Université Paris 8
UFR Sciences et Technologies du Numérique (STN)

Projet d'Outils de Programmation Avancée pour l'IA

Détection de Fraude sur les Transactions par Cartes Bancaires

Réalisé par :

Ines FENDES

Enseignant :

M. Mehdi Ammi

Année Universitaire : 2024 - 2025

Introduction

La détection des fraudes par carte de crédit est essentielle pour bien protéger les transactions financières et maintenir la confiance des clients dans le secteur financier. Ce défi implique un impact financier important, avec des pertes mondiales prévues atteignant jusqu'à 43 milliards de dollars d'ici 2025 [3, 7]. La nécessité de s'adapter à des tactiques de fraude en constante évolution et à divers types de données de transaction ajoute une complexité supplémentaire au développement de systèmes efficaces de détection des fraudes. Les principaux défis comprennent la nature très déséquilibrée des données de transaction, les exigences de traitement en temps réel et la complexité de l'intégration et du traitement de diverses sources de données. De plus, la conformité réglementaire stricte et les coûts élevés associés aux erreurs de détection (faux positifs et faux négatifs) rendent impératif de trouver un équilibre prudent dans les stratégies de détection des fraudes [8]. Pour résoudre ces problèmes, des technologies de big data sophistiquées et des approches d'apprentissage automatique sont utilisées pour améliorer les capacités de détection et l'efficacité opérationnelle des systèmes de détection des fraudes [5]. Ces technologies sont essentielles pour développer des modèles capables d'identifier avec précision les activités frauduleuses et de s'adapter aux nouveaux modèles de fraude [6]. L'objectif de ce projet est d'utiliser des techniques avancées d'apprentissage automatique pour identifier et prévenir efficacement les transactions frauduleuses par carte de crédit. Cela implique de classer les transactions en catégories frauduleuses ou légitimes.

1 Présentation du Dataset

Les données utilisées ont été récupérés de Data.world "Credit Data Card Fraud Detection"qui contient des transactions enregistrées en septembre 2013. L'ensemble de données englobe un total de **284 807** transactions, avec **492** échantillons signalés comme frauduleux. Ce déséquilibre de classe où les transactions frauduleuses ne représentent que 0,172% du total, souligne la nécessité d'une gestion prudente de l'équilibrage des classes avant de construire des modèles. Ces transactions ont eu lieu sur une période de **deux jours**. Pour des raisons de confidentialité, la plupart des variables sont anonymisées et les caractéristiques ont été transformées à l'aide de l'analyse en composantes principales dans l'ensemble de données donné. De plus, il convient de noter que cet ensemble de données est l'un des rares disponibles à des fins de détection de fraude. De plus, les informations sur l'identifiant du titulaire de la carte ne sont pas fournies.

- Il n'y a que des caractéristiques d'entrée numériques résultant des transformations PCA (caractéristiques V1 à V28). Seules les caractéristiques « Time » et « Montant » n'ont pas été transformées avec PCA.

- La caractéristiques «Time» indique les secondes écoulées entre chaque transaction et la première dans l'ensemble de données, tandis que «Montant» représente le montant de la transaction.

- La caractéristique «Classe» sert de variable de réponse, avec une valeur de 1 indiquant une fraude et 0 dans le cas contraire. L'ensemble de données a été fourni sous un format de fichier CSV.

2 Prétraitement

C'est une étape crucial dans le développement des modèles ML et spécialement pour ce cas car les données sont fortement déséquilibrées. Après l'importation des données. On a procédé avec les étapes suivantes:

1. Vérification des valeurs manquantes

2. Vérification du déséquilibre entre les deux classes : dans cette étape il a été confirmé que les transactions frauduleuses ne représentait que 0.17% des données d'où la nécessité d'appliquer des techniques destinées à régler ce déséquilibre entre les deux classes.

3. Visualisation des données : on s'est servit des diagrammes en barre afin de bien montrer la répartition des données entre les deux classes.

4. Traitement des Outliers: puisque l'ensemble de données est entièrement transformé avec PCA, le problème des outliers est déjà été traitées.

5. Division des données en ensembles d'entraînement et de test : nous avons utilisé la fonction `train_test_split()` de scikit-learn pour diviser le jeu de données X et ses étiquettes correspondantes Y en ensembles d'entraînement et de test. La division est effectuée avec un ratio de 80/20 pour l'ensemble de test, et le paramètre `random_state` est fixé à 42 pour garantir que les données soit mélangées d'une manière aléatoire. L'ensemble d'entraînement est utilisé pour ajuster le modèle, tandis que l'ensemble de test est utilisé pour évaluer les performances du modèle sur des données nouvelles et inconnues.

