

Rapport de Projet : Application d'Apprentissage Automatique avec Python

Bouzidi Mariam, Lajmi Sahar, Eya Bouhouch

1 Introduction

1.1 Présentation du Projet

Le projet vise à développer une application Python avec Streamlit qui permet de comparer différents modèles de classification pour déterminer le niveau de risque associé à chaque transaction blockchain dans le Metaverse ouvert. Les transactions seront classées en trois catégories : risque faible, risque modéré et risque élevé. La comparaison des modèles se basera sur diverses métriques telles que la précision, le rappel, la courbe ROC, et d'autres indicateurs pertinents.

1.2 Résumé des Fonctionnalités de l'Application Python avec Streamlit

L'application permettra aux utilisateurs de sélectionner différents modèles de classification, de visualiser les performances de ces modèles via des graphiques interactifs et de comprendre les caractéristiques clés qui influencent la classification des risques. Elle offrira également la possibilité d'ajuster les paramètres des modèles pour optimiser leur performance.

1.3 Brève Explication sur l'Ensemble de Données Utilisé

L'ensemble de données comprend 78,600 transactions financières au sein du Metaverse ouvert, enregistrant divers types de transactions et profils de risque. Ce dataset est une ressource essentielle pour tester et développer des modèles de détection d'anomalies et d'analyse de fraude.

2 Prétraitement des Données

2.1 Description de l'Ensemble de Données

Chaque transaction dans l'ensemble de données est décrite par les attributs suivants :

- **timestamp** : Date et heure de la transaction, utilisées pour analyser les tendances au fil du temps.
- **hourofday** : Heure de la transaction, qui peut être un indicateur des comportements d'achat.
- **amount** : Montant de la transaction, un facteur clé dans l'analyse de risque.
- **transactiontype** : Type de la transaction (par exemple, achat, vente), indiquant la nature de l'activité.
- **locationregion** : Région géographique où la transaction a été effectuée, potentiellement utile pour identifier des schémas spécifiques à certaines zones.
- **loginfrequency** : Fréquence de connexion de l'utilisateur, qui peut signaler des comportements d'utilisateur normaux ou suspects.
- **sessionduration** : Durée de la session utilisateur lors de la transaction, un possible indicateur de risque.
- **purchasepattern** : Modèle d'achat de l'utilisateur, dérivé des comportements d'achat précédents.
- **agegroup** : Groupe d'âge de l'utilisateur, pouvant affecter le type et la fréquence des transactions.
- **riskscore** : Score de risque attribué à la transaction, basé sur divers facteurs.

2.2 Techniques de Prétraitement Appliquées

Le prétraitement des données comprend les étapes suivantes :

- **Suppression de Colonnes** : Élimination des colonnes telles que 'sendingaddress', 'receivingaddress', et 'ipprefix' pour se concentrer sur les attributs les plus pertinents et réduire les dimensions.
- **Transformation des Dates** : Conversion de la colonne 'timestamp' en format datetime pour faciliter les analyses temporelles.
- **Tri et Réinitialisation de l'Index** : Les données ont été triées par timestamp et l'index a été réinitialisé pour garantir la séquentialité.

- **Encodage des Variables Catégorielles :** Les variables 'purchasepattern' et 'agegroup' ont été transformées en nombres entiers pour faciliter leur traitement par les algorithmes de classification.
- **Création de Variables Indicateurs :** Conversion des variables catégorielles 'transactiontype' et 'locationregion' en variables indicatrices.
- **Normalisation des Données :** Application du MinMaxScaler pour normaliser les colonnes numériques et les amener à une échelle commune.

