Compte Rendu : Analyse de données et Modélisation ML sur Colab

Dua Fathallah

December 4, 2025

Contents

1 Introduction

Ce rapport explique comment utiliser un fichier stocké sur Google Drive depuis Google Colab, réaliser un prétraitement des données, une exploration des données (EDA) et une modélisation simple en Machine Learning pour analyser la fraude sur cartes de crédit.

2 Utilisation du fichier depuis Google Drive

2.1 Montage de Google Drive

```
from google.colab import drive
drive.mount('/content/drive')
```

2.2 Chargement du fichier

```
import pandas as pd
import zipfile

zip_path = "/content/drive/MyDrive/creditcard.csv.zip"

with zipfile.ZipFile(zip_path, 'r') as z:
    df = pd.read_csv(z.open(z.namelist()[0]))

df = df.drop_duplicates()
df = df.fillna(df.median(numeric_only=True))
```

3 Exploration des Données (EDA)

3.1 Distribution de la variable cible

```
import matplotlib.pyplot as plt

plt.figure(figsize=(6,4))
df['Class'].value_counts().plot(kind='bar')
plt.title("Distribution de la variable cible")
plt.xlabel("Classe")
plt.ylabel("Nombre d'observations")
plt.savefig("distribution_classe.png")
plt.show()
```

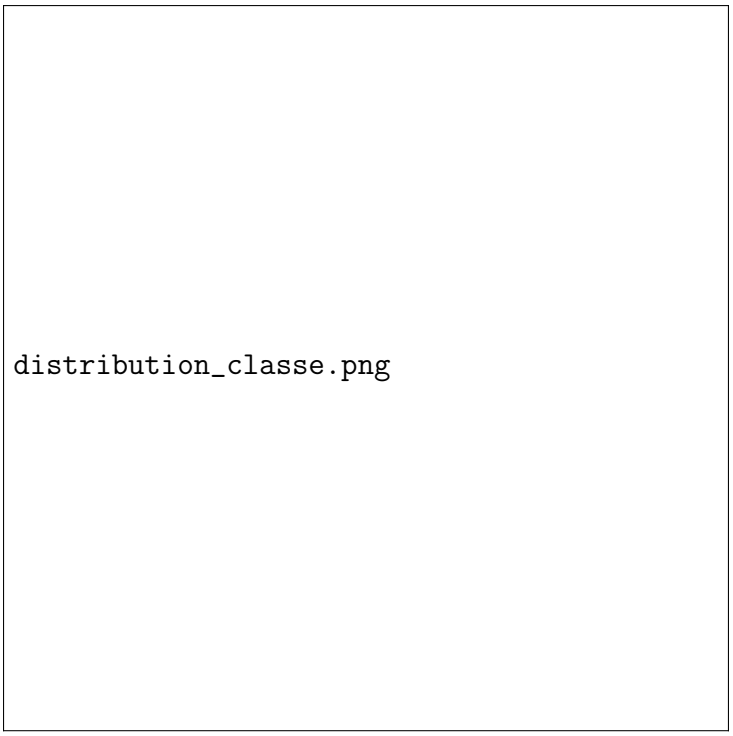

Figure 1: Répartition des transactions : fraude (1) vs normale (0)

Analyse : Le dataset est fortement déséquilibré. La majorité des transactions sont normales (Classe 0), et les fraudes (Classe 1) représentent une très petite proportion. Cela suggère qu'il faudra éventuellement considérer des techniques de rééquilibrage pour les modèles avancés.

3.2 Corrélation des variables

```
import seaborn as sns

plt.figure(figsize=(10,6))
sns.heatmap(df.corr(), cmap="coolwarm", annot=False)
plt.title("Matrice de corr lation")
plt.savefig("correlation_matrix.png")
plt.show()
```