6. Mise à l'échelle des caractéristiques : nous avons choisi d'utiliser le RobustScaler de scikit-learn, car il est particulièrement robuste face aux outliers, qui peuvent fortement biaiser la moyenne et l'écart-type utilisés par d'autres techniques de mise à l'échelle comme StandardScaler et MinMaxScaler. Étant donné que l'analyse en composantes principales (PCA) est déjà appliquée aux caractéristiques du jeu de données de V1 à V28, nous ne mettons à l'échelle que le champ Amount.

7. Vérification et traitement de l'asymétrie : au départ, des histogrammes ont été tracés pour toutes les caractéristiques afin d'identifier visuellement une éventuelle asymétrie. Les graphiques ont révélé que de nombreuses caractéristiques étaient effectivement fortement asymétriques.

   Ensuite, l'asymétrie de chaque caractéristique a été quantifiée en utilisant la méthode skew() de Pandas. Compte tenu de la présence d'asymétries, le PowerTransformer à été utilisé (de la suite de prétraitement de scikit-learn). Cet outil applique la transformation de Yeo-Johnson, qui est efficace pour normaliser les données asymétriques et accepte à la fois les valeurs positives et négatives, rendant ainsi les distributions plus proches d'une distribution gaussienne.

   Pour garantir une application correcte, le PowerTransformer a d'abord été ajusté sur les données d'entraînement. Ensuite, il a été utilisé pour transformer les ensembles de test.

8. Pour gérer l'important déséquilibre entre les transactions frauduleuses et légitimes, nous avons utilisé une des techniques clés de prétraitement qui est le Random Oversampling (suréchantillonage aléatoire. cette méthode augmente le nombre d'instances dans la classe minoritaire (transactions frauduleuses) en les répliquant jusqu'à ce que les deux classes soient approximativement de taille égale. Son objectif est d'améliorer l'apprentissage du modèle à partir de la classe minoritaire [4]

   Voici les étapes détaillées et les paramètres impliqués dans cette méthode [1, 2] : les principales étapes impliquées dans la technique de Suréchantillonnage aléatoire sont :

   (a) Identifier la classe minoritaire : déterminer quelle classe a moins d'instances dans l'ensemble de données. C'est la classe qui doit être suréchantillonnée.

   (b) Calculer le nombre d'instances requises : trouver la différence entre le nombre d'instances dans la classe majoritaire et la classe minoritaire.

C'est le nombre d'instances supplémentaires nécessaires dans la classe minoritaire pour atteindre une distribution égale des classes.

(c) Dupliquer les instances de la classe minoritaire : sélectionner aléatoirement des instances de la classe minoritaire, avec remplacement, et les dupliquer jusqu'à ce que le nombre souhaité d'instances supplémentaires soit atteint.

(d) Combiner la classe minoritaire suréchantillonnée avec la classe majoritaire originale : fusionner les instances de la classe minoritaire suréchantillonnée avec les instances originales de la classe majoritaire pour créer un ensemble de données équilibré.

Paramètres : les principaux paramètres impliqués dans la technique de Suréchantillonnage aléatoire pour gérer les ensembles de données déséquilibrés sont :

- Stratégie d'échantillonnage : ce paramètre détermine le rapport souhaité du nombre d'échantillons dans la classe minoritaire par rapport à la classe majoritaire après le rééchantillonnage. Les options courantes incluent 'auto' (déterminer automatiquement le rapport), 'minority' (faire correspondre le nombre d'échantillons dans la classe majoritaire) ou une valeur flottante spécifique représentant le rapport souhaité.
- État aléatoire (Random State) : ce paramètre contrôle le caractère aléatoire du processus de sélection des instances. Définir une valeur entière spécifique garantit la reproductibilité à travers plusieurs exécutions.
- Nombre de voisins (Number of Neighbors) : détermine le nombre de voisins les plus proches à utiliser lors de la génération de nouveaux échantillons synthétiques. Des valeurs plus élevées peuvent conduire à des échantillons synthétiques plus diversifiés, mais peuvent également introduire du bruit.
- Type de voisins les plus proches (Kind of Nearest Neighbors) : spécifie l'algorithme utilisé pour trouver les voisins les plus proches, tel que 'uniform' (poids égal pour tous les voisins) ou 'distance' (poids inversement proportionnel à la distance).

