitées face à des données complexes.

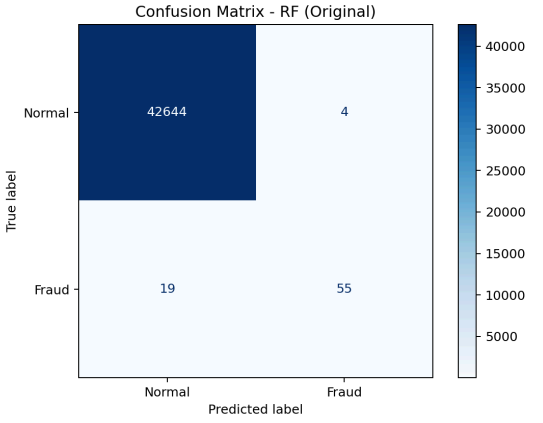
3. KNN :
   Le modèle le plus sensible au prétraitement
   KNN est le modèle dont les performances varient le plus fortement selon la préparation des données.
   La version Original montre des résultats très faibles avec un F1-score quasi nul, en raison de l'absence de normalisation.

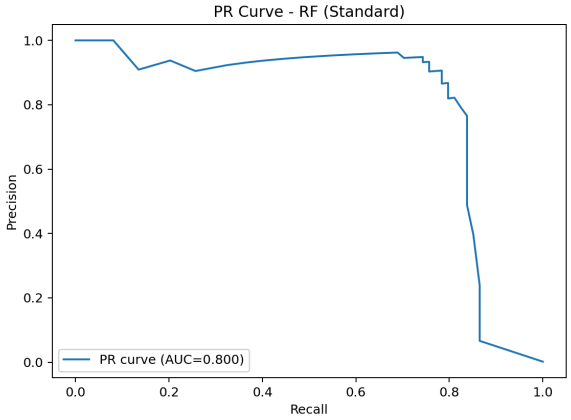
L’application de StandardScaler transforme complètement le comportement du modèle : le rappel atteint 85 %, l’un des meilleurs scores de tous les modèles et prétraitements combinés.
MinMax améliore partiellement les résultats, mais la stabilité n’est pas garantée.
La version PCA_10 entraîne à nouveau une baisse significative des performances.
KNN apparaît ainsi comme le modèle le plus instable, fortement dépendant de la mise à l’échelle des variables.

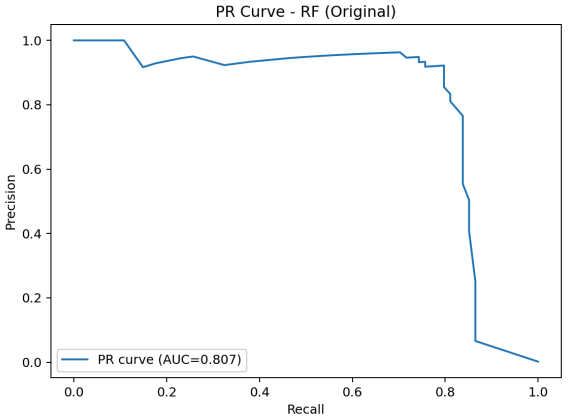
5.4 Visualisation

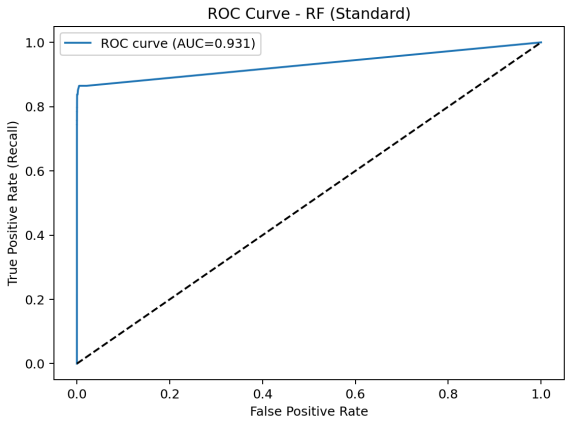
Random Forest avec données originales ou normalisation Standard
Les visualisations des autres modèles sont ajoutées dans les annexes.

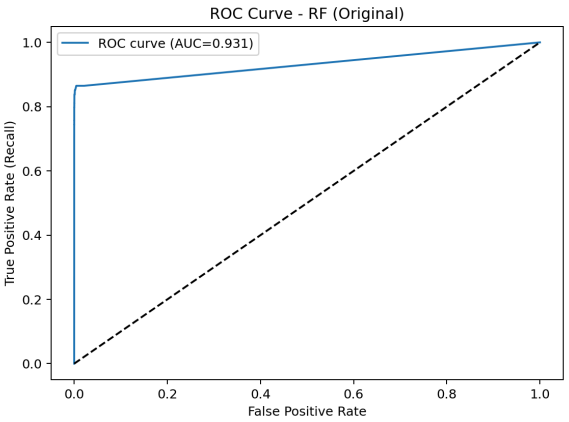












6. Comparaison rapide avec CNN

Afin de compléter l'analyse menée sur les modèles traditionnels, une étude comparative a été réalisée sur quatre variantes d'un modèle Convolutional Neural Network (CNN) développées dans un projet parallèle.
L'objectif est d'évaluer si une architecture plus complexe permet d'améliorer la détection de la classe frauduleuse dans un contexte de données fortement déséquilibrées.

model	accuracy	precision_class1	recall_class1	f1_class1	roc_auc	pr_auc
baseline	99.53%	24.90%	83.78%	38.39%	96.19%	63.96%
simple	99.76%	35.35%	47.30%	40.46%	95.30%	37.34%
deep	99.84%	54.55%	48.65%	51.43%	92.34%	47.71%
dropout	99.67%	32.46%	83.78%	46.79%	96.33%	62.67%

