Compte Rendu - Mini Projet Python pour la Data Science

Amine Mimis, Mohssen Mohammed, Mamadou Tahiro Dialo, Achraf Alaabouch

Approche adoptée de Pré-traitement

Le dataset **Ransomwares_Goodwares_Dataset.xlsx** contient deux types d'échantillons :

- **G (Goodwares)** : logiciels sains,
- **R (Ransomwares)** : logiciels malveillants.

Les étapes de pré-traitement sont les suivantes :

1. **Suppression des doublons** avec `drop_duplicates()` pour éviter le biais.
2. **Vérification des valeurs manquantes** à l'aide de `isna()`.
3. **Encodage des étiquettes** : `Sample_Type` est convertie en valeurs numériques (0 pour G, 1 pour R).
4. **Normalisation des données** par `MinMaxScaler()` pour homogénéiser les échelles.
5. **Export du dataset** nettoyé sous forme Excel.

Évaluation du Dataset Nettoyé

Après nettoyage, le dataset contient uniquement des lignes uniques, sans valeurs manquantes ni bruit. Les variables explicatives ont été sélectionnées en excluant la variable cible `Sample_Type`. Le dataset a ensuite été divisé en :

— 80% pour l'entraînement,

— 20% pour le test.

Modélisation Machine Learning et Deep Learning

Nous avons testé plusieurs modèles pour la classification binaire (ransomware vs goodware).

1. Random Forest

— Classificateur d'arbres de décision agrégés,

— Avantage : robustesse face à l'overfitting et bonne capacité de généralisation,

— Résultats : précision d'environ **96%**.

2. Support Vector Machine (SVM)

— Modèle efficace pour les problèmes de classification avec marge maximale,

— Utilisation d'un noyau `RBF` pour gérer la non-linéarité,

— Résultats : précision d'environ **94%**.

3. Réseau de Neurones (Deep Learning avec Keras)

Architecture du modèle :

— Couche dense (128 neurones, `ReLU`),

— `Dropout`(0.3),

— Couche dense (64 neurones, `ReLU`),

— `Dropout`(0.3),

— Couche de sortie (1 neurone, `Sigmoid`).

Compilation :

— Perte : `binary_crossentropy`,

— Optimiseur : `Adam`,

— Métrique : `accuracy`.

Résultats :

— Précision atteinte : **97%**,

— Courbe ROC très performante avec AUC proche de 1,

— Rapport de classification : excellent rappel et F1-score.

Insights

— Le modèle Deep Learning offre la meilleure performance, suivi de près par Random Forest.

— SVM reste compétitif, mais un peu en retrait pour des données plus complexes.

— Les variables les plus informatives concernent les comportements système (ex. nombre d'appels API).

— Tous les modèles montrent une séparation claire entre les deux classes.

Conclusion

Ce projet a permis de comparer différents modèles de classification sur un jeu de données mêlant logiciels malveillants et sains. La meilleure précision a été atteinte avec le réseau de neurones, bien que les modèles Random Forest et SVM aient également montré de très bonnes performances. Le pré-traitement a joué un rôle crucial dans la qualité des résultats.

Perspectives :

— Ajouter d'autres modèles (XGBoost, LSTM sur logs temporels),

— Faire de l'explicabilité (SHAP, LIME),

— Appliquer le modèle en temps réel via une API.