```python
import opendatasets as od
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import os
import seaborn as sns
from sklearn.preprocessing import StandardScaler
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import classification_report, confusion_matrix,
roc_auc_score
import xgboost as xgb
from xgboost import XGBClassifier

data_set = 'https://www.kaggle.com/datasets/mos3santos/deteco-de-
fraude-de-carto-de-crdito'
od.download(data_set)
```

```
Skipping, found downloaded files in ".\deteco-de-fraude-de-carto-de-
crdito" (use force=True to force download)
```

```python
data_dir = '.\deteco-de-fraude-de-carto-de-crdito'
file_path = os.path.join(data_dir, "creditcard.csv")
print("Chemin utilisé :", file_path)  # vérification
creditcard = pd.read_csv(file_path)
```

```
Chemin utilisé : .\deteco-de-fraude-de-carto-de-crdito\creditcard.csv
```

```python
display(creditcard.head())
```

```
   Time        V1        V2        V3        V4        V5        V6
V7  \
0   0.0 -1.359807 -0.072781  2.536347  1.378155 -0.338321  0.462388
0.239599
1   0.0  1.191857  0.266151  0.166480  0.448154  0.060018 -0.082361 -
0.078803
2   1.0 -1.358354 -1.340163  1.773209  0.379780 -0.503198  1.800499
0.791461
3   1.0 -0.966272 -0.185226  1.792993 -0.863291 -0.010309  1.247203
0.237609
4   2.0 -1.158233  0.877737  1.548718  0.403034 -0.407193  0.095921
0.592941

         V8        V9       V10       V11       V12       V13
V14  \
0  0.098698  0.363787  0.090794 -0.551600 -0.617801 -0.991390 -
0.311169
1  0.085102 -0.255425 -0.166974  1.612727  1.065235  0.489095 -
0.143772
2  0.247676 -1.514654  0.207643  0.624501  0.066084  0.717293 -
0.165946
```

```
3  0.377436 -1.387024 -0.054952 -0.226487  0.178228  0.507757 -
0.287924
4 -0.270533  0.817739  0.753074 -0.822843  0.538196  1.345852 -
1.119670

        V15       V16       V17       V18       V19       V20
V21  \
0  1.468177 -0.470401  0.207971  0.025791  0.403993  0.251412 -
0.018307
1  0.635558  0.463917 -0.114805 -0.183361 -0.145783 -0.069083 -
0.225775
2  2.345865 -2.890083  1.109969 -0.121359 -2.261857  0.524980
0.247998
3 -0.631418 -1.059647 -0.684093  1.965775 -1.232622 -0.208038 -
0.108300
4  0.175121 -0.451449 -0.237033 -0.038195  0.803487  0.408542 -
0.009431

        V22       V23       V24       V25       V26       V27
V28  \
0  0.277838 -0.110474  0.066928  0.128539 -0.189115  0.133558 -
0.021053
1 -0.638672  0.101288 -0.339846  0.167170  0.125895 -0.008983
0.014724
2  0.771679  0.909412 -0.689281 -0.327642 -0.139097 -0.055353 -
0.059752
3  0.005274 -0.190321 -1.175575  0.647376 -0.221929  0.062723
0.061458
4  0.798278 -0.137458  0.141267 -0.206010  0.502292  0.219422
0.215153

   Amount  Class  Amount_scaled
0  149.62      0       0.244964
1    2.69      0      -0.342475
2  378.66      0       1.160686
3  123.50      0       0.140534
4   69.99      0      -0.073403
```

```
print(creditcard.info())
```

```
