```
import kagglehub
path = kagglehub.dataset_download("cristobaltudela/credit-card-transaction-l
```

```
n outdated `kagglehub` version (installed: 0.3.13), please consider upgrading
e.com/api/v1/datasets/download/cristobaltudela/credit-card-transaction-legiti
0<00:00, 82.5MB/s]Extracting files...
```

```
from google.colab import files
uploaded = files.upload()
```

Choose Files fraud_detection.csv

fraud_detection.csv(text/csv) - 324710 bytes, last modified: 1/30/2026 - 100% done

```
Saving fraud_detection.csv to fraud_detection.csv
```

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns

from sklearn.preprocessing import StandardScaler, LabelEncoder
from sklearn.ensemble import IsolationForest
```

New section

```
#Step 2. Load data
df = pd.read_csv("/content/fraud_detection.csv")
df.head()
#Step 3. Basic checks
df.shape
print(df.columns)
#df['TransactionID'].value_counts() # This line caused the error. After insp
```

```
Index(['TransactionID', 'AccountID', 'TransactionAmount', 'TransactionDate',
       'TransactionType', 'Location', 'DeviceID', 'IP Address', 'MerchantID',
       'Channel', 'CustomerAge', 'CustomerOccupation', 'TransactionDuration',
       'LoginAttempts', 'AccountBalance', 'PreviousTransactionDate',
       'is_outlier'],
      dtype='object')
```

```
# 3. Missing value percentage
missing_pct = df.isnull().mean() * 100
print("\nMissing Value Percentage (%):")
print(missing_pct)
```

Show hidden output

```
cat_cols = [
    'TransactionType', 'Location', 'DeviceID',
    'MerchantID', 'Channel', 'CustomerOccupation'
]

le = LabelEncoder()
for col in cat_cols:
    df[col] = le.fit_transform(df[col])
```

```
print(df.columns)
```

```
Index(['TransactionID', 'AccountID', 'TransactionAmount', 'TransactionDate',
       'TransactionType', 'Location', 'DeviceID', 'IP Address', 'MerchantID',
       'Channel', 'CustomerAge', 'CustomerOccupation', 'TransactionDuration',
       'LoginAttempts', 'AccountBalance', 'PreviousTransactionDate'],
      dtype='object')
```

```
df['TransactionDate'] = pd.to_datetime(df['TransactionDate'])
df['PreviousTransactionDate'] = pd.to_datetime(df['PreviousTransactionDate']

df['DaysSinceLastTransaction'] = (
    df['TransactionDate'] - df['PreviousTransactionDate']
).dt.days
```

```
df.drop(['TransactionDate', 'PreviousTransactionDate'], axis=1, inplace=True
```

```
info = df.info()
head = df.head()
```

Show hidden output

```
scaler = StandardScaler()

# Identify columns that are not suitable for numerical scaling
# These include identifier columns (TransactionID, AccountID, IP Address) ar
columns_to_exclude = ['TransactionID', 'AccountID', 'IP Address', 'is_outlie
```

```
# Create a subset of the DataFrame containing only the columns to be scaled
df_to_scale = df.drop(columns=columns_to_exclude)

df_scaled = scaler.fit_transform(df_to_scale)
```

```
iso = IsolationForest(
    n_estimators=200,
    contamination=0.01,   # assume 1% fraud
    random_state=42
)

df['Anomaly'] = iso.fit_predict(df_scaled)
```

```
df['FraudFlag'] = df['Anomaly'].map({1: 0, -1: 1})
df['FraudFlag'].value_counts()
```

