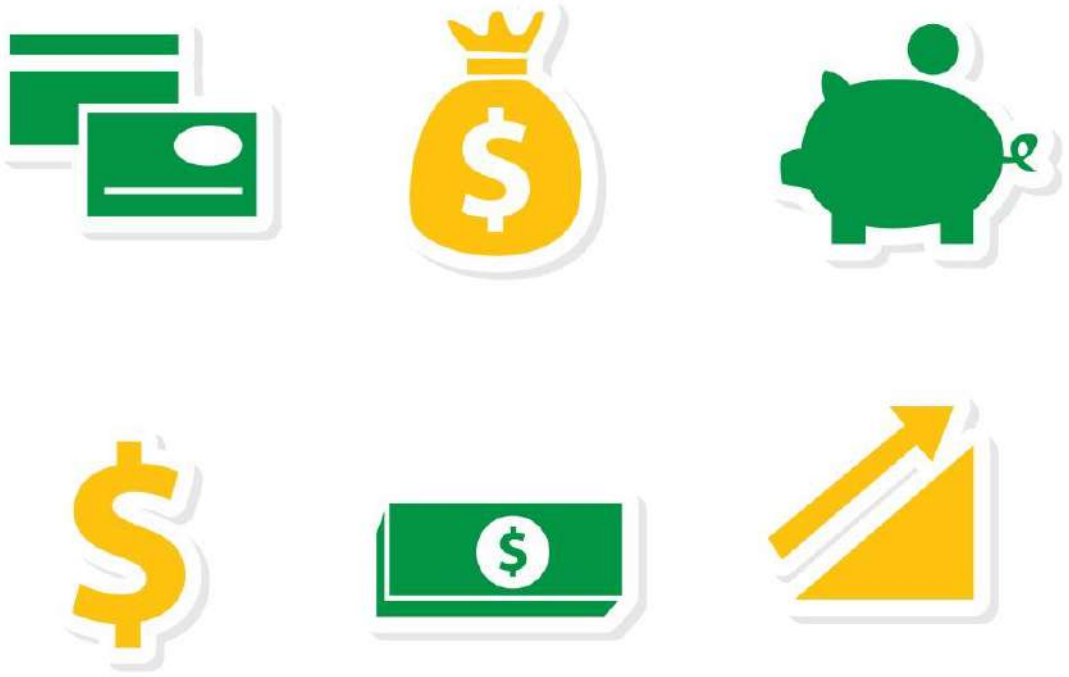
Financial Fraud Detection using python

By: Ankit Kumar Ray



About Dataset

Dataset Overview

Introduction

This dataset presents a synthetic representation of mobile money transactions, meticulously crafted to mirror the complexities of real-world financial activities while integrating fraudulent behaviors for research purposes. Derived from a simulator named PaySim, which utilizes aggregated data from actual financial logs of a mobile money service in an African country, this dataset aims to fill the gap in publicly available financial datasets for fraud detection studies. It encompasses a variety of transaction types including CASH-IN, CASH-OUT, DEBIT, PAYMENT, and TRANSFER over a simulated period of 30 days, providing a comprehensive environment for evaluating fraud detection methodologies. By addressing the intrinsic privacy concerns associated with financial transactions, this dataset offers a unique resource for researchers and analysts in the field of financial security and fraud detection, scaled to 1/4 of the original dataset size for efficient use within the Kaggle platform. Please note that transactions marked as fraudulent have been nullified, emphasizing the importance of non-balance columns for fraud analysis. This dataset is a contribution to the field from the "Scalable resource-efficient systems for big data analytics" project, funded by the Knowledge Foundation in Sweden.

Dataset Details

PaySim synthesizes mobile money transactions using data derived from a month's worth of financial logs from a mobile money service operating in an African country. These logs were provided by a multinational company that offers this financial service across more than 14 countries globally.

This synthetic dataset has been scaled to one-quarter the size of the original dataset and is specifically tailored for Kaggle.

Important Note: Transactions identified as fraudulent are annulled. Hence, for fraud detection analysis, the following columns should not be utilized: oldbalanceOrg, newbalanceOrig, oldbalanceDest, newbalanceDest.