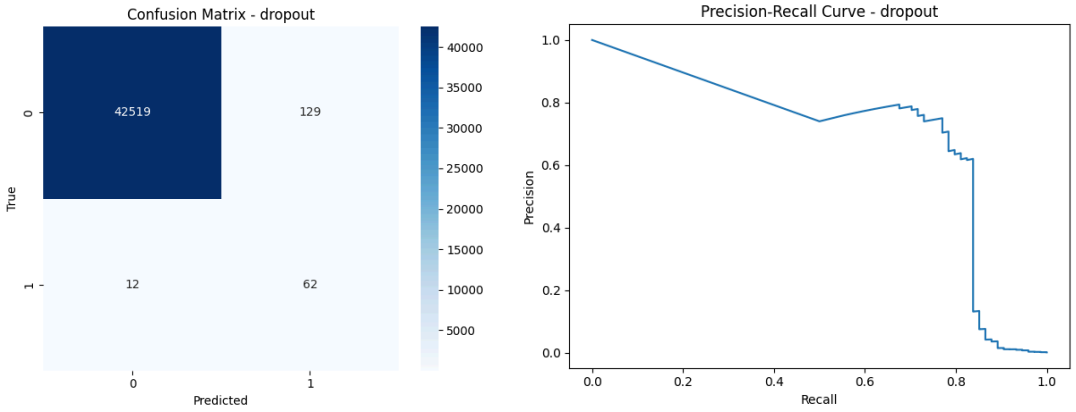
L'analyse des résultats montre que, malgré une accuracy globalement très élevée pour l'ensemble des architectures (≈ 99,7 %), les performances spécifiques à la classe minoritaire diffèrent nettement. Parmi les quatre variantes, le modèle CNN intégrant uniquement du Dropout obtient les meilleurs résultats :
– un recall particulièrement élevé (≈ 84 %), montrant une capacité renforcée à identifier les fraudes ;
– un F1-score supérieur aux autres variantes, indiquant un meilleur équilibre entre précision et rappel ;
– des valeurs de ROC AUC (≈ 0,96) et PR AUC élevées, positionnant cette architecture comme la plus discriminante.

Cette supériorité s'explique par l'effet régularisateur du Dropout, qui limite le sur-apprentissage sur la classe majoritaire et favorise une meilleure généralisation. La matrice de confusion confirme cette tendance, avec une couverture étendue des fraudes au prix d'un nombre accru de faux positifs, situation acceptable dans un contexte où le coût d'un faux négatif est beaucoup plus élevé.

Comparé au meilleur modèle traditionnel du projet (Random Forest), le CNN – Dropout présente un rappel nettement supérieur et une capacité discriminante plus forte, au détriment toutefois d'un coût computationnel plus important et d'une

explicabilité plus faible. Dans un contexte opérationnel, le choix doit donc équilibrer performance, ressources de calcul et exigences métiers.

Enfin, plusieurs travaux de recherche soulignent que les CNN sont mieux adaptés aux données présentant une structure spatiale (images, signaux, séquences temporelles), tandis que les modèles fondés sur les arbres décisionnels conservent un avantage naturel sur les données tabulaires, grâce à leur capacité à capturer des interactions non linéaires et à leur robustesse face au déséquilibre des classes.



7. Conclusion

L'ensemble des analyses menées dans ce projet met en évidence plusieurs enseignements clés concernant la détection de fraude sur un dataset fortement déséquilibré. Les expérimentations montrent que les modèles fondés sur des arbres, en particulier Random Forest, offrent les performances les plus robustes et les plus stables, indépendamment du prétraitement appliqué. Leur capacité à capturer les relations non linéaires, leur faible sensibilité à l'échelle des variables et leur aptitude à gérer efficacement le déséquilibre des classes en font des solutions particulièrement adaptées aux données tabulaires.

Les modèles KNN et CART apportent également des résultats cohérents, mais leurs performances restent inférieures à celles des ensembles en arbres. KNN se distingue par une forte dépendance à la normalisation, tandis que PCA_10 induit une perte d'information qui affecte systématiquement l'ensemble des modèles.

La comparaison rapide avec un modèle CNN confirme qu'une architecture plus complexe peut améliorer la détection de la classe fraud, notamment grâce à un rappel plus élevé et une forte capacité discriminante mesurée par les AUC. Cependant, le coût computationnel plus important et un niveau d'explicabilité réduit limitent son intérêt dans un environnement opérationnel fondé sur des données tabulaires.

Dans l’ensemble, Random Forest apparaît comme la solution la plus fiable pour ce problème de détection de fraude, à la fois en termes de performance, de stabilité et de faisabilité opérationnelle. Le choix final du modèle doit néanmoins intégrer les besoins métiers, en tenant compte des compromis entre performance, coût, et interprétabilité.

De plus, comme le dataset utilisé dans ce projet a déjà été transformé par une PCA, l’interprétation précise des variables reste limitée. Dans un système de détection de fraude réel, les institutions financières ne se reposent pas uniquement sur un modèle, mais sur une combinaison de plusieurs niveaux d’analyse : habitudes de consommation du client, écarts comportementaux, montants et fréquences de transactions (statistical signals), ainsi que des règles métier (rule-based system) telles que les transactions répétées sur un intervalle court, les opérations transfrontalières ou les activités nocturnes.

Dans ces architectures hybrides, les signaux statistiques et les règles métier assurent un premier filtrage, tandis que les modèles de machine learning évaluent le niveau de risque de manière plus fine.

Ainsi, si le modèle développé dans ce projet pouvait être enrichi par des signaux métier ou des caractéristiques issues des données brutes (non réduites par PCA), ses performances réelles pourraient encore être renforcées dans un contexte opérationnel.