Informations générales :
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 284807 entries, 0 to 284806
Data columns (total 32 columns):
 #   Column         Non-Null Count   Dtype
---  ------         --------------   -----
 0   Time           284807 non-null  float64
 1   V1             284807 non-null  float64
 2   V2             284807 non-null  float64
```

```
 3   V3              284807 non-null  float64
 4   V4              284807 non-null  float64
 5   V5              284807 non-null  float64
 6   V6              284807 non-null  float64
 7   V7              284807 non-null  float64
 8   V8              284807 non-null  float64
 9   V9              284807 non-null  float64
 10  V10             284807 non-null  float64
 11  V11             284807 non-null  float64
 12  V12             284807 non-null  float64
 13  V13             284807 non-null  float64
 14  V14             284807 non-null  float64
 15  V15             284807 non-null  float64
 16  V16             284807 non-null  float64
 17  V17             284807 non-null  float64
 18  V18             284807 non-null  float64
 19  V19             284807 non-null  float64
 20  V20             284807 non-null  float64
 21  V21             284807 non-null  float64
 22  V22             284807 non-null  float64
 23  V23             284807 non-null  float64
 24  V24             284807 non-null  float64
 25  V25             284807 non-null  float64
 26  V26             284807 non-null  float64
 27  V27             284807 non-null  float64
 28  V28             284807 non-null  float64
 29  Amount          284807 non-null  float64
 30  Class           284807 non-null  int64
 31  Amount_scaled   284807 non-null  float64
dtypes: float64(31), int64(1)
memory usage: 69.5 MB
None
```

```
print("\nStatistiques descriptives :")
display(creditcard.describe())
```

```
Statistiques descriptives :

                Time            V1            V2            V3
V4  \
count  284807.000000  2.848070e+05  2.848070e+05  2.848070e+05
2.848070e+05
mean    94813.859575  1.168375e-15  3.416908e-16 -1.379537e-15
2.074095e-15
std     47488.145955  1.958696e+00  1.651309e+00  1.516255e+00
1.415869e+00
min         0.000000 -5.640751e+01 -7.271573e+01 -4.832559e+01 -
5.683171e+00
25%     54201.500000 -9.203734e-01 -5.985499e-01 -8.903648e-01 -
```

```
8.486401e-01
50%      84692.000000  1.810880e-02  6.548556e-02  1.798463e-01 -
1.984653e-02
75%     139320.500000  1.315642e+00  8.037239e-01  1.027196e+00
7.433413e-01
max     172792.000000  2.454930e+00  2.205773e+01  9.382558e+00
1.687534e+01

                 V5            V6            V7            V8
V9  \
count  2.848070e+05  2.848070e+05  2.848070e+05  2.848070e+05
2.848070e+05
mean   9.604066e-16  1.487313e-15 -5.556467e-16  1.213481e-16 -
2.406331e-15
std    1.380247e+00  1.332271e+00  1.237094e+00  1.194353e+00
1.098632e+00
min   -1.137433e+02 -2.616051e+01 -4.355724e+01 -7.321672e+01 -