Dataset Structure

- step: Represents a unit of time in the real world, with 1 step equating to 1 hour. The total simulation spans 744 steps, equivalent to 30 days.
- type: Transaction types include CASH-IN, CASH-OUT, DEBIT, PAYMENT, and TRANSFER.
- amount: The transaction amount in the local currency.

- nameOrig: The customer initiating the transaction.
- oldbalanceOrg: The initial balance before the transaction.
- newbalanceOrig: The new balance after the transaction.
- nameDest: The transaction's recipient customer.
- oldbalanceDest: The initial recipient's balance before the transaction. Not applicable for customers identified by 'M' (Merchants).
- newbalanceDest: The new recipient's balance after the transaction. Not applicable for 'M' (Merchants).
- isFraud: Identifies transactions conducted by fraudulent agents aiming to deplete customer accounts through transfers and cash-outs.
- isFlaggedFraud: Flags large-scale, unauthorized transfers between accounts, with any single transaction exceeding 200,000 being considered illegal.

Previous Research and Acknowledgments

This dataset has been generated through multiple runs of the PaySim simulator, each simulating a month of real-time transactions over 744 steps. Each run produced approximately 24 million financial records across the five transaction categories.

This project is part of the "Scalable resource-efficient systems for big data analytics" research, supported by the Knowledge Foundation (grant: 20140032) in Sweden.

For citations and further references, please use:

E. A. Lopez-Rojas, A. Elmir, and S. Axelsson. "PaySim: A financial mobile money simulator for fraud detection". In: The 28th European Modeling and Simulation Symposium-EMSS, Larnaca, Cyprus. 2016

DATASET FOUNDED ON =KAGGLE : here is the link

Financial Fraud Detection.py

File Edit View Insert Runtime Tools Help All changes saved

Share

+ Code + Text

RAM Disk

Gemini

```
[ ] Start coding or generate with AI.
```

#Financial Fraud Detection Analysis

Import Libraries

```
[6] import numpy as np
    import pandas as pd
    import matplotlib.pyplot as plt

    df = pd.read_csv('/content/Synthetic_Financial_datasets_log.csv')
    df
```

	step	type	amount	nameOrig	oldbalanceOrg	newbalanceOrig	nameDest	oldbalanceDest	newbalanceDest	isFraud	isFlaggedFraud
0	1	PAYMENT	9839.64	C1231006815	170136.00	160296.36	M1979787155	0.00	0.00	0.0	0.0

Connected to Python 3 Google Compute Engine backend

Financial Fraud Detection.py

File Edit View Insert Runtime Tools Help All changes saved

Share

+ Code + Text

RAM Disk

Gemini

[6] ...

2075455	182	PAYMENT	2588.57	C675879550	0.00	0.00	M265200354	0.00	0.00	0.0	0.0
2075456	182	PAYMENT	4121.50	C538403583	0.00	0.00	M1945443399	0.00	0.00	0.0	0.0
2075457	182	CASH_OUT	78183.77	C319775405	176017.00	97833.23	C1914772936	616233.31	694417.08	0.0	0.0
2075458	182	CASH_IN	196872.80	C1764494236	97833.23	294706.03	C1743891259	216020.94	19148.14	0.0	0.0
2075459	182	CASH_IN	74504.71	C1924631772	294706.03	369210.74	C739910885	770440.06	69.00	NaN	NaN

2075460 rows x 11 columns

```
[19] # shape of Dataset
     df.shape
```

```
(2075460, 11)
```

```
[ ] Start coding or generate with AI.
```