8. Annexes

CART

Original

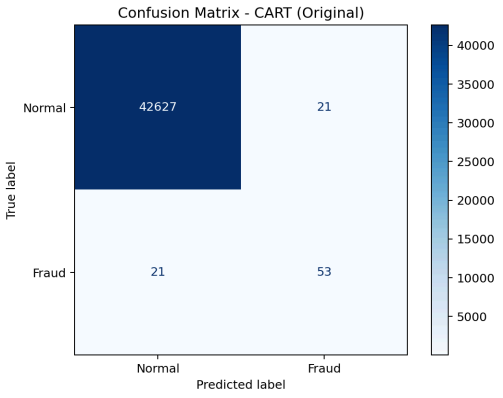


Figure C1 – Matrice de confusion : CART (Original)

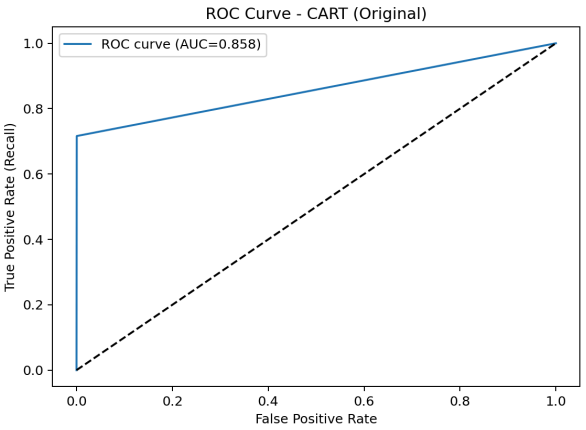


Figure C2 – Courbe ROC : CART (Original)

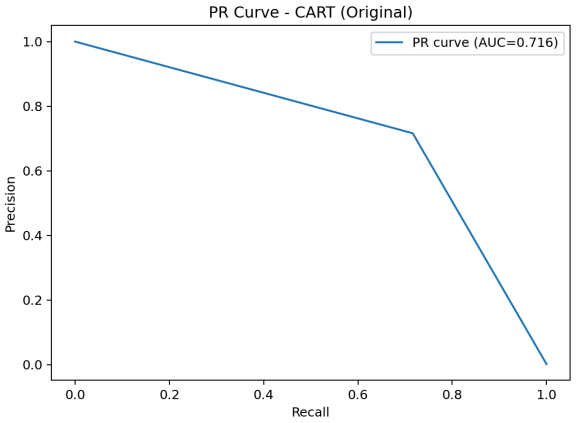


Figure C3 – Courbe PR : CART (Original)

Standard

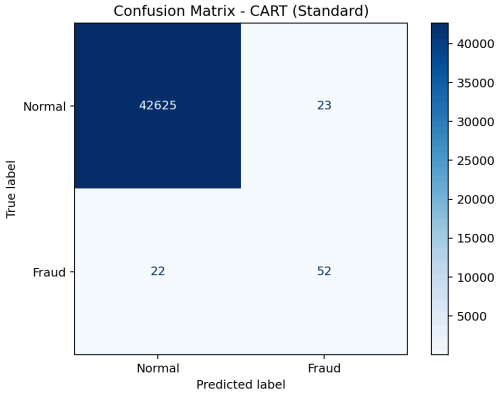


Figure C4 – Matrice de confusion : CART (Standard)

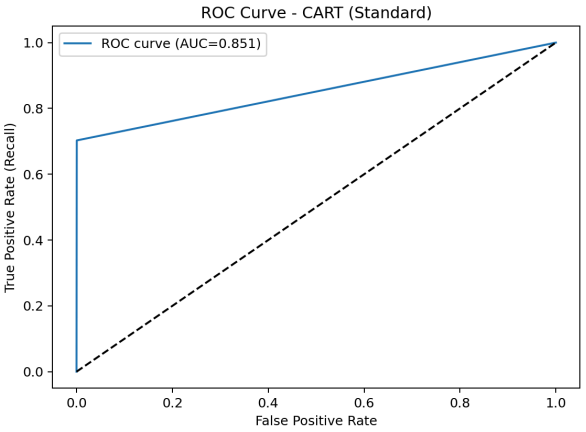


Figure C5 – Courbe ROC : CART (Standard)

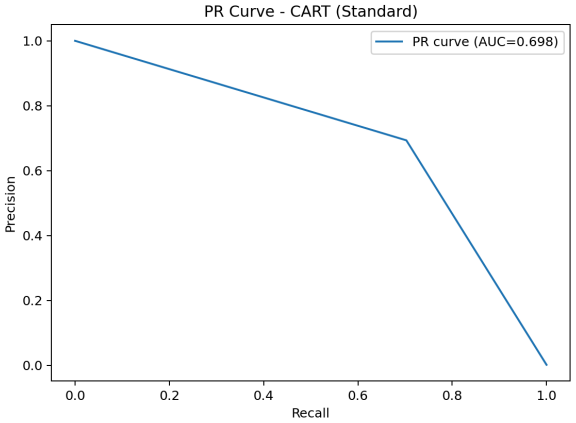


Figure C6 – Courbe PR : CART (Standard)

MinMax

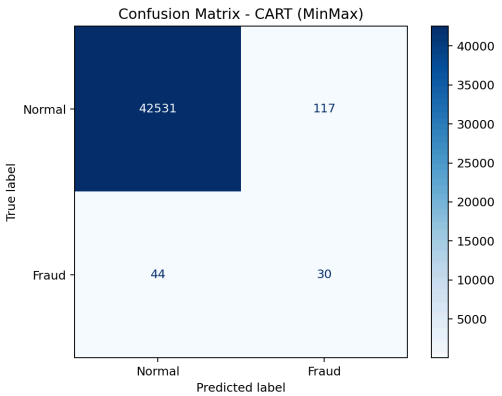


Figure C7 – Matrice de confusion : CART (MinMax)