```
           count
FraudFlag
    0       2486
    1         26

dtype: int64
```

```
from sklearn.neighbors import LocalOutlierFactor

lof = LocalOutlierFactor(contamination=0.01)
df['Fraud_LOF'] = lof.fit_predict(df_scaled)
```

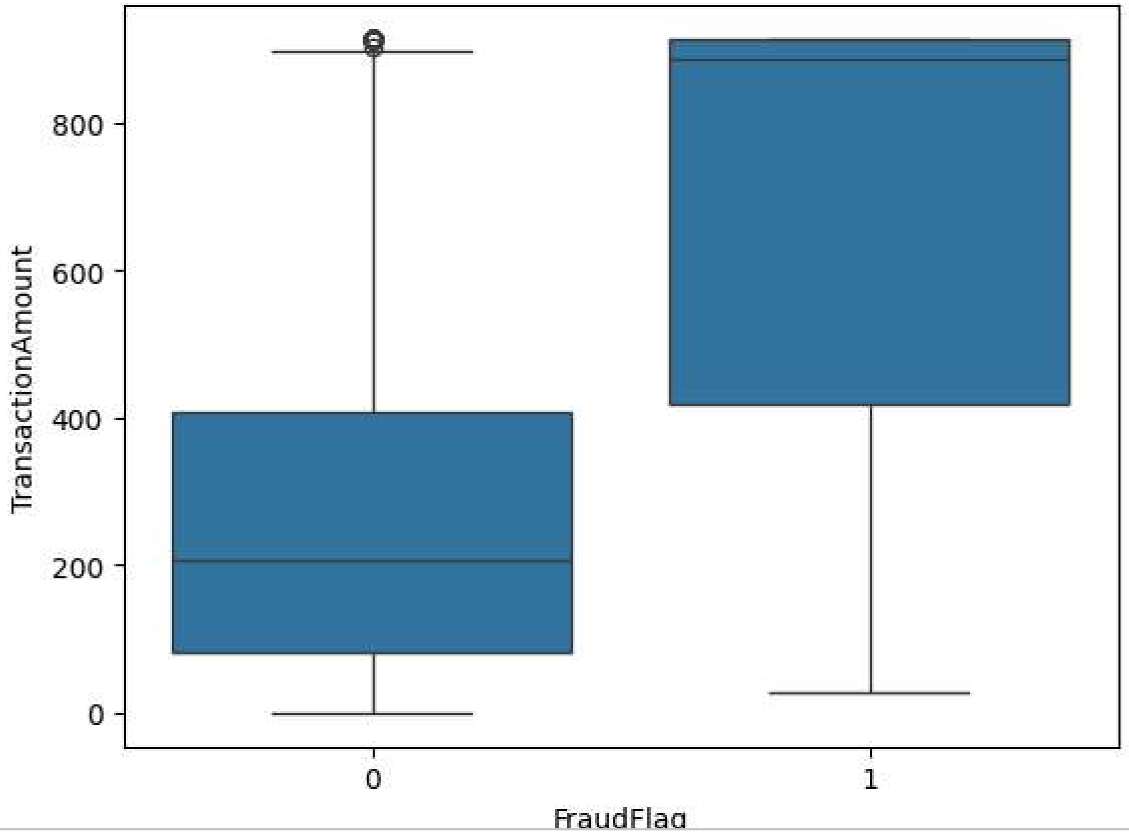
```
iso = IsolationForest(
    n_estimators=200,
    contamination=0.01,   # assume 1% fraud
    random_state=42
)

df['Anomaly'] = iso.fit_predict(df_scaled)
```

```
df['FraudFlag'] = df['Anomaly'].map({1: 0, -1: 1})
df['FraudFlag'].value_counts()
```

FraudFlag	count
0	2486
1	26

dtype: int64