Preview of Dataset

Connected to Python 3 Google Compute Engine backend

Financial Fraud Detection.py

File Edit View Insert Runtime Tools Help All changes saved

Share

+ Code + Text

RAM Disk

Gemini

```
[7]
```

1	1	PAYMENT	1864.28	C1666544295	21249.0	19384.72	M2044282225	0.0	0.0	0.0	0.0
2	1	TRANSFER	181.00	C1305486145	181.0	0.00	C553264065	0.0	0.0	1.0	0.0
3	1	CASH_OUT	181.00	C840083671	181.0	0.00	C38997010	21182.0	0.0	1.0	0.0
4	1	PAYMENT	11668.14	C2048537720	41554.0	29885.86	M1230701703	0.0	0.0	0.0	0.0

```
[8] df.tail()
```

	step	type	amount	nameOrig	oldbalanceOrg	newbalanceOrig	nameDest	oldbalanceDest	newbalanceDest	isFraud	isFlaggedFraud
2075455	182	PAYMENT	2588.57	C675879550	0.00	0.00	M265200354	0.00	0.00	0.0	0.0
2075456	182	PAYMENT	4121.50	C538403583	0.00	0.00	M1945443399	0.00	0.00	0.0	0.0
2075457	182	CASH_OUT	78183.77	C319775405	176017.00	97833.23	C1914772936	616233.31	694417.08	0.0	0.0
2075458	182	CASH_IN	196872.80	C1764494236	97833.23	294706.03	C1743891259	216020.94	19148.14	0.0	0.0
2075459	182	CASH_IN	74504.71	C1924631772	294706.03	369210.74	C739910885	770440.06	69.00	NaN	NaN

Understand the Dataset

Connected to Python 3 Google Compute Engine backend

Financial Fraud Detection.py

File Edit View Insert Runtime Tools Help All changes saved

Share

+ Code + Text

RAM Disk

Gemini

```
[9] # Columns in Datasets
```

```
df.columns
```

```
Index(['step', 'type', 'amount', 'nameOrig', 'oldbalanceOrg', 'newbalanceOrig',
       'nameDest', 'oldbalanceDest', 'newbalanceDest', 'isFraud',
       'isFlaggedFraud'],
      dtype='object')
```

[11]
```
#Column names and data type
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2075460 entries, 0 to 2075459
Data columns (total 11 columns):
 #   Column          Dtype
---  ------          -----
 0   step            int64
 1   type            object
 2   amount          float64
 3   nameOrig        object
 4   oldbalanceOrg   float64
 5   newbalanceOrig  float64
 6   nameDest        object
 7   oldbalanceDest  float64
 8   newbalanceDest  float64
```

Connected to Python 3 Google Compute Engine backend

Financial Fraud Detection.py

File Edit View Insert Runtime Tools Help All changes saved

Share

+ Code + Text

RAM Disk

Gemini

[27]
```
df.describe()
df.describe(include='all')
```

	step	type	amount	nameOrig	oldbalanceOrg	newbalanceOrig	nameDest	oldbalanceDest	newbalanceDest	isFraud	isFlaggedFraud
count	2.075460e+06	2075460	2.075460e+06	2075460	2.075460e+06	2.075460e+06	2075460	2.075460e+06	2.075460e+06	2.075459e+06	2075459.0
unique	NaN	5	NaN	2074469	NaN	NaN	881527	NaN	NaN	NaN	NaN
top	NaN	CASH_OUT	NaN	C1999539787	NaN	NaN	C985934102	NaN	NaN	NaN	NaN
freq	NaN	739803	NaN	3	NaN	NaN	102	NaN	NaN	NaN	NaN
mean	8.832062e+01	NaN	1.617586e+05	NaN	8.543965e+05	8.757275e+05	NaN	9.972815e+05	1.115323e+06	9.920697e-04	0.0
std	6.416732e+01	NaN	2.715589e+05	NaN	2.923186e+06	2.959411e+06	NaN	2.305203e+06	2.399234e+06	3.148152e-02	0.0
min	1.000000e+00	NaN	6.000000e-02	NaN	0.000000e+00	0.000000e+00	NaN	0.000000e+00	0.000000e+00	0.000000e+00	0.0
25%	2.000000e+01	NaN	1.312667e+04	NaN	0.000000e+00	0.000000e+00	NaN	0.000000e+00	0.000000e+00	0.000000e+00	0.0
50%	9.400000e+01	NaN	7.845177e+04	NaN	1.491700e+04	0.000000e+00	NaN	1.380952e+05	2.287034e+05	0.000000e+00	0.0
75%	1.540000e+02	NaN	2.169498e+05	NaN	1.196988e+05	1.612174e+05	NaN	9.490263e+05	1.152259e+06	0.000000e+00	0.0
max	1.820000e+02	NaN	1.000000e+07	NaN	3.893942e+07	3.894623e+07	NaN	4.220740e+07	4.228378e+07	1.000000e+00	0.0