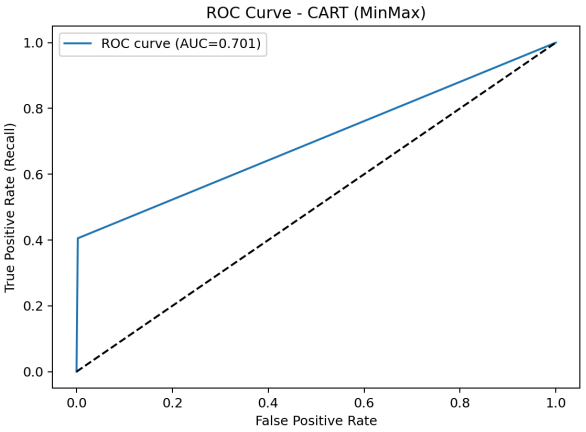


Figure C8 – Courbe ROC : CART (MinMax)

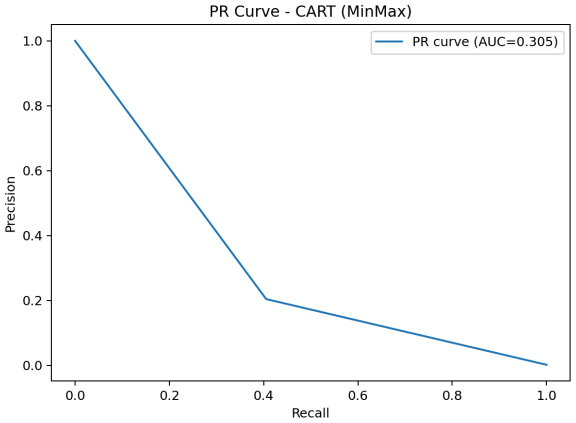


Figure C9 – Courbe PR : CART (MinMax)

PCA_10

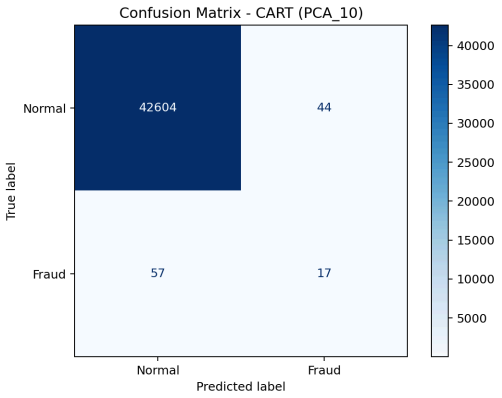


Figure C10 – Matrice de confusion : CART (PCA_10)

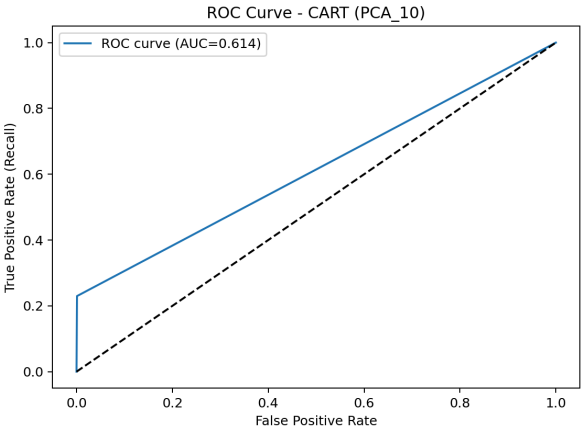


Figure C11 – Courbe ROC : CART (PCA_10)

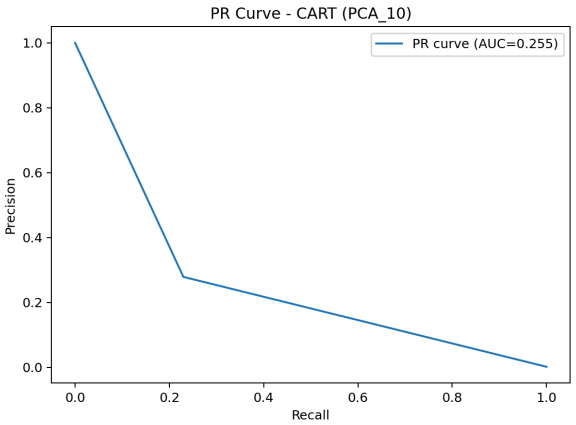


Figure C12 – Courbe PR : CART (PCA_10)

KNN

Original

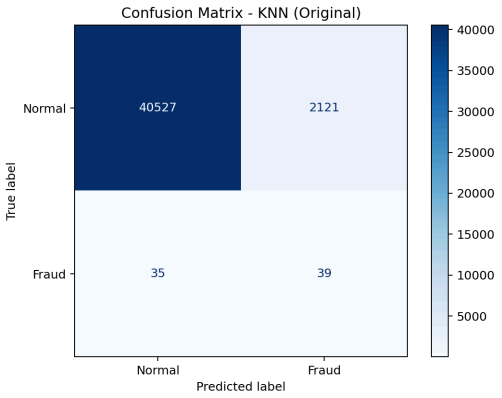


Figure K1 – Matrice de confusion : KNN (Original)

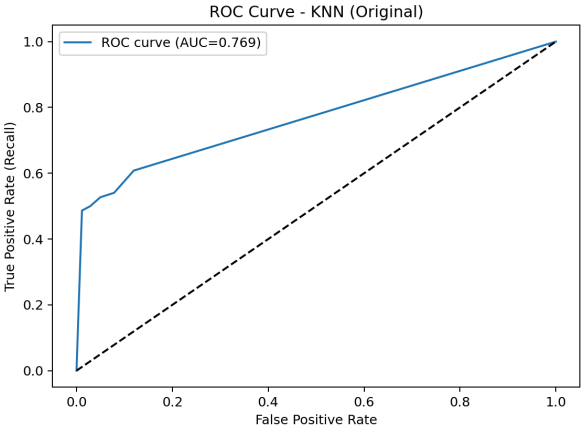


Figure K2 – Courbe ROC : KNN (Original)