1.343407e+01
25%   -6.915971e-01 -7.682956e-01 -5.540759e-01 -2.086297e-01 -
6.430976e-01
50%   -5.433583e-02 -2.741871e-01  4.010308e-02  2.235804e-02 -
5.142873e-02
75%    6.119264e-01  3.985649e-01  5.704361e-01  3.273459e-01
5.971390e-01
max    3.480167e+01  7.330163e+01  1.205895e+02  2.000721e+01
1.559499e+01

       ...           V21           V22           V23           V24  \
count  ...  2.848070e+05  2.848070e+05  2.848070e+05  2.848070e+05
mean   ...  1.654067e-16 -3.568593e-16  2.578648e-16  4.473266e-15
std    ...  7.345240e-01  7.257016e-01  6.244603e-01  6.056471e-01
min    ... -3.483038e+01 -1.093314e+01 -4.480774e+01 -2.836627e+00
25%    ... -2.283949e-01 -5.423504e-01 -1.618463e-01 -3.545861e-01
50%    ... -2.945017e-02  6.781943e-03 -1.119293e-02  4.097606e-02
75%    ...  1.863772e-01  5.285536e-01  1.476421e-01  4.395266e-01
max    ...  2.720284e+01  1.050309e+01  2.252841e+01  4.584549e+00

                V25           V26           V27           V28
Amount  \
count  2.848070e+05  2.848070e+05  2.848070e+05  2.848070e+05
284807.000000
mean   5.340915e-16  1.683437e-15 -3.660091e-16 -1.227390e-16
88.349619
std    5.212781e-01  4.822270e-01  4.036325e-01  3.300833e-01
250.120109
min   -1.029540e+01 -2.604551e+00 -2.256568e+01 -1.543008e+01
0.000000
25%   -3.171451e-01 -3.269839e-01 -7.083953e-02 -5.295979e-02
5.600000
50%    1.659350e-02 -5.213911e-02  1.342146e-03  1.124383e-02
```

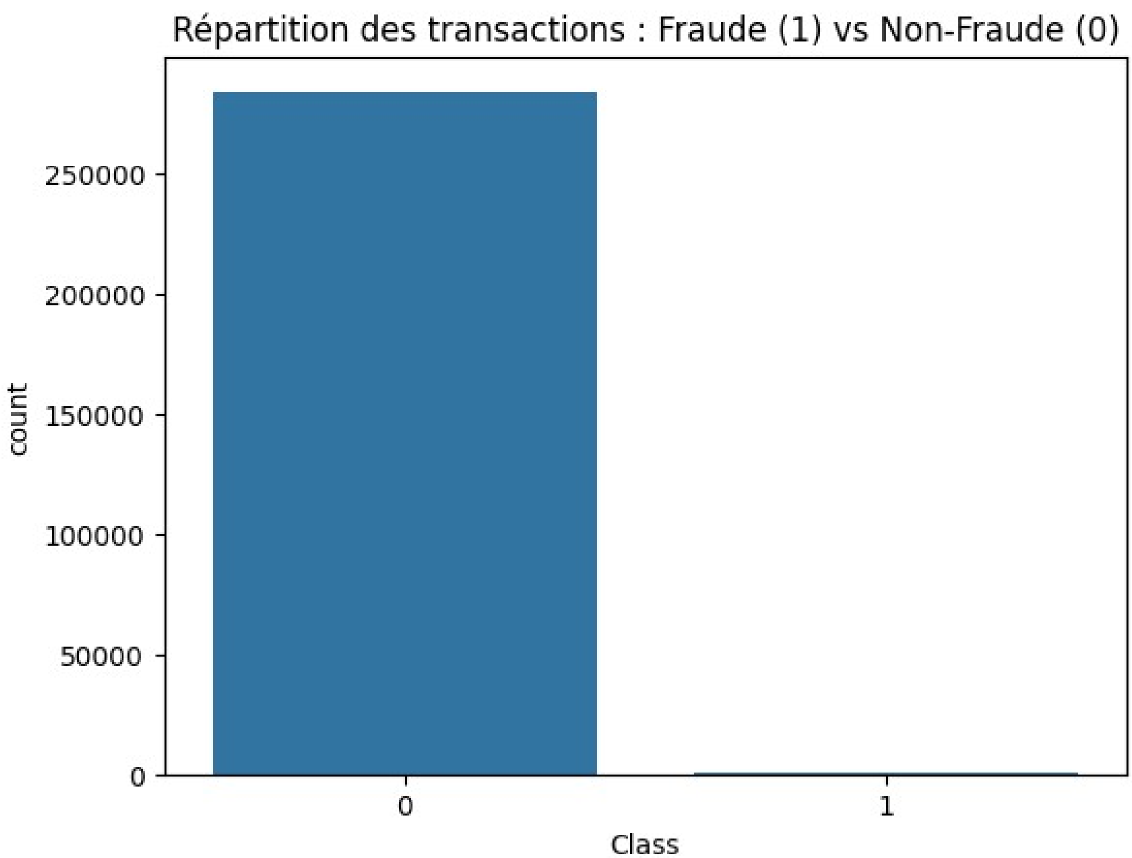
```
22.000000
75%    3.507156e-01  2.409522e-01  9.104512e-02  7.827995e-02
77.165000
max    7.519589e+00  3.517346e+00  3.161220e+01  3.384781e+01
25691.160000

               Class
count  284807.000000
mean        0.001727
std         0.041527
min         0.000000
25%         0.000000
50%         0.000000
75%         0.000000
max         1.000000

[8 rows x 31 columns]
```

```
val_nul = creditcard.isnull().sum()
missing_creditcard = pd.DataFrame({
    'Colonne': val_nul.index,
    'Valeurs manquantes': val_nul.values
})
print("\nValeurs manquantes par colonne :")
print(missing_creditcard)
```

```

Valeurs manquantes par colonne :
   Colonne  Valeurs manquantes
0     Time                   0
1       V1                   0
2       V2                   0
3       V3                   0
4       V4                   0
5       V5                   0
6       V6                   0
7       V7                   0
8       V8                   0
9       V9                   0
10     V10                   0
11     V11                   0
12     V12                   0
13     V13                   0
14     V14                   0
15     V15                   0
16     V16                   0
17     V17                   0
18     V18                   0
19     V19                   0
20     V20                   0
```

```
21     V21                      0
22     V22                      0
23     V23                      0