Connected to Python 3 Google Compute Engine backend

Financial Fraud Detection.py

File Edit View Insert Runtime Tools Help All changes saved

Share

+ Code + Text

RAM Disk

Gemini

[12]

	0
step	0
type	0
amount	0
nameOrig	0
oldbalanceOrg	0
newbalanceOrig	0
nameDest	0
oldbalanceDest	0
newbalanceDest	0
isFraud	1
isFlaggedFraud	1

dtype: int64

Connected to Python 3 Google Compute Engine backend

Financial Fraud Detection.py

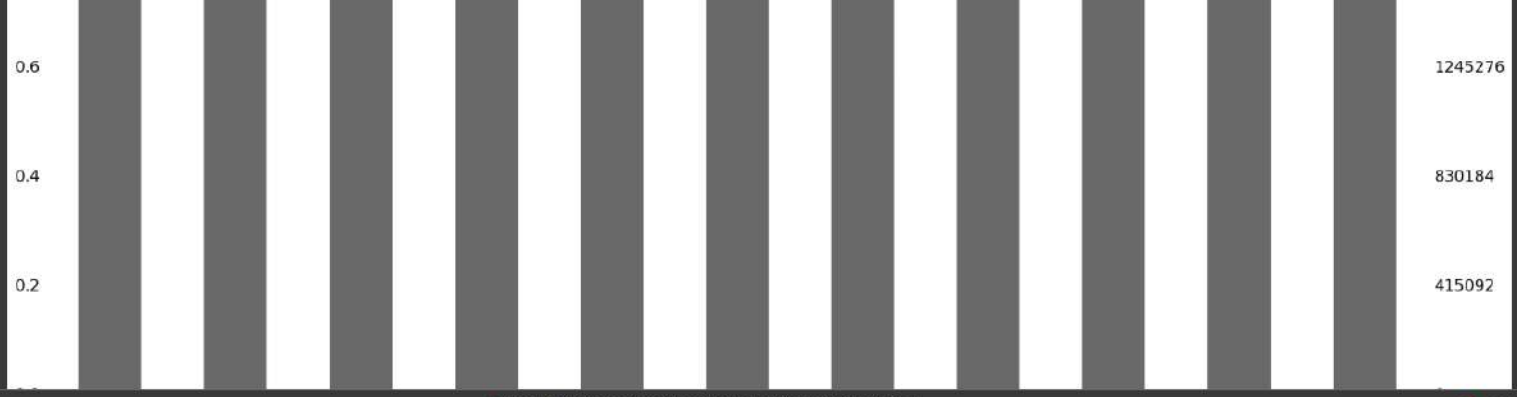
File Edit View Insert Runtime Tools Help All changes saved

Share

+ Code + Text

RAM Disk

Gemini



Connected to Python 3 Google Compute Engine backend

Financial Fraud Detection.py

File Edit View Insert Runtime Tools Help All changes saved

Share

+ Code + Text

RAM Disk

Gemini

Check for Duplicates

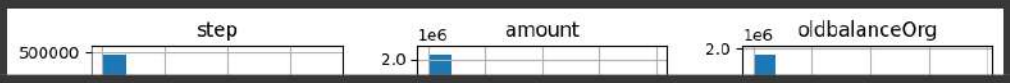
```
[18] #duplicate rows
     df.duplicated().sum()
```

```
0
```

Understand Data Distribution

```
[21] # Exploring numerical columns

     df.hist(figsize=(10, 8))
     plt.show()
```