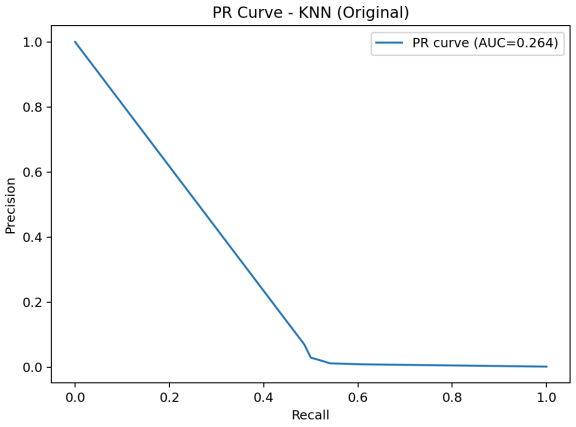


Figure K3 – Courbe PR : KNN (Original)

Standard

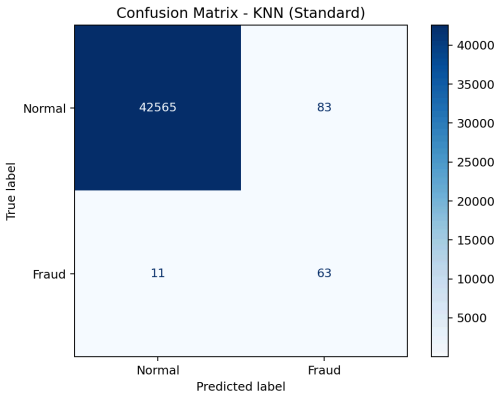


Figure K4 – Matrice de confusion : KNN (Standard)

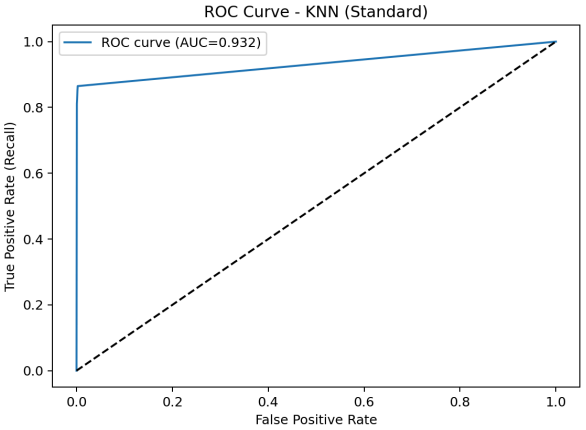


Figure K5 – Courbe ROC : KNN (Standard)

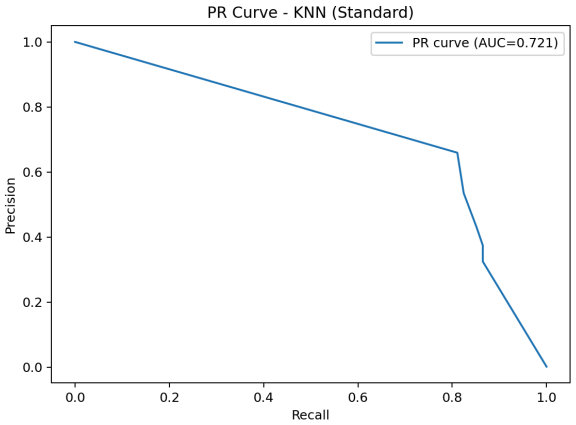


Figure K6 – Courbe PR : KNN (Standard)

MinMax

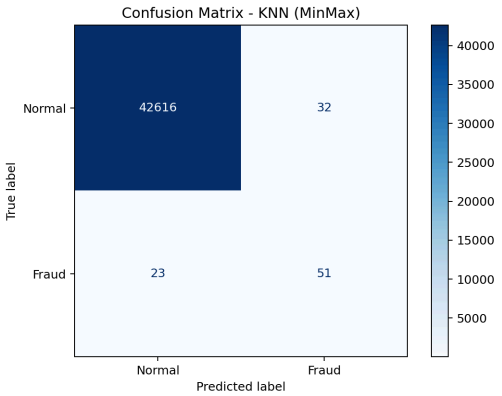


Figure K7 – Matrice de confusion : KNN (MinMax)

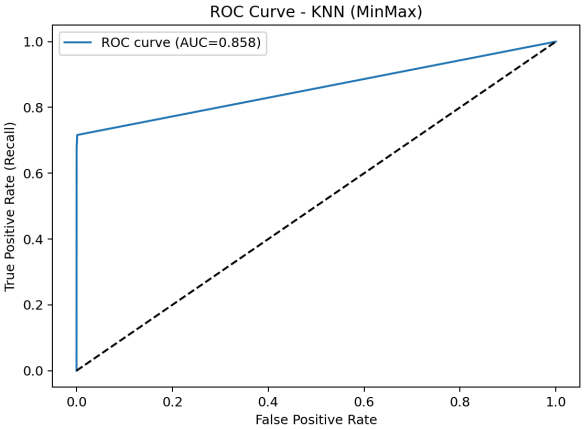


Figure K8 – Courbe ROC : KNN (MinMax)