2.3 Justification des Choix de Prétraitement

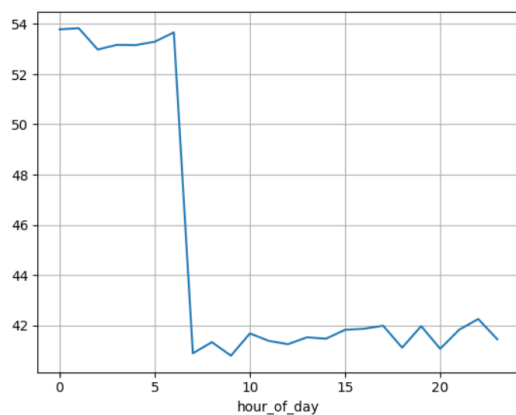
Ces choix de prétraitement visent à simplifier l'ensemble de données en éliminant les informations redondantes ou peu utiles, à garantir l'intégrité des types de données pour les analyses ultérieures, et à préparer les données pour une utilisation efficace dans des modèles de machine learning.

3 Exploration des Données

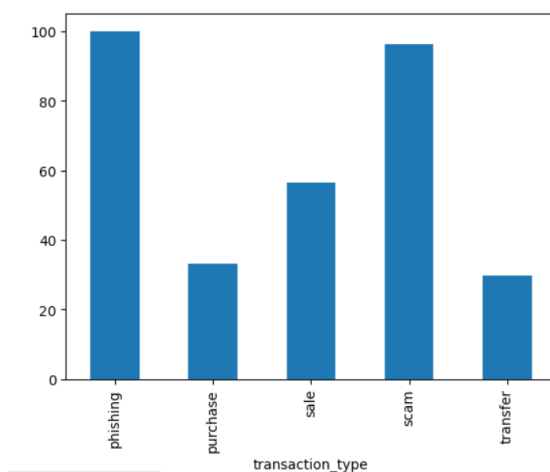
3.1 Visualisations des Données Exploratoires

Pour mieux comprendre la distribution des données et leurs relations, des visualisations exploratoires ont été réalisées. Voici quelques-unes des visualisations effectuées :

- Graphique linéaire de la Moyenne du Score de Risque par Heure du Jour



- Diagramme à Barres du moyenne Score de Risque par Type de Transaction



3.2 Analyses Statistiques des Caractéristiques Importantes

Des analyses statistiques ont été effectuées pour examiner les caractéristiques importantes des données. Cela inclut des calculs de moyennes, de médianes, d'écart-types, etc., pour différentes variables afin de mieux comprendre leur distribution et leur variation.

3.3 Insights Tirés de l'Exploration des Données

L'exploration des données a permis de tirer plusieurs insights intéressants :

- Les transactions effectuées aux heures de faible luminosité présentent généralement des risques plus élevés, ce qui peut indiquer une activité frauduleuse accrue pendant ces périodes.
- Certains types de transactions, tels que les escroqueries et les phishing, ont tendance à avoir des scores de risque plus élevés, soulignant l'importance de surveiller de près ces types d'activités.
- Les transactions effectuées dans certaines régions géographiques sont associées à des niveaux de risque différents, ce qui peut être attribuable à des facteurs socio-économiques ou à des politiques régionales spécifiques.
- Les utilisateurs appartenant à certains groupes d'âge présentent des schémas d'achat et des comportements de risque distincts, ce qui souligne l'importance de prendre en compte les caractéristiques démographiques dans l'évaluation du risque de transaction.

4 Interface Graphique

4.1 Description de l'Interface Graphique

L'interface graphique de notre application, propose une expérience conviviale à l'utilisateur, lui permettant d'interagir aisément avec divers modèles de Machine Learning et d'explorer les résultats.

Cette interface intuitive intègre des composants interactifs tels que des curseurs, des menus déroulants et des boutons, offrant ainsi la possibilité de sélectionner et de personnaliser les modèles de Machine Learning et les paramètres d'entraînement.

En outre, elle propose des fonctionnalités de visualisation de données, permettant l'exploitation des caractéristiques de l'ensemble de données et les résultats des modèles via des graphiques et des diagrammes, offrant à l'utilisateur la possibilité de visualiser les distributions, les corrélations et d'autres aspects des données.