Connected to Python 3 Google Compute Engine backend

Financial Fraud Detection.py

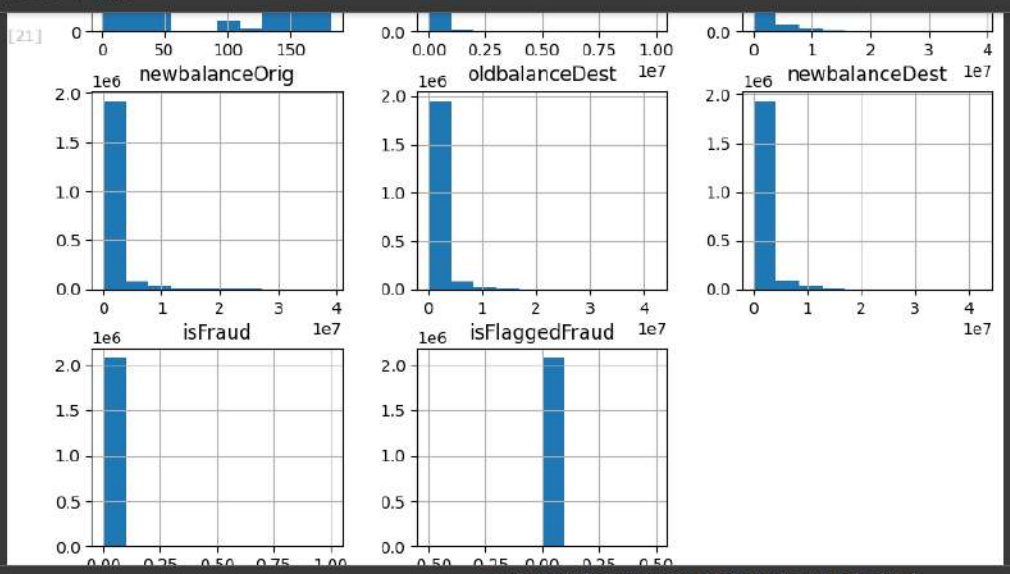
File Edit View Insert Runtime Tools Help All changes saved

Share

+ Code + Text

RAM Disk

Gemini



Connected to Python 3 Google Compute Engine backend

Financial Fraud Detection.py

File Edit View Insert Runtime Tools Help All changes saved

Share

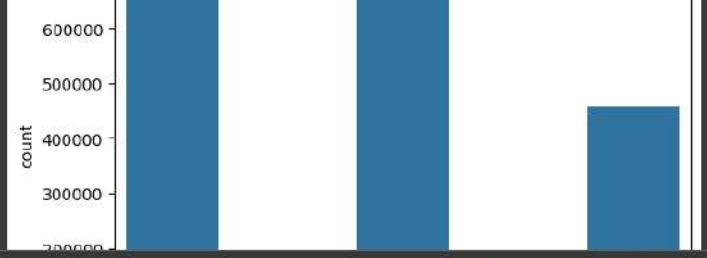
+ Code + Text

RAM Disk

Gemini

```
[32] z
```

```
type
CASH_OUT    739803
PAYMENT     692879
CASH_IN     456973
TRANSFER    172057
DEBIT        13748
Name: count, dtype: int64
<Axes: xlabel='type', ylabel='count'>
```



Connected to Python 3 Google Compute Engine backend

Financial Fraud Detection.py

File Edit View Insert Runtime Tools Help All changes saved

Share

+ Code + Text

RAM Disk

Gemini

Check for Outliers

```
sns.boxplot(data=df, x='amount')
```

```
<Axes: xlabel='amount'>
```

Connected to Python 3 Google Compute Engine backend

Financial Fraud Detection.py

File Edit View Insert Runtime Tools Help All changes saved

Share

+ Code + Text

RAM Disk

Gemini

```
[38] #rename column
     df.rename(columns={'type': 'payment_type'}, inplace=True)
     df
```