L'image ci-dessous montre un extrait de code de la technique de Suréchantillonnage aléatoire sur les données de notre problème.

```python
from imblearn.over_sampling import RandomOverSampler

# Define the RandomOverSampler
ros = RandomOverSampler(sampling_strategy='minority', random_state=42)

# Resample the training data using RandomOverSampler
X_ros_train_pt, y_ros_train_pt = ros.fit_resample(X_train_pt, y_train_pt)
```

Figure 1: Random oversampling

3 Modèles de classification utilisés

Afin de répondre à notre problématique, nous avons utilisé deux méthodes de classification:

- La régression logistique qui est très utilisé dans la classification binaire comme dans notre cas
- XGboost (Extreme Gradient Boosting) qui consiste à combiner plusieurs modèles d'arbres de décision faibles pour créer un modèle prédictif fort en ajoutant itérativement de nouveaux arbres à l'ensemble afin d'améliorer la précision globale des prédictions.

4 Métriques d'évaluation des modèles

Pour l'évaluation des modéles nouas avon utilisés les métriques suivantes:

- **Précision :** c'est une métrique de performance utilisée pour évaluer l'exactitude des modèles de classification ou de prédiction. Elle représente le rapport entre le nombre de cas positifs correctement prédits (vrais positifs) et le nombre total de cas positifs prédits, qui inclut à la fois les vrais positifs et les faux positifs (cas positifs prédits incorrectement). En essence, la précision mesure la proportion des cas positifs prédits qui sont en réalité des vrais positifs. Sa formule est comme suit:

$$\text{Précision} = \frac{\text{Vrai positifs}}{\text{Vrai positifs} + \text{Faux positifs}} \tag{1}$$

- **Rappel:**Un indicateur clé dans les modèles de classification ou de prédiction, le rappel mesure à quel point le modèle identifie de manière exhaustive les cas positifs. Il est calculé en divisant le nombre de vrais positifs (cas positifs

correctement prédits) par la somme des vrais positifs et des faux négatifs (cas positifs manqués).

$$\text{Rappel} = \frac{\text{Vrai positifs}}{\text{Vrai positifs} + \text{Faux négatifs}} \tag{2}$$

- **Score AUC :** Elle sert de métrique fondamentale pour évaluer l'efficacité des classificateurs binaires. Elle quantifie la capacité d'un modèle à différencier entre les classes positives et négatives en calculant l'aire sous la courbe de caractéristique de fonctionnement du récepteur (ROC).

- **Score F1 :**C'est une métrique couramment utilisée pour évaluer la performance des classificateurs binaires. Elle combine la précision et le rappel en un seul score et fournit une mesure de l'exactitude globale du modèle. Elle est calculée par la formule suivante :

$$\text{F1-Score} = 2 \times \frac{\text{Précision} \times \text{Rappel}}{\text{Précision} + \text{Rappel}} \tag{3}$$

5 Optimisation et choix des hyper-paramètres

Afin d'évaluer rigoureusement l'efficacité de chaque méthode de classification, nous avons utilisé une approche en deux étapes pour le réglage et la validation du modèle.

Dans un premier temps, une validation croisée stratifiée (Stratified Cross Validation) a été mise en œuvre pour garantir que chaque pli des ensembles d'entraînement et de validation représente une proportion similaire de distributions de classes. Après la validation croisée stratifiée, nous avons utilisé GridSearchCV pour ajuster les hyperparamètres de chaque modèle. Cette approche systématique d'optimisation des hyperparamètres a été réalisée après avoir réduit la plage des paramètres en fonction des premiers résultats de performance du modèle. Cette méthode garantit non seulement une configuration optimale du modèle, mais améliore également la capacité de généralisation des modèles pour détecter les transactions frauduleuses.

6 Résultats :

Modéles de classification	ROC-AUC Score	F1-Score	Precision	Recall
Régression logistique	0.971	0.932	0.925	0.938
XGboost	0.978	0.938	0.991	0.890

La figure ci dessous 2 résume les résultas obtenus en termes de métriques d'évaluation pour les deux modèles :