4.2 Les Bibliothèques utilisées

Pour développer notre interface graphique, nous avons exploité plusieurs bibliothèques Python. Voici un aperçu des principales bibliothèques que nous avons intégrées :

- **Streamlit**: l'élément central de notre développement. Elle nous a permis de créer une interface utilisateur en Python de manière intuitive et efficace. Avec Streamlit, nous avons pu intégrer des composants tels que des boutons, des curseurs et des menus déroulants, offrant une expérience fluide et agréable.
- **Pandas**: Pour la manipulation et l'analyse des données, nous avons utilisé Pandas. Cette bibliothèque nous a permis de charger et de prétraiter les données provenant de notre ensemble de données.
- **Matplotlib et Seaborn**: Pour la visualisation des données, nous avons fait appel à Matplotlib et Seaborn. Ces bibliothèques nous ont offert un large éventail d'options de visualisation: Des histogrammes, des nuages de points et des matrices de corrélation font partie des graphiques que nous avons inclus pour aider l'utilisateur à mieux comprendre les données et les résultats des modèles.
- **Scikit-learn**: Scikit-learn a été notre choix principal pour l'entraînement et l'évaluation des modèles de Machine Learning. Cette bibliothèque offre une large gamme d'algorithmes de ML et des outils puissants pour évaluer la performance des modèles.

En combinant ces bibliothèques avec Streamlit, nous avons pu créer une interface graphique interactive et informatique pour notre application d'apprentissage automatique.

5 Modèles de Machine Learning

Dans cette section, nous décrivons les différents algorithmes de Machine Learning que nous avons utilisés dans notre application. Chaque algorithme est présenté avec une brève description de son fonctionnement et de ses caractéristiques.

5.1 Support Vector Machine (SVM)

5.1.1 Description du Principe

Le Support Vector Machine (SVM) est un algorithme d'apprentissage supervisé utilisé pour la classification et la régression. Son principe est de trouver un hyperplan dans un espace de données à N dimensions (où N est le nombre de caractéristiques) qui sépare les classes de manière optimale.

5.1.2 Origine Mathématique

L'origine mathématique de SVM repose sur la notion de marge maximale. L'objectif est de trouver l'hyperplan qui maximise la distance entre les points les plus proches de chaque classe, appelés vecteurs de support. Cela revient à résoudre un problème d'optimisation quadratique.

5.1.3 Paramètres Ajustables Utilisés

Les paramètres ajustables utilisés dans le code pour le SVM sont :

- **C (Paramètre de régularisation) :** Contrôle le compromis entre maximiser la marge et minimiser l'erreur de classification. Une valeur plus élevée de C conduit à une marge plus étroite mais une meilleure classification des données d'entraînement.
- **Kernel (Noyau) :** Spécifie le type de fonction noyau à utiliser pour transformer l'espace de données. Les noyaux couramment utilisés incluent linéaire, polynomial et gaussien (rbf).
- **Gamma (Coefficient du noyau) :** Contrôle la souplesse de la frontière de décision. Une valeur plus élevée de γ signifie que le modèle sera plus sensible aux points de données individuels.

The image shows a web interface titled "Choose Classifier". It features a dropdown menu for "Classifier" with "Support Vector Machine (SVM)" selected. Below this is a checked checkbox for "Best Parameters". The "Model Hyperparameters" section includes a "C (Regularization Parameter)" input field set to "0,01" with minus and plus buttons. The "Kernel" section has two radio buttons: "rbf" (selected) and "linear". The "Gamma(Kernel coefficient)" section has two radio buttons: "scale" (selected) and "auto". A "What metrics to plot?" dropdown menu shows "Choose an option". At the bottom is a red "Classify" button.

Figure 1: SVM

5.2 Régression Logistique

5.2.1 Description du Principe

La Régression Logistique est un modèle statistique utilisé pour la classification binaire. Contrairement à son nom, elle est utilisée pour la classification plutôt que pour la régression. Le modèle estime la probabilité qu'une observation appartienne à une classe particulière en utilisant une fonction logistique.