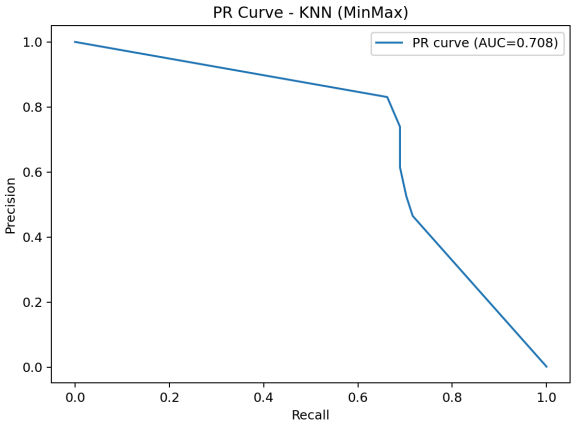


Figure K9 – Courbe PR : KNN (MinMax)

PCA_10

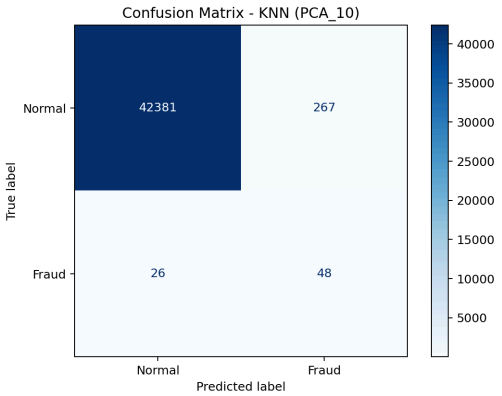


Figure K10 – Matrice de confusion : KNN (PCA_10)

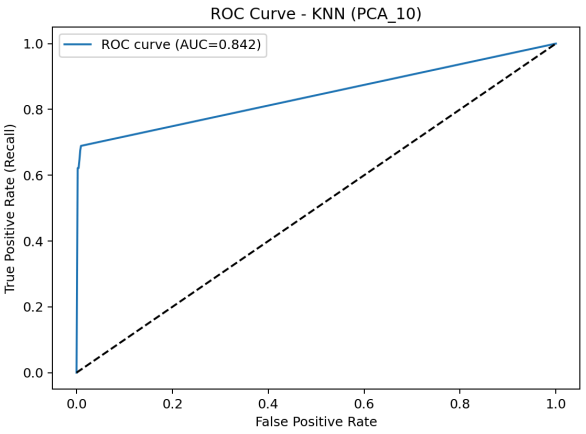


Figure K11 – Courbe ROC : KNN (PCA_10)

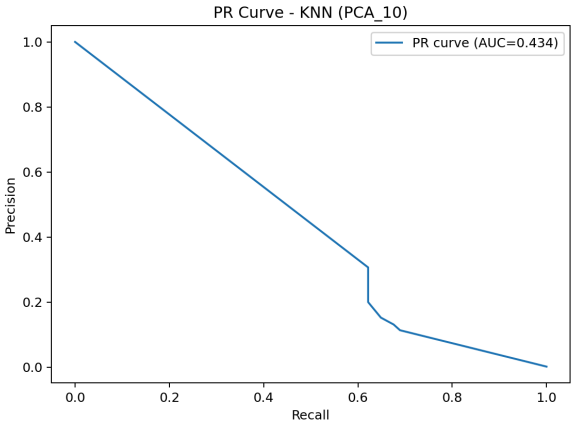


Figure K12 – Courbe PR : KNN (PCA_10)

Random Forest

Original

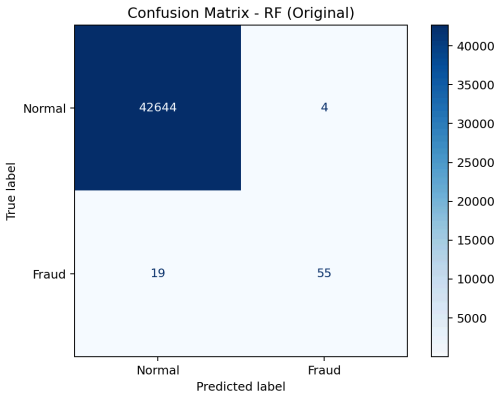


Figure R1 – Matrice de confusion : Random Forest (Original)

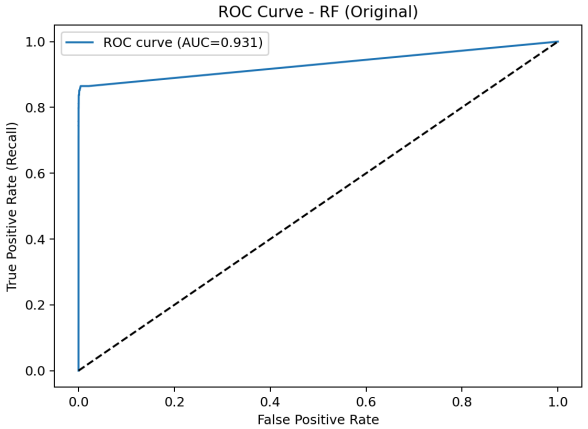


Figure R2 – Courbe ROC : Random Forest (Original)

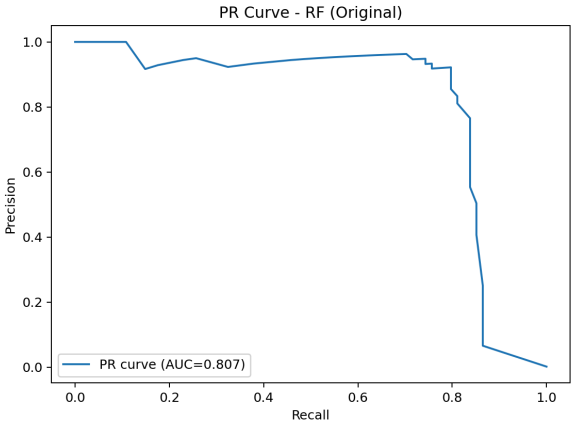


Figure R3 – Courbe PR : Random Forest (Original)