	step	payment_type	amount	nameOrig	oldbalanceOrg	newbalanceOrig	nameDest	oldbalanceDest	newbalanceDest	isFraud	isFlaggedFraud
0	1	PAYMENT	9839.64	C1231006815	170136.00	160296.36	M1979787155	0.00	0.00	0.0	0.0
1	1	PAYMENT	1864.28	C1666544295	21249.00	19384.72	M2044282225	0.00	0.00	0.0	0.0
2	1	TRANSFER	181.00	C1305486145	181.00	0.00	C553264065	0.00	0.00	1.0	0.0
3	1	CASH_OUT	181.00	C840083671	181.00	0.00	C38997010	21182.00	0.00	1.0	0.0
4	1	PAYMENT	11668.14	C2048537720	41554.00	29885.86	M1230701703	0.00	0.00	0.0	0.0
...	...	...	...	...	...	...	...	...	...	...	...
2075455	182	PAYMENT	2588.57	C675879550	0.00	0.00	M265200354	0.00	0.00	0.0	0.0
2075456	182	PAYMENT	4121.50	C538403583	0.00	0.00	M1945443399	0.00	0.00	0.0	0.0
2075457	182	CASH_OUT	78183.77	C319775405	176017.00	97833.23	C1914772936	616233.31	694417.08	0.0	0.0

Connected to Python 3 Google Compute Engine backend

Financial Fraud Detection.py

File Edit View Insert Runtime Tools Help All changes saved

Share

+ Code + Text

RAM Disk

Gemini

```
[39] df.columns
```

```
Index(['step', 'payment_type', 'amount', 'nameOrig', 'oldbalanceOrg',
       'newbalanceOrig', 'nameDest', 'oldbalanceDest', 'newbalanceDest',
       'isFraud', 'isFlaggedFraud'],
      dtype='object')
```

Handling missing values

```
[40] df.isnull().sum()
```

	0
step	0
payment_type	0
amount	0

nameOrig	0
oldbalanceOrg	0
newbalanceOrig	0

Connected to Python 3 Google Compute Engine backend

Financial Fraud Detection.py

File Edit View Insert Runtime Tools Help All changes saved

Share

+ Code + Text

RAM Disk

Gemini

[40]

isFlaggedFraud	1

dtype: int64

[41]
```
df['isFraud'].value_counts()
```

isFraud	count
0.0	2073400
1.0	2059

dtype: int64

[42]
```
df['isFraud'].isnull()
```

	isFraud
0	False
1	False

Connected to Python 3 Google Compute Engine backend

Financial Fraud Detection.py

File Edit View Insert Runtime Tools Help All changes saved

Share

+ Code + Text

RAM Disk

Gemini

[42]

2075456	False
2075457	False
2075458	False
2075459	True

2075460 rows x 1 columns

dtype: bool

[43]
```
df['isFraud'].isnull().sum()
```

```
1
```

[49]
```
df['isFraud'].fillna(df['isFraud'].mean(), inplace=True)
df.isnull().sum()
```

```
<ipython-input-49-0183622795da>:1: FutureWarning: A value is trying to be set on a copy of a DataFrame or Series through chained assignment using an inplace method.
The behavior will change in pandas 3.0. This inplace method will never work because the intermediate object on which we are setting values always behaves as a copy.

For example, when doing 'df[col].method(value, inplace=True)', try using 'df.method({col: value}, inplace=True)' or df[col] = df[col].method(value) instead, to perform
```

Connected to Python 3 Google Compute Engine backend

Financial Fraud Detection.py

File Edit View Insert Runtime Tools Help All changes saved

Share

+ Code + Text

RAM Disk

Gemini

[49]

amount	0
nameOrig	0
oldbalanceOrg	0
newbalanceOrig	0
nameDest	0
oldbalanceDest	0
newbalanceDest	0
isFraud	0
isFlaggedFraud	1

dtype: int64

[50]
```
df['isFlaggedFraud'].isnull().sum()
```

```
1
```

[51]
```
df['isFlaggedFraud'].fillna(df['isFlaggedFraud'].mean(), inplace=True)
```

Connected to Python 3 Google Compute Engine backend

Financial Fraud Detection.py

File Edit View Insert Runtime Tools Help All changes saved

Share

+ Code + Text

RAM Disk

Gemini

```
[51] df['isFlaggedFraud'].fillna(df['isFlaggedFraud'].mean(), inplace=True)
```

	0
step	0
payment_type	0
amount	0
nameOrig	0
oldbalanceOrg	0
newbalanceOrig	0
nameDest	0
oldbalanceDest	0
newbalanceDest	0
isFraud	0
isFlaggedFraud	0

dtype: int64

Connected to Python 3 Google Compute Engine backend

Financial Fraud Detection.py

File Edit View Insert Runtime Tools Help All changes saved

Share

+ Code + Text

RAM Disk

Gemini

[53]