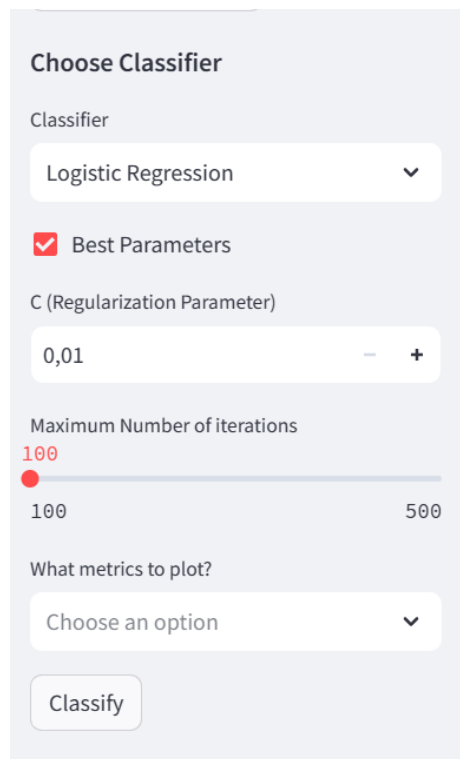
5.2.2 Origine Mathématique

L'origine mathématique de la régression logistique réside dans la fonction logistique, qui est une fonction sigmoïde qui transforme une valeur continue en une probabilité comprise entre 0 et 1. Le modèle cherche à ajuster les paramètres pour maximiser la vraisemblance des observations dans l'échantillon.

5.2.3 Paramètres Ajustables Utilisés

Les paramètres ajustables utilisés dans le code pour la régression logistique sont :

- **C (Paramètre de régularisation)** : Contrôle la force de la régularisation. Une valeur plus élevée de C réduit la régularisation, ce qui peut conduire à un surapprentissage.
- **Max_iter (Nombre maximal d'itérations)** : Spécifie le nombre maximal d'itérations pour la convergence de l'algorithme de descente de gradient lors de l'ajustement des paramètres.



The image shows a web-based interface for configuring a Logistic Regression classifier. The interface is titled "Choose Classifier" and includes several adjustable parameters:

- Classifier**: A dropdown menu currently set to "Logistic Regression".
- Best Parameters**: A checkbox that is checked, indicating that the system should find the best parameters.
- C (Regularization Parameter)**: A numeric input field set to "0,01", with minus and plus buttons for adjustment.
- Maximum Number of iterations**: A slider control ranging from 100 to 500, with a red dot indicating the current value is 100.
- What metrics to plot?**: A dropdown menu currently set to "Choose an option".
- Classify**: A button at the bottom to execute the classification process.

Figure 2: Regression logistique

5.3 Arbre de Décision

5.3.1 Description du Principe

Un arbre de décision est un modèle d'apprentissage supervisé utilisé pour la classification et la régression. Il divise l'espace de données en partitions récursives basées sur des règles simples dérivées des caractéristiques.

5.3.2 Origine Mathématique

L'origine mathématique des arbres de décision réside dans la théorie de l'information et de l'entropie. L'algorithme construit l'arbre en choisissant à chaque étape la caractéristique qui maximise la réduction d'entropie ou de gain d'information.

5.3.3 Paramètres Ajustables Utilisés

Les paramètres ajustables utilisés dans le code pour les arbres de décision sont :

- **Max_depth (Profondeur maximale de l'arbre)** : Limite la profondeur maximale de l'arbre de décision pour éviter le surapprentissage.
- **Min_samples_split (Nombre minimum d'échantillons requis pour diviser un nœud interne)** : Spécifie le nombre minimum d'échantillons requis pour diviser un nœud interne de l'arbre.
- **Min_samples_leaf (Nombre minimum d'échantillons requis pour être à un nœud feuille)** : Spécifie le nombre minimum d'échantillons requis pour former un nœud feuille de l'arbre.