24     V24                      0
25     V25                      0
26     V26                      0
27     V27                      0
28     V28                      0
29  Amount                      0
30   Class                      0
```

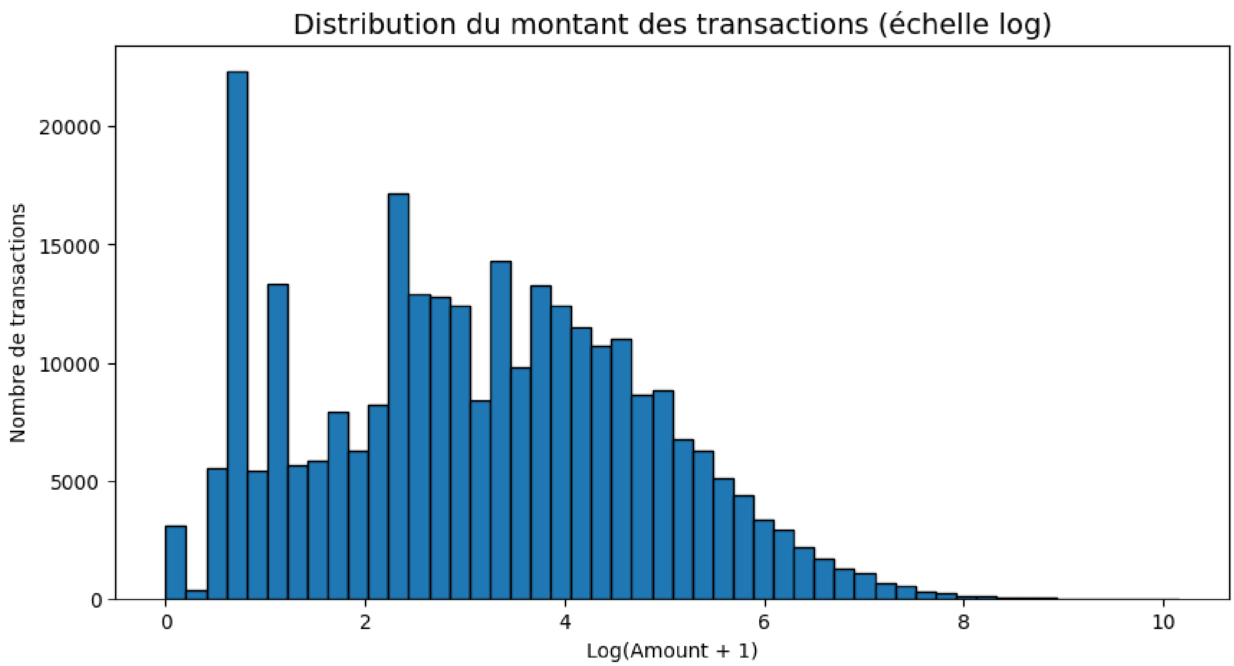
```
sns.countplot(x='Class', data=creditcard)
plt.title("Répartition des transactions : Fraude (1) vs Non-Fraude (0)")
plt.show()
print("Pourcentage de fraude :")
print(creditcard['Class'].value_counts(normalize=True)*100)
```



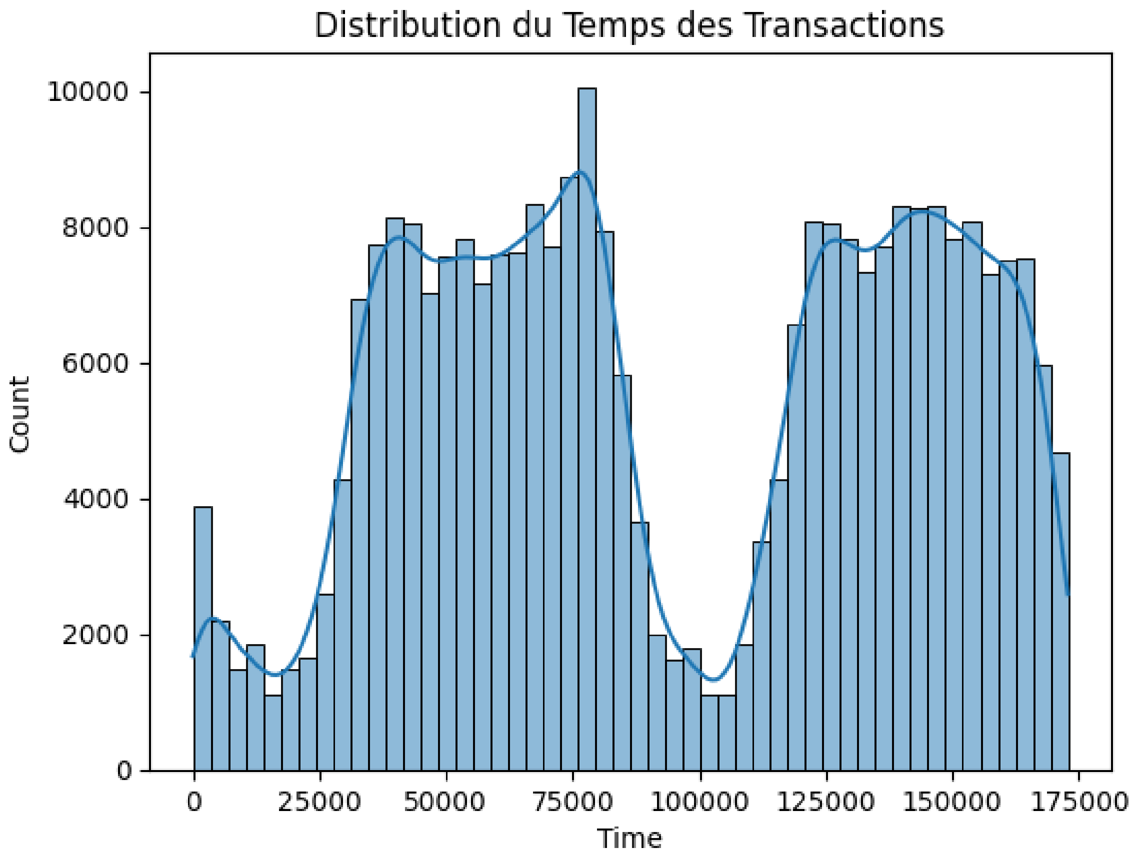

```
Pourcentage de fraude :
Class
0    99.827251
```

```
1       0.172749
Name: proportion, dtype: float64
```

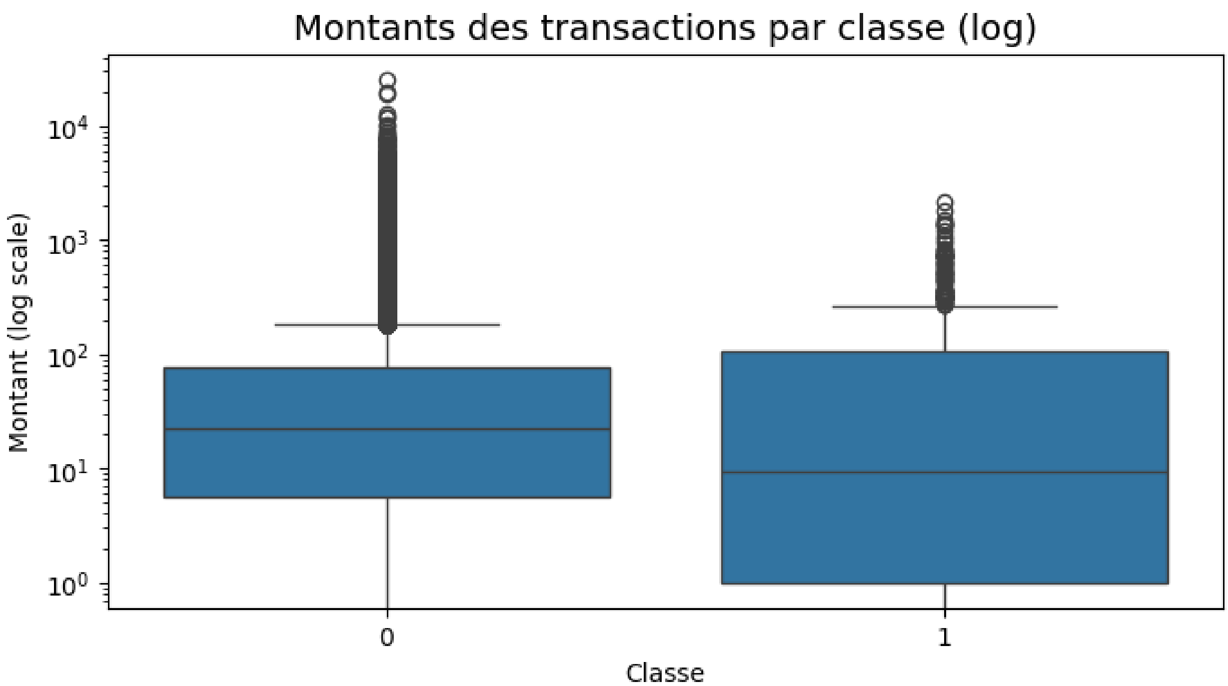
```
plt.figure(figsize=(10,5))
plt.hist(np.log1p(creditcard['Amount']), bins=50, color='#1f77b4',
edgecolor='k')
plt.title("Distribution du montant des transactions (échelle log)",
fontsize=14)
plt.xlabel("Log(Amount + 1)")
plt.ylabel("Nombre de transactions")
plt.show()
```




```
sns.histplot(creditcard['Time'], bins=50, kde=True)
plt.title("Distribution du Temps des Transactions")
plt.show()
```




```
plt.figure(figsize=(8,4))
sns.boxplot(x='Class', y='Amount', data=creditcard)
plt.yscale('log')
plt.title("Montants des transactions par classe (log)", fontsize=14)
plt.xlabel("Classe")
plt.ylabel("Montant (log scale)")
plt.show()
```




```
corr = creditcard.corr()
# Affichage des valeurs numériques
pd.set_option('display.max_columns', None)  # pour voir toutes les
colonnes
display(corr.round(2))  # arrondi à 2 décimales pour plus de
lisibilité
```

```
                  Time    V1    V2    V3    V4    V5    V6    V7    V8
V9  \
Time              1.00  0.12 -0.01 -0.42 -0.11  0.17 -0.06  0.08 -0.04 -
0.01
V1                0.12  1.00  0.00 -0.00 -0.00  0.00 -0.00 -0.00 -0.00 -
0.00
V2               -0.01  0.00  1.00  0.00 -0.00  0.00  0.00  0.00 -0.00
0.00
V3               -0.42 -0.00  0.00  1.00  0.00 -0.00  0.00  0.00 -0.00
0.00
V4               -0.11 -0.00 -0.00  0.00  1.00 -0.00 -0.00 -0.00  0.00
0.00
V5                0.17  0.00  0.00 -0.00 -0.00  1.00  0.00  0.00  0.00
0.00
V6               -0.06 -0.00  0.00  0.00 -0.00  0.00  1.00  0.00 -0.00
0.00
V7                0.08 -0.00  0.00  0.00 -0.00  0.00  0.00  1.00  0.00
0.00
V8               -0.04 -0.00 -0.00 -0.00  0.00  0.00 -0.00  0.00  1.00
0.00
V9               -0.01 -0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00
```

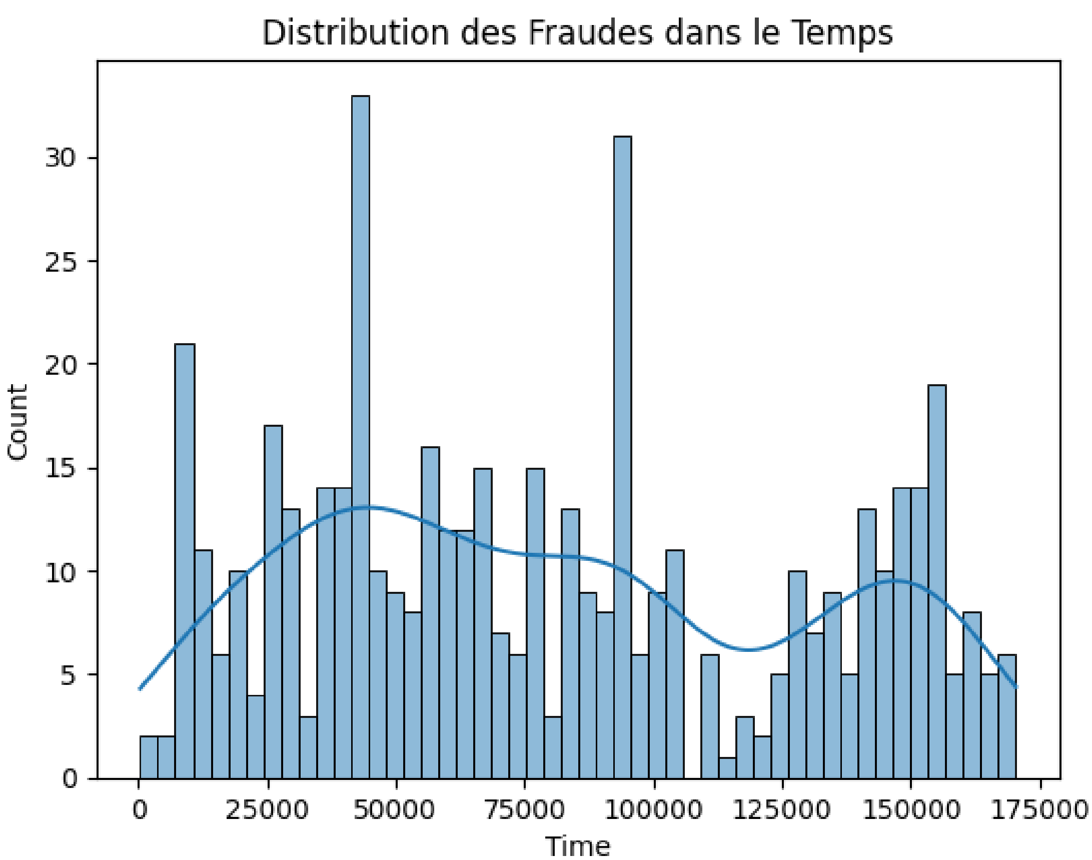
```
1.00
V10             0.03  0.00 -0.00  0.00  0.00 -0.00  0.00 -0.00 -0.00 -
0.00
V11            -0.25  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  
0.00
V12             0.12  0.00 -0.00  0.00 -0.00  0.00  0.00 -0.00  0.00 -
0.00
V13            -0.07 -0.00  0.00  0.00  0.00  0.00 -0.00  0.00 -0.00  
0.00
V14            -0.10 -0.00 -0.00  0.00  0.00  0.00  0.00  0.00 -0.00  
0.00
V15            -0.18  0.00 -0.00  0.00  0.00 -0.00 -0.00 -0.00  0.00 -