Standard

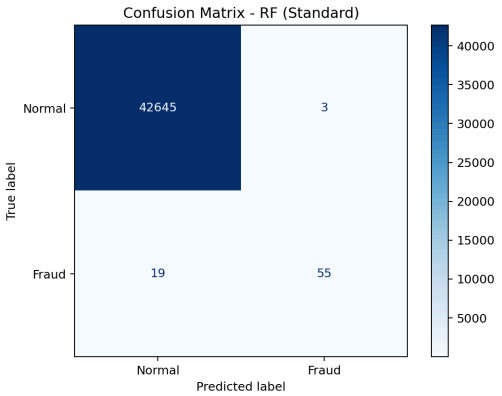


Figure R4 – Matrice de confusion : Random Forest (Standard)

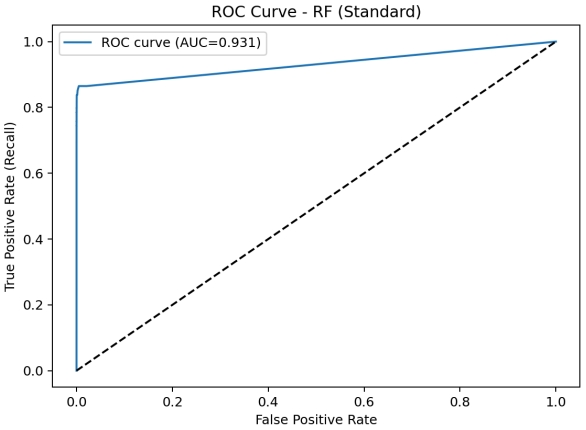


Figure R5 – Courbe ROC : Random Forest (Standard)

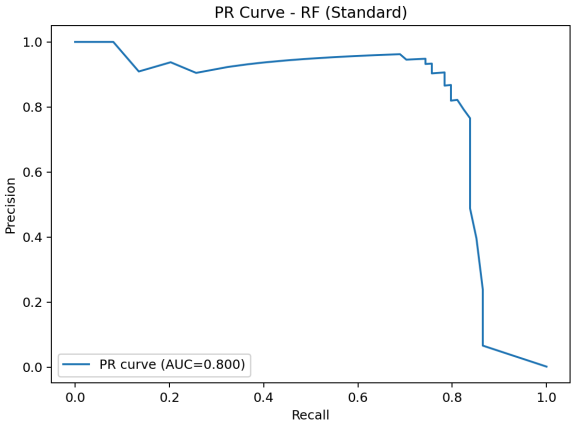


Figure R6 – Courbe PR : Random Forest (Standard)

MinMax

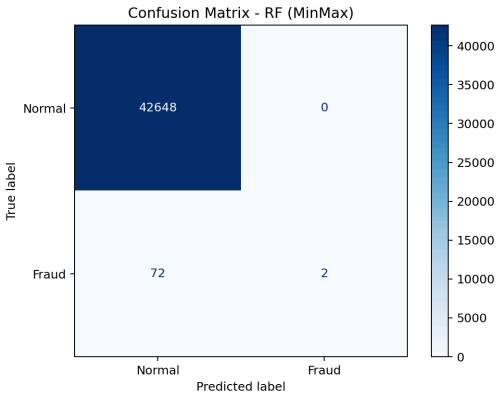


Figure R7 – Matrice de confusion : Random Forest (MinMax)

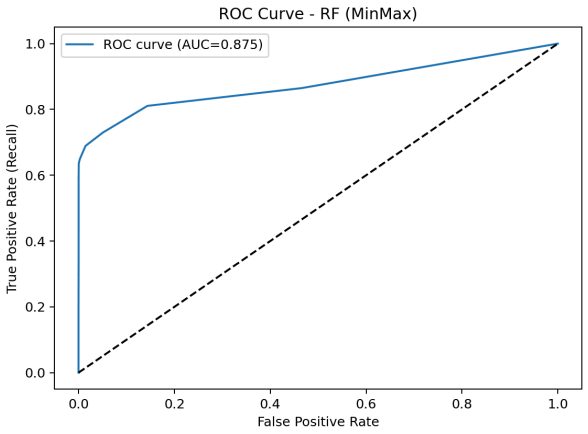


Figure R8 – Courbe ROC : Random Forest (MinMax)

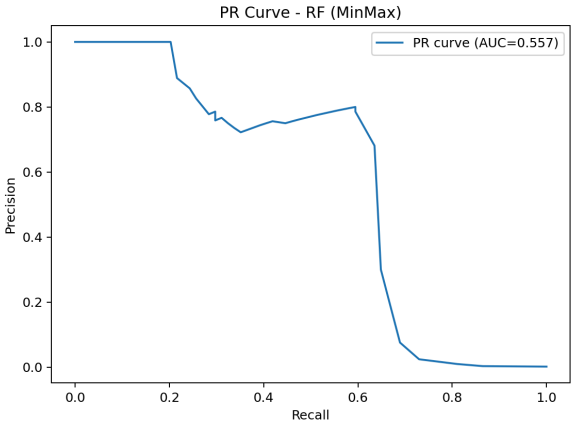


Figure R9 – Courbe PR : Random Forest (MinMax)

PCA_10

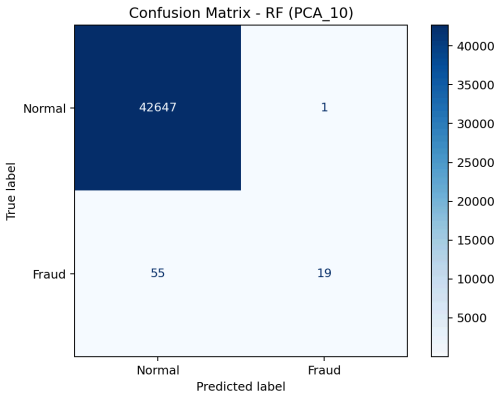


Figure R10 – Matrice de confusion : Random Forest (PCA_10)