The image shows a user interface for selecting a classifier and adjusting its parameters. The title is "Choose Classifier". Under the "Classifier" label, a dropdown menu is set to "Decision Tree". Below this, there is a checkbox labeled "Best Parameters" which is checked. Three sliders are visible, each with a red dot indicating the current value and a range of values shown below the slider. The first slider is labeled "Max Depth" and is set to 1, with a range from 1 to 20. The second slider is labeled "Min Samples Split" and is set to 2, with a range from 2 to 10. The third slider is labeled "Min Samples Leaf" and is set to 1, with a range from 1 to 10. At the bottom, there is a text input field labeled "What metrics to plot?".

Figure 3: arbre de decesion

5.4 Forêt Aléatoire

5.4.1 Description du Principe

La forêt aléatoire est une méthode d'ensemble utilisée pour la classification et la régression. Elle construit plusieurs arbres de décision lors de l'apprentissage et les combine pour obtenir une prédiction plus robuste et précise.

5.4.2 Origine Mathématique

L'origine mathématique des forêts aléatoires réside dans le concept d'agrégation des prédictions de multiples arbres de décision. Chaque arbre est construit sur un sous-ensemble aléatoire des données et des caractéristiques, puis les prédictions sont combinées par vote majoritaire (classification) ou moyenne (régression).

5.4.3 Paramètres Ajustables Utilisés

Les paramètres ajustables utilisés dans le code pour les forêts aléatoires sont :

- **Nombre d'estimateurs (N_estimators)** : Spécifie le nombre d'arbres de décision à construire dans la forêt.
- **Profondeur maximale de chaque arbre (Max_depth)** : Limite la profondeur maximale de chaque arbre de décision pour éviter le surapprentissage.
- **Échantillonnage avec remplacement (Bootstrap)** : Indique si les échantillons sont tirés avec remplacement lors de la construction des arbres.

Choose Classifier

Classifier

Random Forest

☐ Best Parameters

Number of Estimators

100

100

5000

Max Depth

100

100

500

What metrics to plot?

Choose an option

Classify

Figure 4: forets aleatoire

5.5 K-Nearest Neighbors (KNN)

5.5.1 Description du Principe

K-Nearest Neighbors est un algorithme d'apprentissage supervisé utilisé pour la classification et la régression. Il classe les nouvelles observations en fonction de la majorité des voisins les plus proches dans l'espace de caractéristiques.

5.5.2 Origine Mathématique

L'origine mathématique de KNN réside dans le calcul de la distance entre les points dans l'espace de caractéristiques. La classification d'une nouvelle observation est déterminée en identifiant les k voisins les plus proches et en attribuant la classe majoritaire parmi ces voisins.

5.5.3 Paramètres Ajustables Utilisés

Les paramètres ajustables utilisés dans le code pour KNN sont :

- **N_neighbors (Nombre de voisins)** : Spécifie le nombre de voisins à considérer lors de la classification d'une nouvelle observation.
- **Weights (Poids)** : Indique comment pondérer les voisins lors de la classification. Les options incluent *uniform*, où tous les voisins ont le même poids, et *distance*, où les poids sont inverses à la distance entre les points.

Choose Classifier

Classifier

K-Nearest Neighbors (KNN) ▼

☐ Best Parameters

Number of Neighbors

1

1 20

Weights

☒ uniform

☐ distance

What metrics to plot?

Confusion Matrix × × ▼

Classify

Figure 5: KNN

5.6 Naive Bayes

5.6.1 Description du Principe

Naive Bayes est un algorithme d'apprentissage supervisé basé sur le théorème de Bayes. Il est particulièrement efficace pour la classification de texte et de données à haute dimensionnalité. L'algorithme suppose que les caractéristiques sont indépendantes les unes des autres, d'où le terme "naïf".

5.6.2 Origine Mathématique

L'origine mathématique de Naive Bayes repose sur le théorème de Bayes, qui décrit la probabilité conditionnelle des événements. L'algorithme calcule la probabilité a posteriori de chaque classe pour une observation donnée en utilisant les probabilités a priori des classes et les distributions des caractéristiques.

5.6.3 Paramètres Ajustables Utilisés

Les paramètres ajustables utilisés dans le code pour Naive Bayes sont :

- **Alpha** : Paramètre de lissage de Laplace pour éviter les probabilités nulles lorsqu'une caractéristique n'apparaît pas dans l'ensemble d'entraînement.