0.00
V16             0.01  0.00  0.00  0.00 -0.00  0.00  0.00  0.00 -0.00 -
0.00
V17            -0.07 -0.00 -0.00  0.00 -0.00  0.00  0.00  0.00 -0.00  
0.00
V18             0.09  0.00  0.00  0.00 -0.00  0.00  0.00  0.00 -0.00  
0.00
V19             0.03  0.00 -0.00  0.00 -0.00 -0.00 -0.00 -0.00 -0.00 -
0.00
V20            -0.05  0.00  0.00 -0.00 -0.00 -0.00 -0.00  0.00  0.00 -
0.00
V21             0.04 -0.00 -0.00  0.00 -0.00 -0.00  0.00 -0.00  0.00  
0.00
V22             0.14 -0.00  0.00 -0.00 -0.00  0.00 -0.00 -0.00  0.00 -
0.00
V23             0.05  0.00  0.00 -0.00  0.00 -0.00  0.00 -0.00  0.00 -
0.00
V24            -0.02 -0.00  0.00  0.00  0.00 -0.00 -0.00  0.00 -0.00 -
0.00
V25            -0.23 -0.00 -0.00 -0.00  0.00  0.00  0.00 -0.00 -0.00  
0.00
V26            -0.04 -0.00  0.00 -0.00 -0.00  0.00 -0.00 -0.00 -0.00 -
0.00
V27            -0.01  0.00 -0.00  0.00  0.00  0.00 -0.00 -0.00  0.00 -
0.00
V28            -0.01  0.00 -0.00  0.00 -0.00 -0.00  0.00 -0.00 -0.00  
0.00
Amount         -0.01 -0.23 -0.53 -0.21  0.10 -0.39  0.22  0.40 -0.10 -
0.04
Class          -0.01 -0.10  0.09 -0.19  0.13 -0.09 -0.04 -0.19  0.02 -
0.10
Amount_scaled -0.01 -0.23 -0.53 -0.21  0.10 -0.39  0.22  0.40 -0.10 -
0.04

                 V10   V11   V12   V13   V14   V15   V16   V17   V18  
V19  \
Time            0.03 -0.25  0.12 -0.07 -0.10 -0.18  0.01 -0.07  0.09
```

```
0.03
V1              0.00  0.00  0.00 -0.00 -0.00  0.00  0.00 -0.00  0.00
0.00
V2             -0.00  0.00 -0.00  0.00 -0.00 -0.00  0.00 -0.00  0.00 -
0.00
V3              0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00
0.00
V4              0.00  0.00 -0.00  0.00  0.00  0.00 -0.00 -0.00 -0.00 -
0.00
V5             -0.00  0.00  0.00  0.00  0.00 -0.00  0.00  0.00  0.00 -
0.00
V6              0.00  0.00  0.00 -0.00  0.00 -0.00  0.00  0.00  0.00 -
0.00
V7             -0.00  0.00 -0.00  0.00  0.00 -0.00  0.00  0.00  0.00 -
0.00
V8             -0.00  0.00  0.00 -0.00 -0.00  0.00 -0.00 -0.00 -0.00 -
0.00
V9             -0.00  0.00 -0.00  0.00  0.00 -0.00 -0.00  0.00  0.00 -
0.00
V10             1.00 -0.00  0.00 -0.00  0.00  0.00  0.00  0.00  0.00
0.00
V11            -0.00  1.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00 -
0.00
V12             0.00  0.00  1.00 -0.00  0.00 -0.00  0.00  0.00  0.00
0.00
V13            -0.00  0.00 -0.00  1.00  0.00  0.00  0.00  0.00  0.00 -
0.00
V14             0.00  0.00  0.00  0.00  1.00 -0.00 -0.00  0.00  0.00
0.00
V15             0.00  0.00 -0.00  0.00 -0.00  1.00  0.00  0.00  0.00 -
0.00
V16             0.00  0.00  0.00  0.00 -0.00  0.00  1.00  0.00 -0.00
0.00
V17             0.00  0.00  0.00  0.00  0.00  0.00  0.00  1.00 -0.00 -
0.00
V18             0.00  0.00  0.00  0.00  0.00  0.00 -0.00 -0.00  1.00 -
0.00
V19             0.00 -0.00  0.00 -0.00  0.00 -0.00  0.00 -0.00 -0.00
1.00
V20            -0.00 -0.00  0.00  0.00 -0.00  0.00  0.00 -0.00 -0.00