1	1	PAYMENT	1864.28	C1666544295	21249.0	19384.72	M2044282225	0.0	0.0	0.0	0.0
2	1	TRANSFER	181.00	C1305486145	181.0	0.00	C553264065	0.0	0.0	1.0	0.0
3	1	CASH_OUT	181.00	C840083671	181.0	0.00	C38997010	21182.0	0.0	1.0	0.0
4	1	PAYMENT	11668.14	C2048537720	41554.0	29885.86	M1230701703	0.0	0.0	0.0	0.0

Data Transformation

[64] Start coding or generate with AI.

```
[63] df.head(2)
```

	step	payment_type	amount	nameOrig	oldbalanceOrg	newbalanceOrig	nameDest	oldbalanceDest	newbalanceDest	isFraud	isFlaggedFraud	amount_category
0	1	PAYMENT	9839.64	C1231006815	170136.0	160296.36	M1979787155	0.0	0.0	0.0	0.0	medium
1	1	PAYMENT	1864.28	C1666544295	21249.0	19384.72	M2044282225	0.0	0.0	0.0	0.0	low

[61] # Feature Engineering - Adding a New Column

Connected to Python 3 Google Compute Engine backend

Financial Fraud Detection.py

File Edit View Insert Runtime Tools Help All changes saved

Share

+ Code + Text

RAM Disk

Gemini

```
[61] df.head()
```

	step	payment_type	amount	nameOrig	oldbalanceOrg	newbalanceOrig	nameDest	oldbalanceDest	newbalanceDest	isFraud	isFlaggedFraud	amount_category
0	1	PAYMENT	9839.64	C1231006815	170136.0	160296.36	M1979787155	0.0	0.0	0.0	0.0	medium
1	1	PAYMENT	1864.28	C1666544295	21249.0	19384.72	M2044282225	0.0	0.0	0.0	0.0	low
2	1	TRANSFER	181.00	C1305486145	181.0	0.00	C553264065	0.0	0.0	1.0	0.0	low
3	1	CASH_OUT	181.00	C840083671	181.0	0.00	C38997010	21182.0	0.0	1.0	0.0	low
4	1	PAYMENT	11668.14	C2048537720	41554.0	29885.86	M1230701703	0.0	0.0	0.0	0.0	high

[70] Start coding or generate with AI.

```
[69] # data aggregation
transaction_summary = df.groupby('payment_type').agg(
    total_amount=('amount', 'sum'),
    num_transactions=('amount', 'count'),
    avg_amount=('amount', 'mean'),
    max_amount=('amount', 'max')
).reset_index()
```

Connected to Python 3 Google Compute Engine backend

Financial Fraud Detection.py

File Edit View Insert Runtime Tools Help All changes saved

Share

+ Code + Text

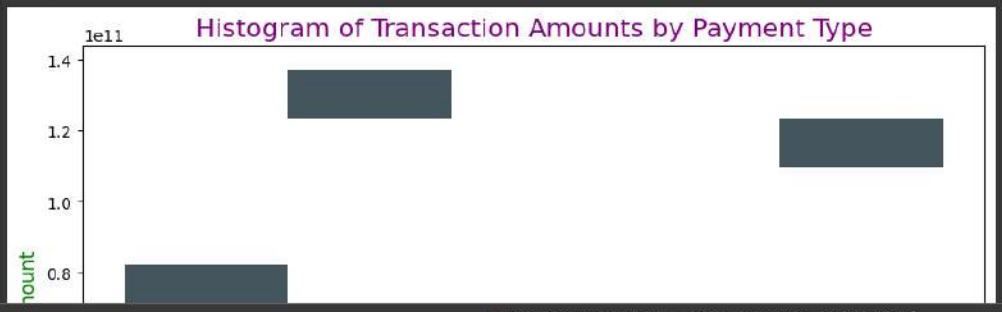
RAM Disk

Gemini

3	PAYMENT	8.269850e+09	692879	11935.489923	146931.72

```
4    TRANSFER  1.122174e+11          172057  652210.724549  10000000.00
```

```
plt.figure(figsize=(10, 6))
sns.histplot(x='payment_type',y='total_amount',data=transaction_summary,color='skyblue', bins=10)
plt.title('Histogram of Transaction Amounts by Payment Type', fontsize=16, color='purple')
plt.xlabel('Payment Type', fontsize=12, color='green')
plt.ylabel('Total Amount', fontsize=12, color='green')
plt.show()
```