- **Fit_prior** : Indique si les probabilités a priori des classes doivent être ajustées en fonction des données ou si elles doivent être considérées comme uniformes.

Choose Classifier

Classifier

Naive Bayes ▼

☐ Best Parameters

Alpha (Smoothing Parameter)

0.01

0.01 2.00

Fit Prior

☒ True

☐ False

What metrics to plot?

Choose an option ▼

Classify

Figure 6: naive bayes

5.7 Réseau de Neurones (Neural Network)

5.7.1 Description du Principe

Un réseau de neurones est un modèle d'apprentissage supervisé inspiré du fonctionnement du cerveau humain. Il est composé de couches de neurones interconnectés qui transforment les entrées en sorties à l'aide de fonctions d'activation.

5.7.2 Origine Mathématique

L'origine mathématique des réseaux de neurones réside dans la théorie des réseaux de neurones artificiels et de l'apprentissage profond. Les neurones effectuent des opérations linéaires et non linéaires sur les entrées pondérées par des poids, et l'apprentissage est effectué par la rétropropagation du gradient.

5.7.3 Paramètres Ajustables Utilisés

Les paramètres ajustables utilisés dans le code pour les réseaux de neurones sont :

- **Hidden_layer_sizes** : Nombre et taille des couches cachées du réseau.
- **Activation** : Fonction d'activation utilisée dans chaque couche.
- **Solver** : Algorithme d'optimisation utilisé pour apprendre les poids du réseau.
- **Alpha** : Paramètre de régularisation pour contrôler le surapprentissage.
- **Learning_rate** : Taux d'apprentissage utilisé par l'optimiseur lors de la mise à jour des poids du réseau.

Classifier

Neural Network ▼

☒ Best Parameters

Hidden Layer Sizes

(100,)

(100,) (100, 100, 100)

Activation Function

identity ▼

Solver

adam ▼

Alpha

0.00

0.00 1.00

Learning Rate

constant ▼

What metrics to plot?

Choose an option ▼

Figure 7: Reseau de neurones

Ces paramètres permettent de régler la complexité et la capacité de généralisation du réseau de neurones en fonction de la tâche d'apprentissage.

5.8 Option de Meilleurs Paramètres

L'option de meilleurs paramètres permet à l'utilisateur de trouver les combinaisons optimales des hyperparamètres pour chaque algorithme de Machine Learning. Plutôt que de spécifier manuellement les valeurs des hyperparamètres, cette fonctionnalité utilise une technique de recherche systématique appelée recherche de grille (*grid search*).

La recherche de grille explore un ensemble prédéfini de combinaisons d'hyperparamètres, évalue la performance du modèle pour chaque combinaison en utilisant une métrique spécifiée, puis sélectionne la combinaison qui donne les meilleurs résultats. Cela permet d'automatiser le processus de réglage des hyperparamètres et d'obtenir des modèles plus performants.

Une fois que la recherche de grille est effectuée, les meilleurs hyperparamètres trouvés sont utilisés pour entraîner le modèle, ce qui permet d'obtenir une meilleure performance par rapport à l'utilisation de valeurs arbitraires pour les hyperparamètres.

Metaverse Blockchain Transaction Risk Analysis

K-Nearest Neighbors Results

Best Parameters: n_neighbors= 1 weights= uniform

Accuracy: 1.0

Precision: 1.0

Recall: 1.0

Figure 8: meilleurs paramètres KNN

Metaverse Blockchain Transaction Risk Analysis

Logistic Regression Results

Best Parameters: C= 10 max_iter= 300

Accuracy: 0.99

Precision: 0.97

Recall: 0.98

Figure 9: meilleurs parametres regression logistiques

5.9 Méthodologie pour l'Entraînement

La méthodologie pour l'entraînement des modèles commence par la séparation des données en ensembles d'entraînement et de test. Cela garantit que le modèle est évalué sur des données qu'il n'a pas vues pendant l'entraînement, ce qui permet d'estimer sa capacité à généraliser sur de nouvelles données.