```
sns.boxplot(x='FraudFlag', y='TransactionAmount', data=df)
plt.show()
```

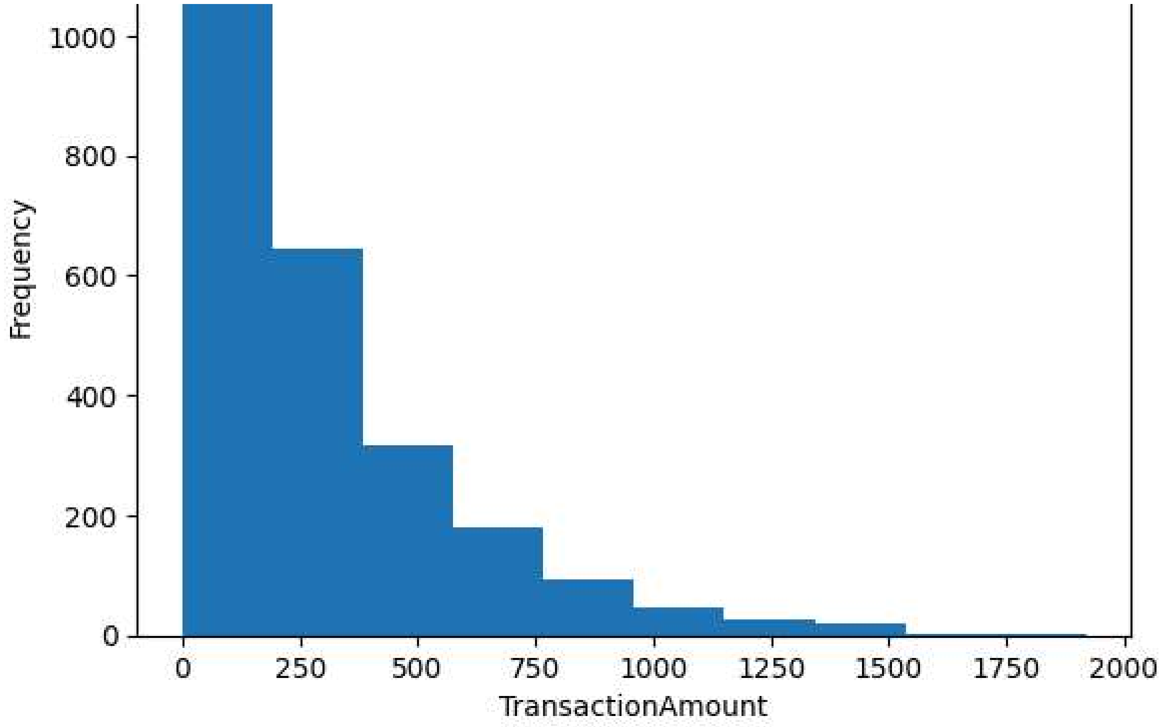


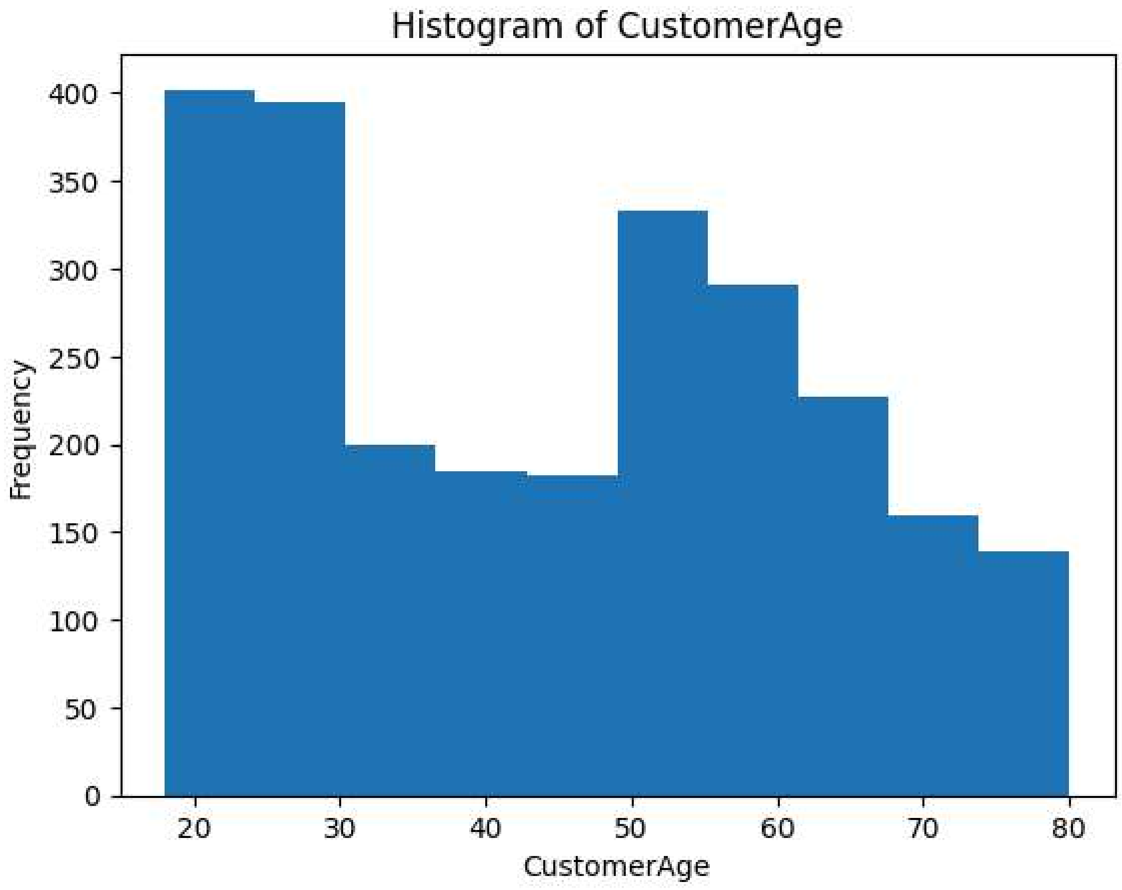
```
# 2. Descriptive statistics
print("\nDescriptive Statistics:")
print(df.describe(include="all"))
```