0.00
V21             0.00 -0.00  0.00  0.00 -0.00  0.00 -0.00 -0.00 -0.00
0.00
V22            -0.00  0.00  0.00  0.00  0.00 -0.00 -0.00 -0.00 -0.00 -
0.00
V23             0.00 -0.00  0.00 -0.00  0.00 -0.00  0.00  0.00 -0.00
0.00
V24            -0.00  0.00  0.00 -0.00  0.00 -0.00 -0.00 -0.00 -0.00
0.00
```

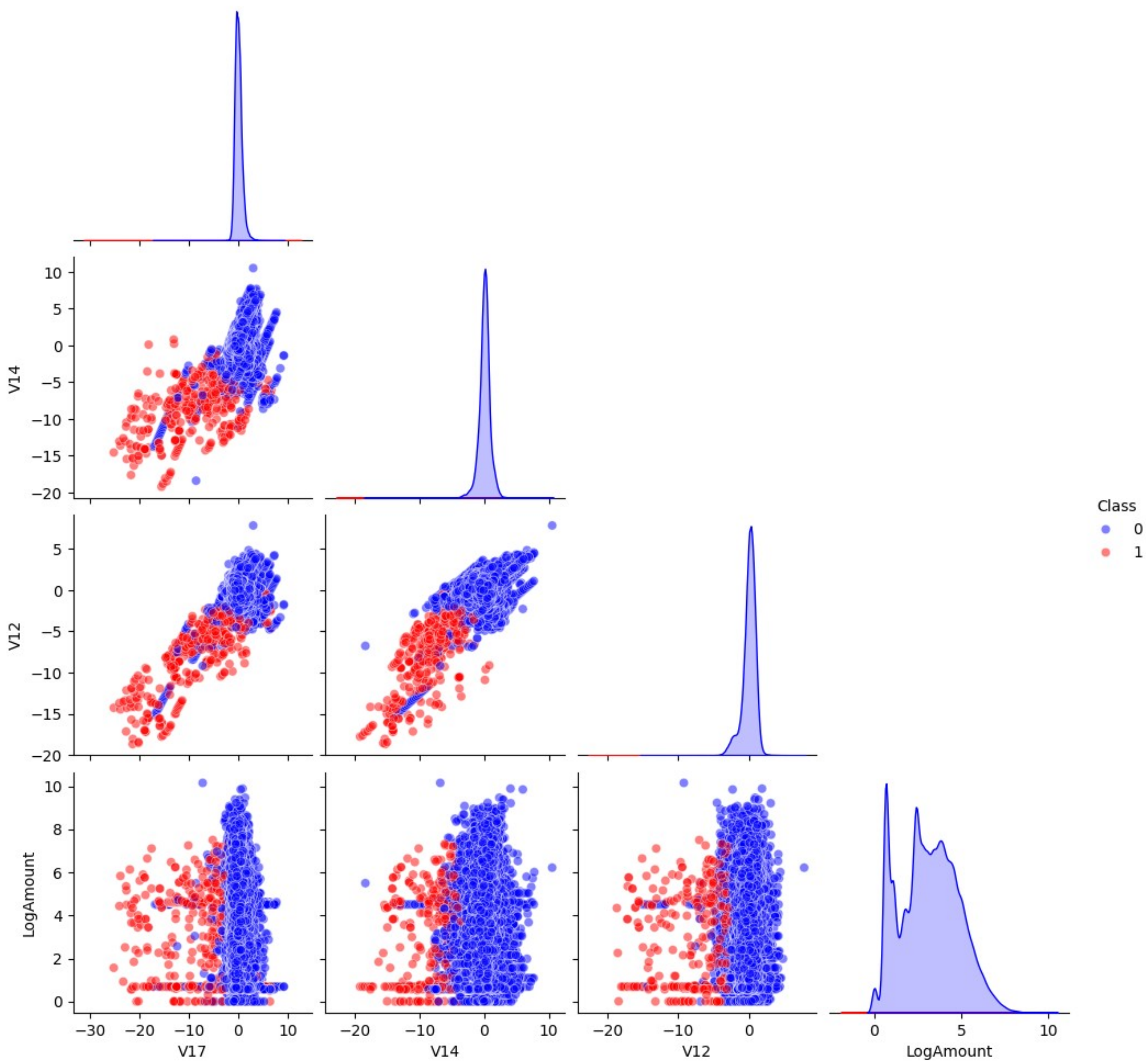
```
V25           -0.00 -0.00 -0.00 -0.00 -0.00  0.00 -0.00  0.00 -0.00
0.00
V26           -0.00 -0.00 -0.00 -0.00 -0.00  0.00 -0.00  0.00  0.00
0.00
V27           -0.00 -0.00 -0.00 -0.00  0.00 -0.00  0.00  0.00  0.00 -
0.00
V28            0.00 -0.00  0.00  0.00  0.00 -0.00  0.00 -0.00  0.00 -
0.00
Amount        -0.10  0.00 -0.01  0.01  0.03 -0.00 -0.00  0.01  0.04 -
0.06
Class         -0.22  0.15 -0.26 -0.00 -0.30 -0.00 -0.20 -0.33 -0.11
0.03
Amount_scaled -0.10  0.00 -0.01  0.01  0.03 -0.00 -0.00  0.01  0.04 -
0.06

                V20   V21   V22   V23   V24   V25   V26   V27   V28
Amount  \
Time          -0.05  0.04  0.14  0.05 -0.02 -0.23 -0.04 -0.01 -0.01
-0.01
V1             0.00 -0.00 -0.00  0.00 -0.00 -0.00 -0.00  0.00  0.00
-0.23
V2             0.00 -0.00  0.00  0.00  0.00 -0.00  0.00 -0.00 -0.00
-0.53
V3            -0.00  0.00 -0.00 -0.00  0.00 -0.00 -0.00  0.00  0.00
-0.21
V4            -0.00 -0.00 -0.00  0.00  0.00  0.00 -0.00  0.00 -0.00
0.10
V5            -0.00 -0.00  0.00 -0.00 -0.00  0.00  0.00  0.00 -0.00
-0.39
V6            -0.00  0.00 -0.00  0.00 -0.00  0.00 -0.00 -0.00  0.00
0.22
V7             0.00 -0.00 -0.00 -0.00  0.00 -0.00 -0.00 -0.00 -0.00
0.40
V8             0.00  0.00  0.00  0.00 -0.00 -0.00 -0.00  0.00 -0.00
-0.10
V9            -0.00  0.00 -0.00 -0.00 -0.00  0.00 -0.00 -0.00  0.00
-0.04
V10           -0.00  0.00 -0.00  0.00 -0.00 -0.00 -0.00 -0.00  0.00
-0.10
V11           -0.00 -0.00  0.00 -0.00  0.00 -0.00 -0.00 -0.00 -0.00
0.00
V12            0.00  0.00  0.00  0.00  0.00 -0.00 -0.00 -0.00  0.00
-0.01
V13            0.00  0.00  0.00 -0.00 -0.00 -0.00 -0.00 -0.00  0.00
0.01
V14           -0.00 -0.00  0.00  0.00  0.00 -0.00 -0.00  0.00  0.00
0.03
V15            0.00  0.00 -0.00 -0.00 -0.00  0.00  0.00 -0.00 -0.00
-0.00
```

```
V16             0.00 -0.00 -0.00  0.00 -0.00 -0.00 -0.00  0.00  0.00
-0.00
V17            -0.00 -0.00 -0.00  0.00 -0.00  0.00  0.00  0.00 -0.00
0.01
V18            -0.00 -0.00 -0.00 -0.00 -0.00 -0.00  0.00  0.00  0.00
0.04
V19             0.00  0.00 -0.00  0.00  0.00  0.00  0.00 -0.00 -0.00
-0.06
V20             1.00 -0.00  0.00  0.00  0.00  0.00 -0.00 -0.00 -0.00
0.34
V21            -0.00  1.00  0.00  0.00  0.00 -0.00 -0.00 -0.00  0.00
0.11
V22             0.00  0.00  1.00 -0.00  0.00 -0.00 -0.00  0.00 -0.00
-0.06
V23             0.00  0.00 -0.00  1.00  0.00 -0.00  0.00  0.00  0.00
-0.11
V24             0.00  0.00  0.00  0.00  1.00  0.00  0.00 -0.00 -0.00
0.01
V25             0.00 -0.00 -0.00 -0.00  0.00  1.00  0.00 -0.00 -0.00
-0.05