Ensuite, l'utilisateur sélectionne l'algorithme de Machine Learning à entraîner parmi une liste d'options disponibles, telles que SVM, régression logistique, arbre de décision, etc. En fonction de l'algorithme sélectionné, l'utilisateur peut choisir d'ajuster manuellement les hyperparamètres ou d'utiliser l'option de meilleurs paramètres.

Une fois les hyperparamètres définis, le modèle est entraîné sur l'ensemble d'entraînement. Pendant l'entraînement, le modèle ajuste ses paramètres pour minimiser une fonction de perte ou maximiser une fonction d'objectif, en fonction de la tâche d'apprentissage spécifique (classification, régression, etc.).

Après l'entraînement, le modèle est évalué sur l'ensemble de test pour mesurer sa performance. Différentes métriques d'évaluation, telles que l'exactitude, la précision, le rappel, sont calculées pour évaluer la qualité des prédictions du modèle. Ces métriques permettent de déterminer si le modèle est adapté à la tâche donnée et s'il peut généraliser efficacement sur de nouvelles données.

5.10 Performance des Modèles

La performance des modèles est évaluée à l'aide de diverses métriques qui fournissent des informations sur la qualité des prédictions du modèle. Ces métriques comprennent :

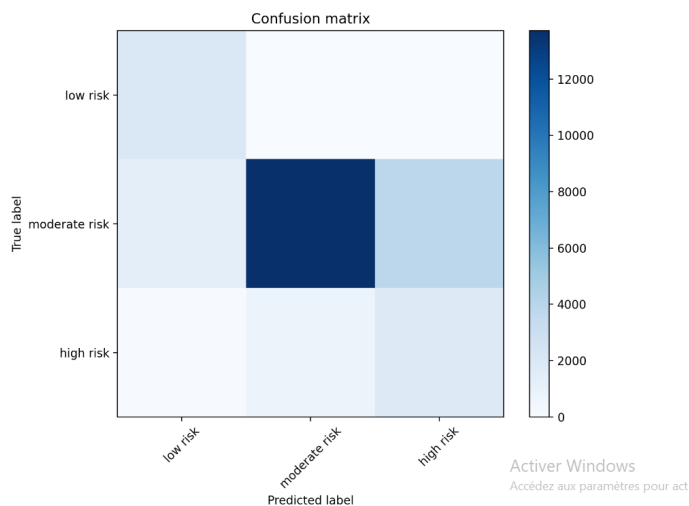


Figure 10: matrice de confusion

- **Exactitude (*Accuracy*)** : Le pourcentage de prédictions correctes parmi toutes les prédictions effectuées par le modèle. Il mesure la capacité globale du modèle à prédire correctement les étiquettes des échantillons.
- **Précision et Rappel (*Precision and Recall*)** : Ces métriques sont utilisées dans les tâches de classification pour évaluer la qualité des prédictions positives par rapport aux faux positifs et aux faux négatifs. La précision mesure le nombre de vrais positifs parmi toutes les prédictions positives, tandis que le rappel mesure le nombre de vrais positifs parmi toutes les instances réellement positives.
- **Courbes ROC et de Précision-Rappel (*ROC and Precision-Recall Curves*)** : Ces courbes sont utilisées pour visualiser la performance d'un modèle de classification à différents seuils de décision. La courbe ROC représente le taux de vrais positifs par rapport au taux de faux positifs, tandis que la courbe de précision-rappel représente la précision par rapport au rappel à différents seuils de décision.

En évaluant les modèles à l'aide de ces métriques, il est possible de comparer leur performance et de sélectionner le meilleur modèle pour la tâche donnée.

6 Sélection du Meilleur Modèle

6.1 Méthodologie pour Comparer les Différents Modèles

Dans cette partie, nous avons comparé plusieurs modèles de Machine Learning afin de déterminer celui qui offre les meilleures performances pour notre tâche de prédiction. Nous avons utilisé une approche systématique pour comparer les modèles, en entraînant chaque modèle sur un ensemble d'entraînement et en évaluant ses performances sur un ensemble de test.