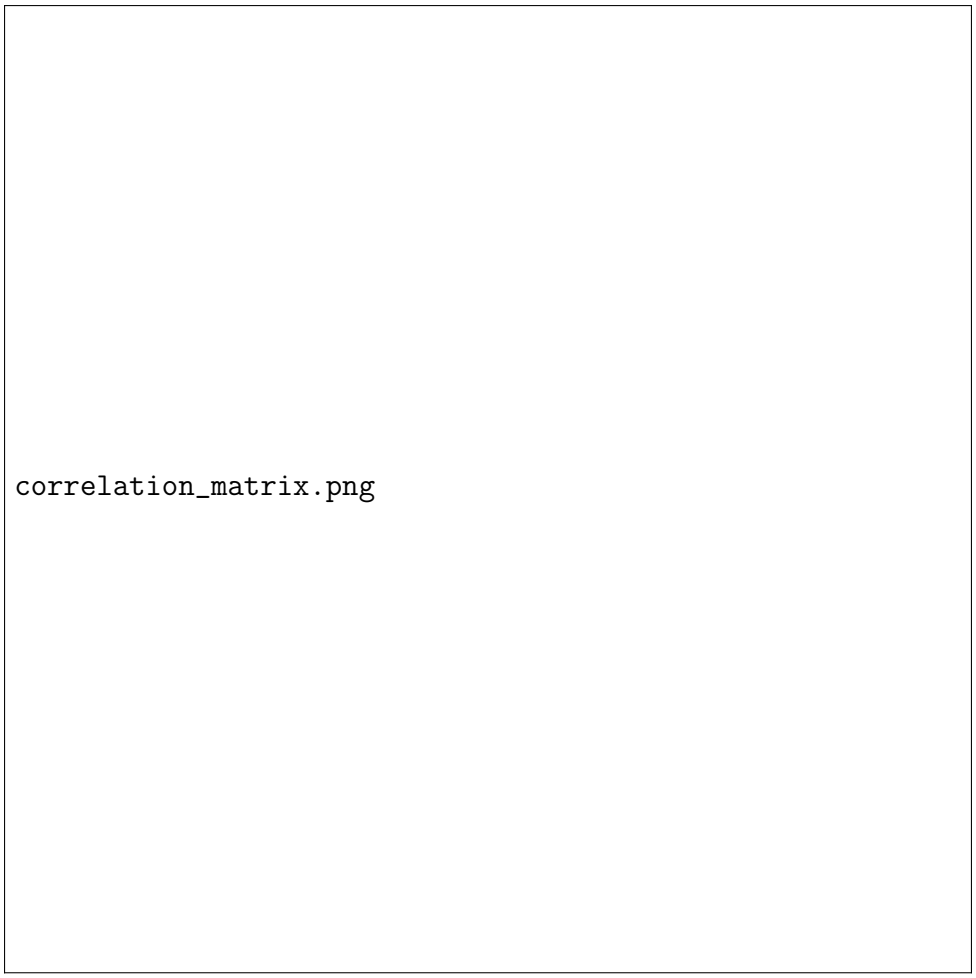

Figure 2: Matrice de corrélation des variables

Analyse : - Les variables sont anonymisées (V1 à V28), donc il est difficile d'interpréter chaque corrélation individuellement. - Cependant, certaines variables semblent légèrement corrélées avec la variable cible, ce qui peut aider le modèle de classification.

4 Modélisation Machine Learning

```
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import classification_report,
   confusion_matrix

X = df.drop("Class", axis=1)
y = df["Class"]

X_train, X_test, y_train, y_test = train_test_split(
    X, y, test_size=0.2, random_state=42, stratify=y
)

scaler = StandardScaler()
```

```
X_train = scaler.fit_transform(X_train)
X_test = scaler.transform(X_test)

model = LogisticRegression(max_iter=200)
model.fit(X_train, y_train)

y_pred = model.predict(X_test)

print("Classification Report :\n", classification_report(y_test,
    y_pred))
print("Confusion Matrix :\n", confusion_matrix(y_test, y_pred))
```

4.1 Analyse des résultats

- **Accuracy :** élevée mais peu informative en raison du déséquilibre. - **Precision et Recall :** plus pertinents pour détecter la fraude. - **Confusion Matrix :** montre le nombre de fraudes correctement identifiées (True Positives) vs manquées (False Negatives).

Remarque : Pour améliorer la détection des fraudes, on pourrait utiliser : - RandomForest ou XGBoost - Techniques de rééchantillonnage (SMOTE, undersampling) - Optimisation des hyperparamètres

5 Conclusion

- Le dataset est fortement déséquilibré, ce qui influence la performance des modèles. - La Logistic Regression simple détecte certaines fraudes, mais les modèles avancés et les techniques de rééquilibrage sont nécessaires pour de meilleures performances. - L'EDA a permis de visualiser la répartition de la classe cible et les corrélations entre variables. - Ce workflow est une base solide pour des analyses plus poussées sur Colab.