V26            -0.00 -0.00 -0.00  0.00  0.00  0.00  1.00 -0.00 -0.00
-0.00
V27            -0.00 -0.00  0.00  0.00 -0.00 -0.00 -0.00  1.00 -0.00
0.03
V28            -0.00  0.00 -0.00  0.00 -0.00 -0.00 -0.00 -0.00  1.00
0.01
Amount          0.34  0.11 -0.06 -0.11  0.01 -0.05 -0.00  0.03  0.01
1.00
Class           0.02  0.04  0.00 -0.00 -0.01  0.00  0.00  0.02  0.01
0.01
Amount_scaled   0.34  0.11 -0.06 -0.11  0.01 -0.05 -0.00  0.03  0.01
1.00

               Class  Amount_scaled
Time           -0.01          -0.01
V1             -0.10          -0.23
V2              0.09          -0.53
V3             -0.19          -0.21
V4              0.13           0.10
V5             -0.09          -0.39
V6             -0.04           0.22
V7             -0.19           0.40
V8              0.02          -0.10
V9             -0.10          -0.04
V10            -0.22          -0.10
V11             0.15           0.00
V12            -0.26          -0.01
V13            -0.00           0.01
V14            -0.30           0.03
```

```
V15             -0.00          -0.00
V16             -0.20          -0.00
V17             -0.33           0.01
V18             -0.11           0.04
V19              0.03          -0.06
V20              0.02           0.34
V21              0.04           0.11
V22              0.00          -0.06
V23             -0.00          -0.11
V24             -0.01           0.01
V25              0.00          -0.05
V26              0.00          -0.00
V27              0.02           0.03
V28              0.01           0.01
Amount           0.01           1.00
Class            1.00           0.01
Amount_scaled    0.01           1.00

fraud = creditcard[creditcard['Class']==1]
sns.histplot(fraud['Time'], bins=50, kde=True)
plt.title("Distribution des Fraudes dans le Temps")
plt.show()
```



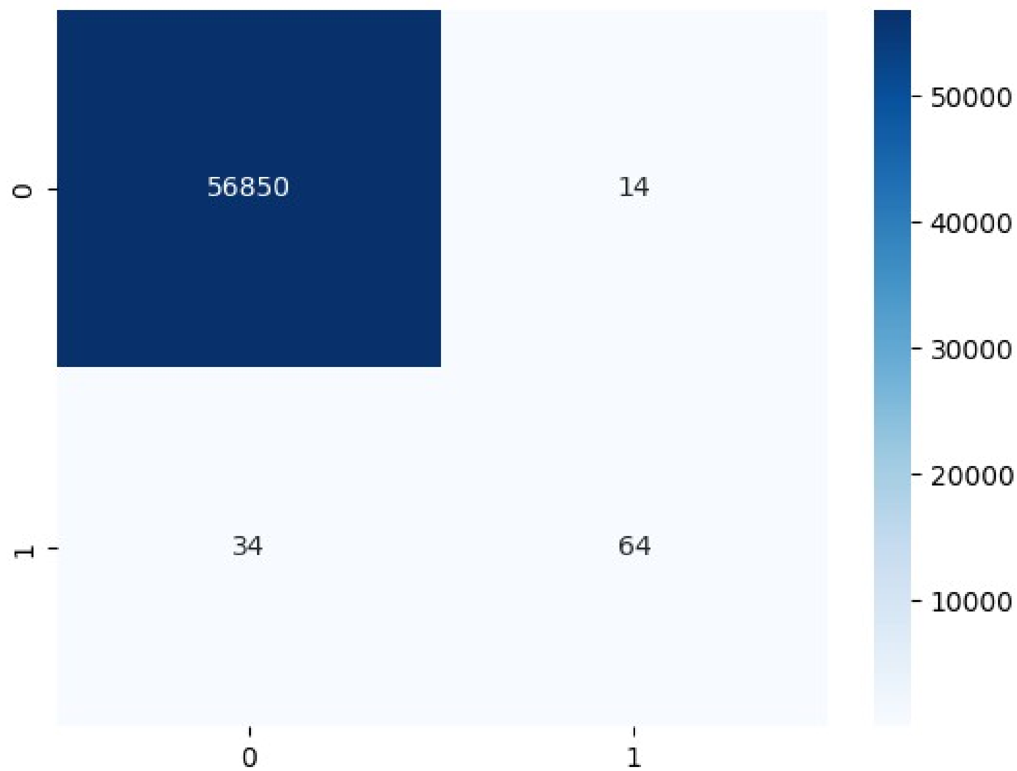
```python
creditcard_plot = creditcard.copy()

# Log-transformer la variable Amount
creditcard_plot['LogAmount'] = np.log1p(creditcard_plot['Amount'])

# Colonnes à visualiser
cols = ['V17','V14','V12','LogAmount','Class']

# Pairplot avec corner=True pour éviter les doublons
sns.pairplot(creditcard_plot[cols], hue='Class', diag_kind='kde',
corner=True, palette={0:'blue', 1:'red'}, plot_kws={'alpha':0.5})
plt.show()
```



<!DOCTYPE html> Interprétation des clusters de transactions body { font-family: Arial, sans-serif; line-height: 1.6; margin: 20px; } h2 { color: #2c3e50; } ul { margin-bottom: 20px; } .blue { color: blue; font-weight: bold; } .red { color: red; font-weight: bold; }

```
creditcard['Amount_scaled'] =
StandardScaler().fit_transform(creditcard['Amount'].values.reshape(-
1,1))
X = creditcard.drop(['Class','Amount'], axis=1)
y = creditcard['Class']

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

model = LogisticRegression(max_iter=5000)
model.fit(X_train, y_train)
y_pred = model.predict(X_test)

print("Classification Report :")