```
Descriptive Statistics:
       TransactionID AccountID  TransactionAmount   TransactionDate  \
count           2512      2512        2512.000000              2512
unique          2512       495                NaN              2405
top         TX002496   AC00460                NaN  11/20/2023 16:29
freq               1        12                NaN                 3
mean             NaN       NaN         297.593778               NaN
std              NaN       NaN         291.946243               NaN
min              NaN       NaN           0.260000               NaN
25%              NaN       NaN          81.885000               NaN
```

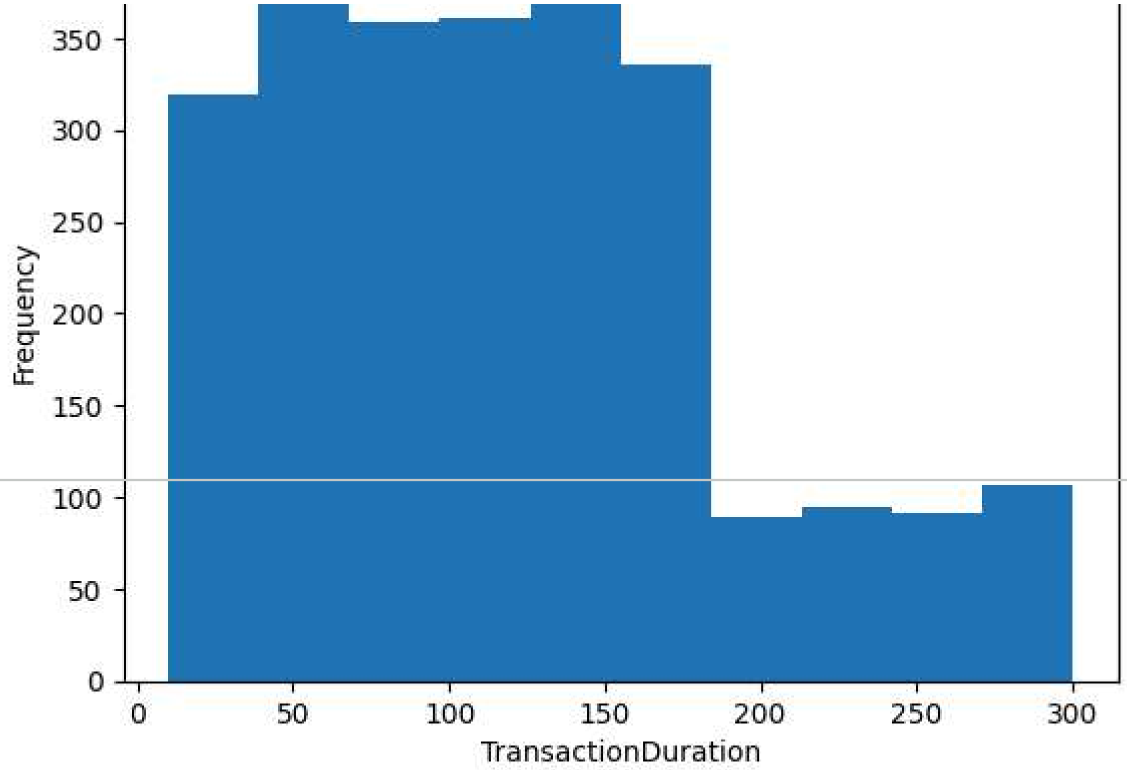
```
50%              NaN        NaN         211.140000                NaN
75%              NaN        NaN         414.527500                NaN
max              NaN        NaN        1919.110000                NaN