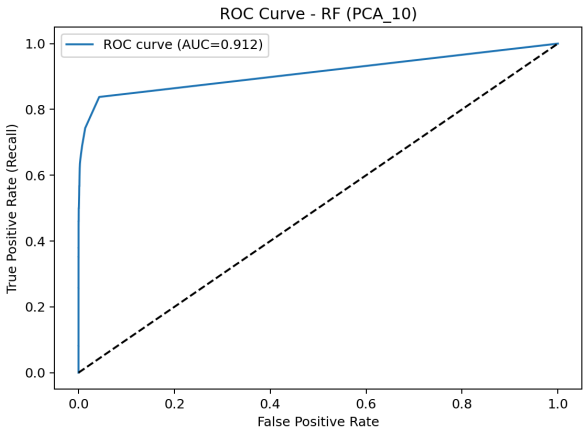


Figure R11 – Courbe ROC : Random Forest (PCA_10)

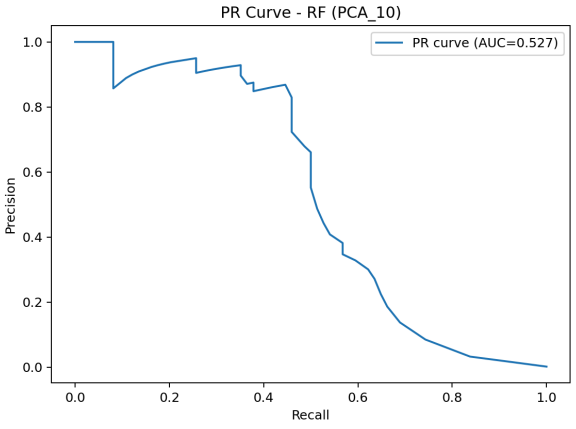


Figure R12 – Courbe PR : Random Forest (PCA_10)

CNN Model

Original

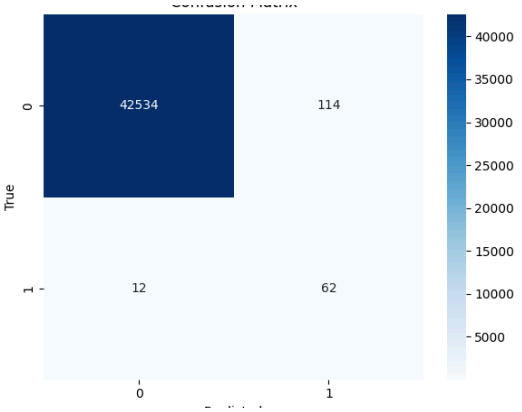


Figure N1 – Matrice de confusion : CNN

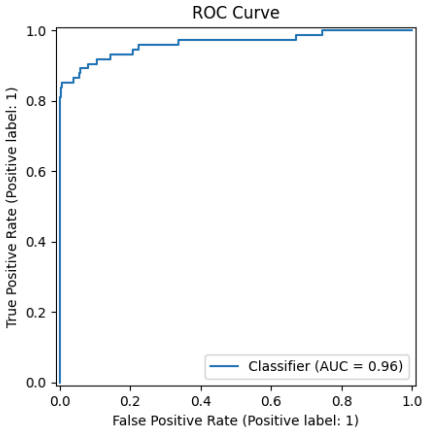


Figure N2 – Courbe ROC : CNN

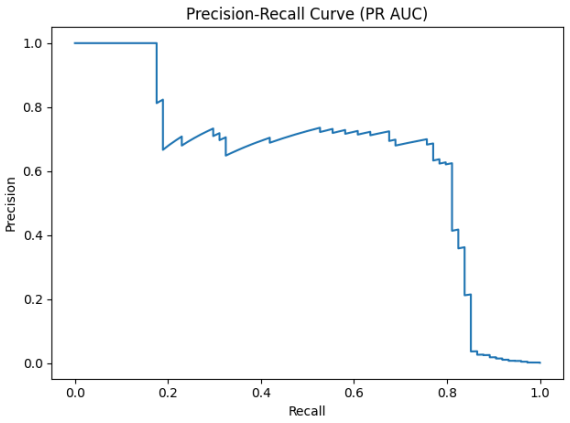


Figure N3 – Courbe PR : CNN

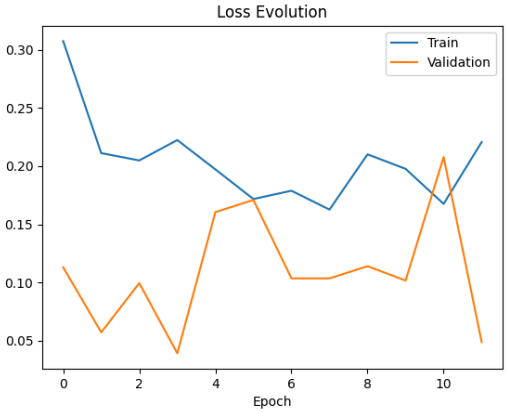


Figure N4 – Courbe de perte (Loss Curve)

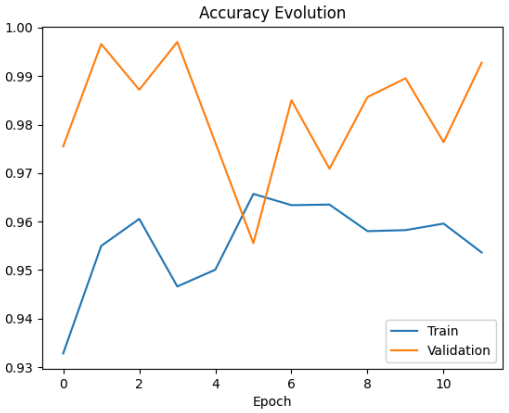


Figure N5 – Accuracy Curve