print(classification_report(y_test, y_pred))
print("\nConfusion Matrix :")
sns.heatmap(confusion_matrix(y_test, y_pred), annot=True, fmt='d',
cmap='Blues')
plt.show()
print("\nROC-AUC :", roc_auc_score(y_test, model.predict_proba(X_test)
[:,1]))
```

```
Classification Report :
              precision    recall  f1-score   support

           0       1.00      1.00      1.00     56864
           1       0.82      0.65      0.73        98

    accuracy                           1.00     56962
   macro avg       0.91      0.83      0.86     56962
weighted avg       1.00      1.00      1.00     56962


Confusion Matrix :
```




```
ROC-AUC : 0.9534650523124275
```

Le modèle est très performant pour détecter les transactions normales (56850/56864 correctes). Il détecte correctement 63 fraudes sur 98 (environ 64% recall), mais rate encore 30 fraudes. Le nombre de faux positifs est très faible (14 transactions normales classées comme fraude), ce qui explique la precision élevée. En résumé : le modèle a un excellent pouvoir discriminant, mais peut encore être amélioré

```
xgb_clf = XGBClassifier(
    n_estimators=300,
    max_depth=6,
    learning_rate=0.1,
    subsample=0.8,
    colsample_bytree=0.8,
    scale_pos_weight=(y.value_counts()[0] / y.value_counts()[1]),  #
équilibrage classes
    random_state=42,
    eval_metric="logloss"
)

xgb_clf.fit(X_train, y_train)

XGBClassifier(base_score=None, booster=None, callbacks=None,
              colsample_bylevel=None, colsample_bynode=None,
```

```
                colsample_bytree=0.8, device=None,
early_stopping_rounds=None,
                enable_categorical=False, eval_metric='logloss',
                feature_types=None, feature_weights=None, gamma=None,
                grow_policy=None, importance_type=None,
                interaction_constraints=None, learning_rate=0.1,
max_bin=None,
                max_cat_threshold=None, max_cat_to_onehot=None,
                max_delta_step=None, max_depth=6, max_leaves=None,
                min_child_weight=None, missing=nan,
monotone_constraints=None,
                multi_strategy=None, n_estimators=300, n_jobs=None,
                num_parallel_tree=None, ...)

y_pred = xgb_clf.predict(X_test)
y_proba = xgb_clf.predict_proba(X_test)[:,1]

print("\nConfusion Matrix :")
print(confusion_matrix(y_test, y_pred))
print("\nClassification Report :")
print(classification_report(y_test, y_pred, digits=4))
print("\nROC-AUC Score :")
print(roc_auc_score(y_test, y_proba))


Confusion Matrix :
[[56853    11]
 [   16    82]]

Classification Report :
              precision    recall  f1-score   support

           0     0.9997    0.9998    0.9998     56864
           1     0.8817    0.8367    0.8586        98

    accuracy                         0.9995     56962
   macro avg     0.9407    0.9183    0.9292     56962
weighted avg     0.9995    0.9995    0.9995     56962


ROC-AUC Score :
0.9725357961136057
```