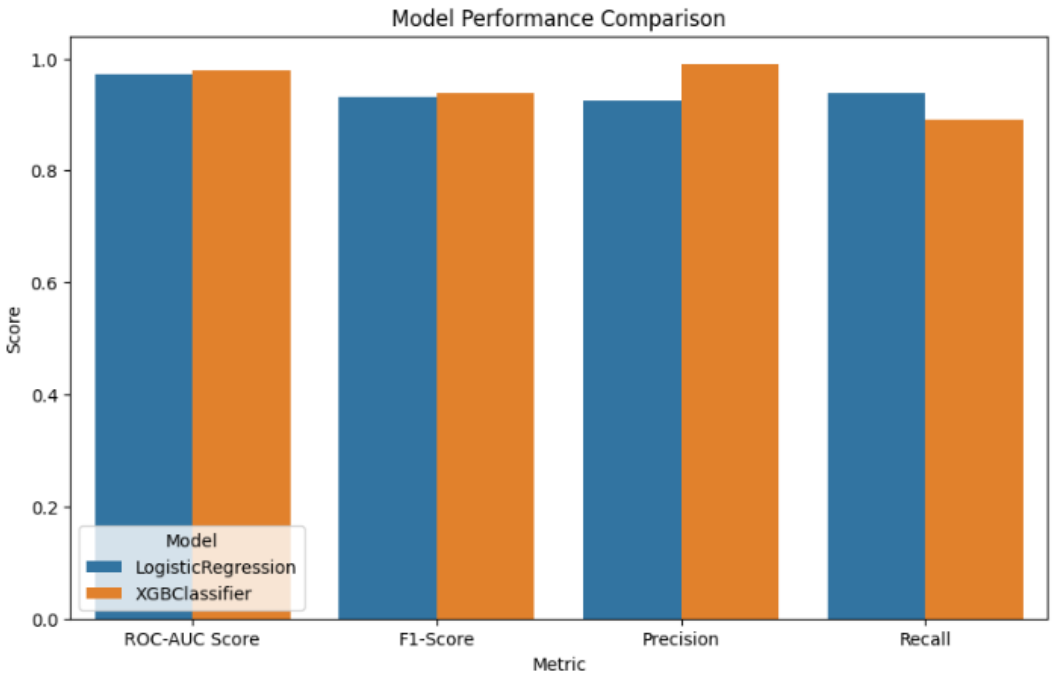

Figure 2: Comparaison des performances des modèles

Nous remarquons que les deux modèles on donné des performances similaires. En termes de temps de prédiction, c'est la régression logistique qui avait un temps de prédiction rapide (2.13 s) par rapport à celui du XGBOOST(8.74s).

7 Conclusion

Dans ce travail, on s'est concentré sur la mise en œuvre de deux modèles d'apprentissage automatique pour former deux classifieurs différents : la régression Logistique et XGBoost. Pour optimiser les performances du modèle, nous avons appliqué une méthode de prétraitement pour gérer les déséquilibres entre nos deux classe . De plus, nous avons utilisé une optimisation des hyperparamètres pour chaque méthode de classification. Une validation croisée a également été utilisé pour produire des modèles plus préciss et plus fiables. Pour faciliter la comparaison entre les différentes approches, nous avons utilisé différentes mesures d'évaluation, avec une concentration principale sur le score ROC AUC en raison de son caractère complet qui mesure la performance du modèle sur différents seuils. Enfin , nos expériences ont révélé que le XGBoost modèle a atteint les performances les plus élevées avec un score de précision de 0,992. Une perspective intéressante serait d'explorer des méthodes hybrides qui combinent l'apprentissage supervisé et non supervisé. Ces méthodes pourraient exploiter les atouts des deux approches pour détecter différents types de fraude avec plus de précision et de robustesse.

References

[1] Gustavo EAPA Batista, Ronaldo C Prati, and Maria Carolina Monard. A study of the behavior of several methods for balancing machine learning training data. *ACM SIGKDD explorations newsletter*, 6(1):20–29, 2004.

[2] Haibo He, Yang Bai, EA Garcia, and Shutao Li. Adasyn: Adaptive synthetic sampling approach for imbalanced learning. *IEEE International Joint Conference on Neural Networks (IEEE World Congress on Computational Intelligence)*, pages 1322–1328, 2008.

[3] Syed Kamaruddin, Md Kamal Hasan, and Md Mustafizur Rahman. Credit card fraud detection using big data analytics. In *Proceedings of the International Conference on Computing Advancements*, pages 1–6, 2018.

[4] Saeed Mirshekari. Dealing with imbalanced data: Oversampling techniques, 2023.

[5] Namrata Pandey, S Rajeshwari, Shobha BN Rani, and B Mounica. Credit card fraud detection using big data framework. *International Journal of Creative Research Thoughts (IJCRT)*, 6(2):523–526, 2018.

[6] Ravi Patidar and Lokesh Sharma. Challenges and complexities in machine learning based credit card fraud detection. *arXiv preprint arXiv:2208.10943*, 2022.

[7] M Sathyapriya and V Thiagarasu. Big data analytics techniques for credit card fraud detection: A review. *International Journal of Science and Research (IJSR)*, 6(5):1–5, 2017.

[8] SEON. Credit card fraud detection: Best methods in 2023. https://seon.io/resources/credit-card-fraud-detection/, 2023.