       TransactionType    Location DeviceID      IP Address MerchantID  \
count             2512        2512     2512            2512       2512
unique               2          43      681             592        100
top              Debit  Fort Worth  D000548  200.136.146.93       M026
freq              1944          70        9              13         45
mean               NaN         NaN      NaN             NaN        NaN
std                NaN         NaN      NaN             NaN        NaN
min                NaN         NaN      NaN             NaN        NaN
25%                NaN         NaN      NaN             NaN        NaN
50%                NaN         NaN      NaN             NaN        NaN
75%                NaN         NaN      NaN             NaN        NaN
max                NaN         NaN      NaN             NaN        NaN

       Channel  CustomerAge CustomerOccupation  TransactionDuration  \
count     2512  2512.000000               2512          2512.000000
unique       3          NaN                  4                  NaN
top     Branch          NaN            Student                  NaN
freq       868          NaN                657                  NaN
mean       NaN    44.673965                NaN           119.643312
std        NaN    17.792198                NaN            69.963757
min        NaN    18.000000                NaN            10.000000
25%        NaN    27.000000                NaN            63.000000
50%        NaN    45.000000                NaN           112.500000
75%        NaN    59.000000                NaN           161.000000
max        NaN    80.000000                NaN           300.000000

        LoginAttempts  AccountBalance PreviousTransactionDate
count     2512.000000     2512.000000                    2512
unique            NaN             NaN                       7
top               NaN             NaN          11/4/2024 8:07
freq              NaN             NaN                     435
mean         1.124602     5114.302966                     NaN
std          0.602662     3900.942499                     NaN
min          1.000000      101.250000                     NaN
25%          1.000000     1504.370000                     NaN
50%          1.000000     4735.510000                     NaN
75%          1.000000     7678.820000                     NaN
max          5.000000    14977.990000                     NaN
```

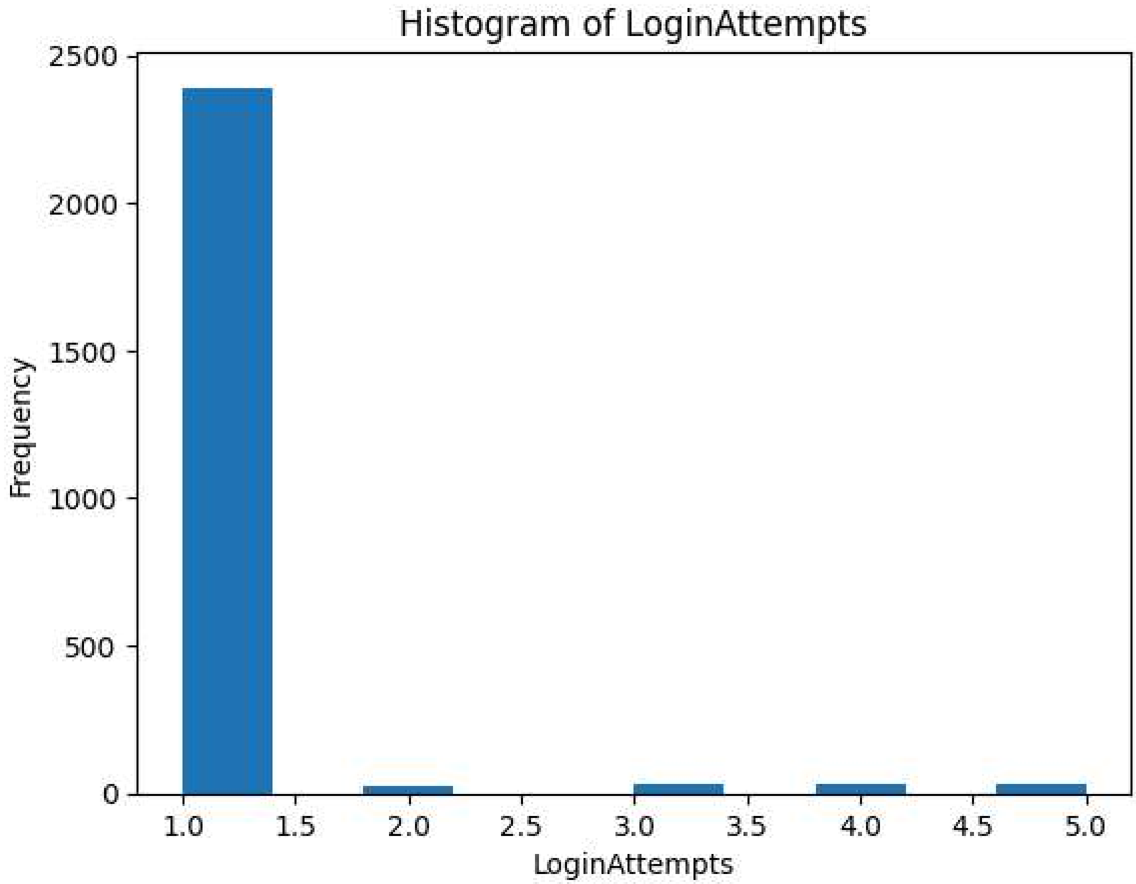
```
# 4. Plot distributions (numeric columns only)
numeric_cols = df.select_dtypes(include=np.number).columns

for col in numeric_cols:
    plt.figure()
    plt.hist(df[col].dropna())
    plt.title(f"Histogram of {col}")
    plt.xlabel(col)
    plt.ylabel("Frequency")
    plt.show()
```

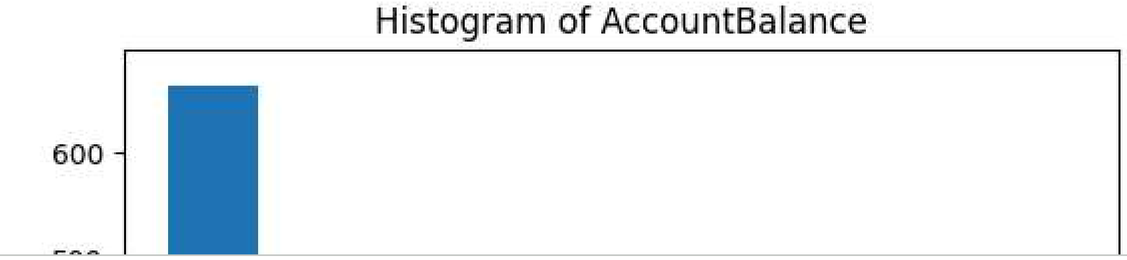

Frequency
1000
800
600
400
200
0
0
250
500
750
1000
1250
1500
1750
2000
TransactionAmount


Histogram of CustomerAge
Frequency
400
350
300
250
200
150
100
50
0
20
30
40
50
60
70
80
CustomerAge

Histogram of TransactionDuration
400


Frequency
350
300
250
200
150
100
50
0
0
50
100
150
200
250
300
TransactionDuration


Histogram of LoginAttempts
Frequency
2500
2000
1500
1000
500
0
1.0
1.5
2.0
2.5
3.0
3.5
4.0
4.5
5.0
LoginAttempts


Histogram of AccountBalance
600