Connected to Python 3 Google Compute Engine backend

Financial Fraud Detection.py

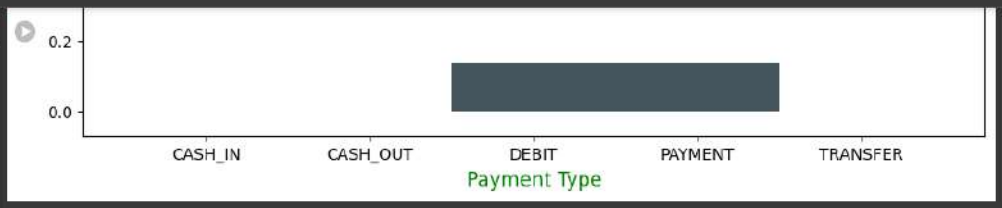
File Edit View Insert Runtime Tools Help All changes saved

Share

+ Code + Text

RAM Disk

Gemini




```
[88] plt.figure(figsize=(10, 6))
     sns.barplot(x='payment_type', y='total_amount', data=transaction_summary, palette='coolwarm')

     plt.title('Transaction Amounts by Payment Type', fontsize=16, color='darkblue')
     plt.xlabel('Payment Type', fontsize=12, color='green')
     plt.ylabel('Total Amount', fontsize=12, color='green')
     plt.show()
```

```
<ipython-input-88-fd885885432f>:2: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `x` variable to `hue` and set `legend=False` for the same effect.

  sns.barplot(x='payment_type', y='total_amount', data=transaction_summary, palette='coolwarm')
```

Connected to Python 3 Google Compute Engine backend

Financial Fraud Detection.py

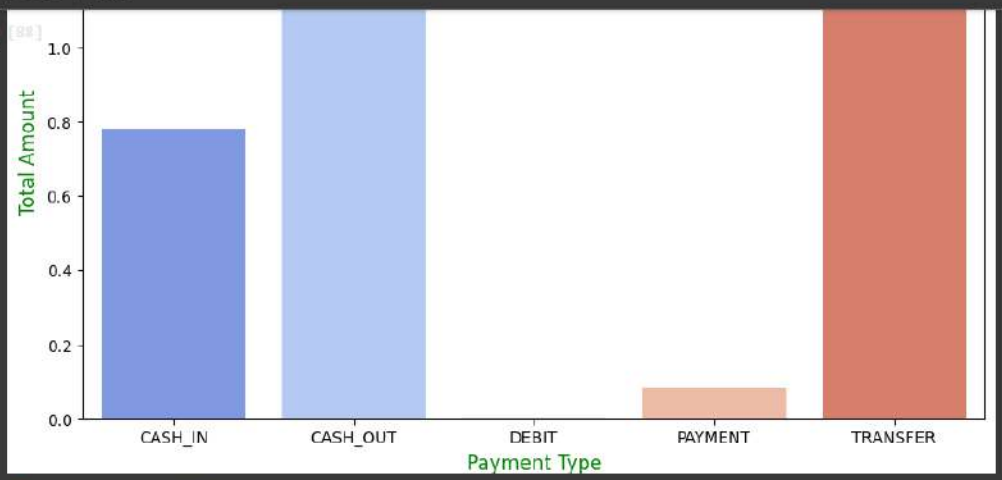
File Edit View Insert Runtime Tools Help All changes saved

Share

+ Code + Text

RAM Disk

Gemini



Financial Fraud Detection.py

File Edit View Insert Runtime Tools Help All changes saved

Share

+ Code + Text

RAM Disk

Gemini