6.2 Critères de Sélection du Meilleur Modèle

Les critères utilisés pour sélectionner le meilleur modèle sont basés sur les performances de chaque modèle sur l'ensemble de test. Nous avons utilisé la métrique d'accuracy (la proportion de prédictions correctes faites par le modèle) comme principal critère de sélection. En plus de l'accuracy, nous avons également pris en considération d'autres critères tels que la capacité d'interprétation du modèle, le temps d'entraînement et d'inférence, ainsi que la robustesse par rapport aux données de test.

6.3 Résultats de la Comparaison des Modèles

Après avoir comparé les différents modèles à l'aide de la méthodologie décrite précédemment, nous avons obtenu les résultats suivants :

- Meilleur Classifieur : Forêt Aléatoire
- Précision (Accuracy) : 0.89
- Pire Classifieur : Naive Bayes
- Précision (Accuracy) : 0.78

Ces résultats montrent que la Forêt Aléatoire a obtenu la meilleure précision avec 0.89 sur l'ensemble de test, tandis que le Naive Bayes a obtenu la précision la plus basse avec 0.78.

6.4 Justification du Choix du Meilleur Modèle

La Forêt Aléatoire a été choisie comme le meilleur modèle en raison de sa précision élevée sur l'ensemble de test. De plus, la Forêt Aléatoire offre une bonne capacité d'interprétation des résultats et est relativement rapide à entraîner et à inférer par rapport à d'autres modèles plus complexes comme les réseaux de neurones. Ces caractéristiques en font un choix judicieux pour notre tâche de prédiction.

Cependant, il convient de noter que le choix du meilleur modèle dépend également des besoins spécifiques du projet et des contraintes telles que la disponibilité des données et les exigences de temps de calcul.

7 Conclusion

7.1 Récapitulatif des Principales Conclusions

Dans l'ensemble, notre étude a démontré l'efficacité de l'application de Machine Learning développée pour l'analyse des transactions dans le monde des métaverses. Nous avons réussi à créer une interface conviviale qui permet à l'utilisateur d'appliquer différents algorithmes de ML sur une seule dataset prétraitée. Les résultats obtenus ont montré que certains modèles performaient mieux que d'autres pour la tâche de prédiction des transactions.

7.2 Limitations de l'Étude

Une des principales limitations de notre étude est que l'application fonctionne uniquement sur une seule dataset pour le moment. Cela limite sa généralisation à d'autres jeux de données et réduit sa capacité à s'adapter à des problèmes similaires dans différents contextes. De plus, les performances des modèles pourraient être affectées par des facteurs tels que la qualité des données, la représentativité de l'échantillon, et la distribution des classes.

7.3 Possibilités d'Amélioration Future

Pour améliorer notre application, nous envisageons plusieurs pistes :

- **Acceptation de Datasets Utilisateur** : Nous prévoyons de rendre l'application plus dynamique en permettant aux utilisateurs de charger leurs propres datasets. Cela élargira la portée de l'application et la rendra plus adaptable à différents contextes.
- **Optimisation des Algorithmes** : Nous explorerons des techniques d'optimisation plus avancées pour améliorer les performances des modèles, telles que l'optimisation bayésienne, l'autoML, et l'optimisation multi-objectifs.
- **Extension des Fonctionnalités** : Nous ajouterons de nouvelles fonctionnalités à l'application, telles que la visualisation interactive des résultats, le suivi des performances des modèles sur plusieurs datasets, et la comparaison des performances avec des benchmarks externes.
- **Validation Externe** : Nous prévoyons de valider les performances de notre application sur des ensembles de données externes et dans des environnements réels pour confirmer sa robustesse et sa fiabilité.

En intégrant ces améliorations, nous visons à faire de notre application un outil encore plus puissant et polyvalent pour l'analyse des transactions dans les métaverses et d'autres domaines connexes de la finance décentralisée.