Exploratory Data Analysis (EDA) on the Dataset

Submitted by:

1. **Dileep Kumar (ERP: 18255)**
2. **Muhammad Arsalan Mubeen (ERP: 23394)**

Source Data (PaySim Data(Analysis))

PaySim simulates mobile money transactions based on a sample of real transactions extracted from one month of financial logs from a mobile money service implemented in an African country. The original logs were provided by a multinational company, who is the provider of the mobile financial service which is currently running in more than 14 countries all around the world.

Dictionary

This is the column definition of the referenced sythentic dataset.

Column Name	Description
step	maps a unit of time in the real world. In this case 1 step is 1 hour of time. Total steps 744 (30 days simulation).
type	CASH-IN, CASH-OUT, DEBIT, PAYMENT and TRANSFER.
amount	amount of the transaction in local currency.
nameOrig	customer who started the transaction
oldbalanceOrg	initial balance before the transaction
newbalanceOrig	new balance after the transaction
nameDest	customer who is the recipient of the transaction
oldbalanceDest	initial balance recipient before the transaction. Note that there is not information for customers that start with M (Merchants).
newbalanceDest	new balance recipient after the transaction. Note that there is not information for customers that start with M (Merchants).

```
[9] import numpy as np # linear algebra
    import pandas as pd # data processing, CSV file I/O (e.g. pd.read_csv)
    import seaborn as sns
    from matplotlib import pyplot as plt
```

```
[2] df= pd.read_csv('/content/sampledata.csv')
    df.head()
```

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

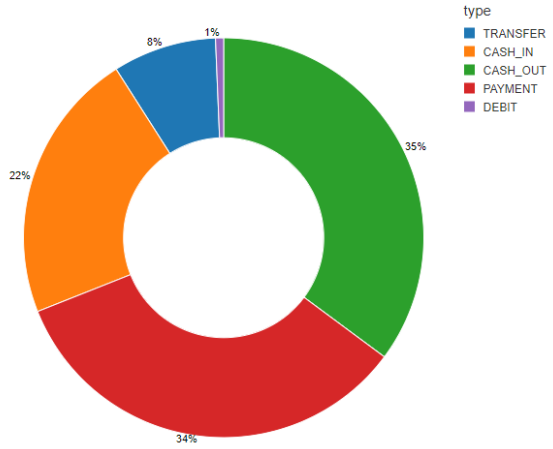
```
[3] df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 101 entries, 0 to 100
Data columns (total 11 columns):
 #   Column          Non-Null Count  Dtype
---  ------          --------------  -----
 0   step            101 non-null    int64
 1   type            101 non-null    object
 2   amount          101 non-null    float64
 3   nameOrig        101 non-null    object
 4   oldbalanceOrg   101 non-null    float64
 5   newbalanceOrig  101 non-null    float64
 6   nameDest        101 non-null    object
 7   oldbalanceDest  101 non-null    float64
 8   newbalanceDest  101 non-null    float64
 9   isFraud         101 non-null    int64
 10  isFlaggedFraud  101 non-null    int64
dtypes: float64(5), int64(3), object(3)
memory usage: 8.8+ KB
```

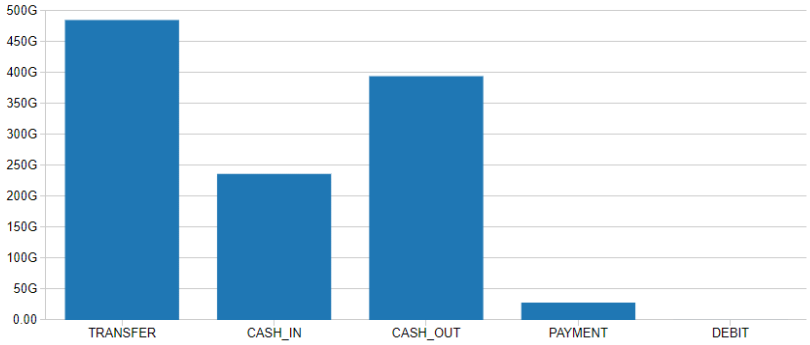
```
[4] df.describe()
```

	step	amount	oldbalanceOrg	newbalanceOrig	oldbalanceDest	newbalanceDest	isFraud	isFlaggedFraud
count	101.0	1.010000e+02	101.000000	101.000000	1.010000e+02	1.010000e+02	101.000000	101.0
mean	1.0	1.153051e+05	55599.275842	49472.277723	1.943734e+05	1.200560e+06	0.019802	0.0
std	0.0	2.970308e+05	142805.702309	141370.969528	6.687200e+05	4.242719e+06	0.140014	0.0
min	1.0	3.866000e+01	0.000000	0.000000	0.000000e+00	0.000000e+00	0.000000	0.0
25%	1.0	3.448920e+03	0.000000	0.000000	0.000000e+00	0.000000e+00	0.000000	0.0
50%	1.0	7.413540e+03	8547.000000	0.000000	0.000000e+00	0.000000e+00	0.000000	0.0
75%	1.0	5.695390e+04	26845.410000	19998.140000	5.275200e+04	1.651836e+04	0.000000	0.0
max	1.0	1.724887e+06	882770.000000	874042.260000	5.195482e+06	1.916920e+07	1.000000	0.0

What are the type of transactions?



How much money are we talking about (synthetically)?

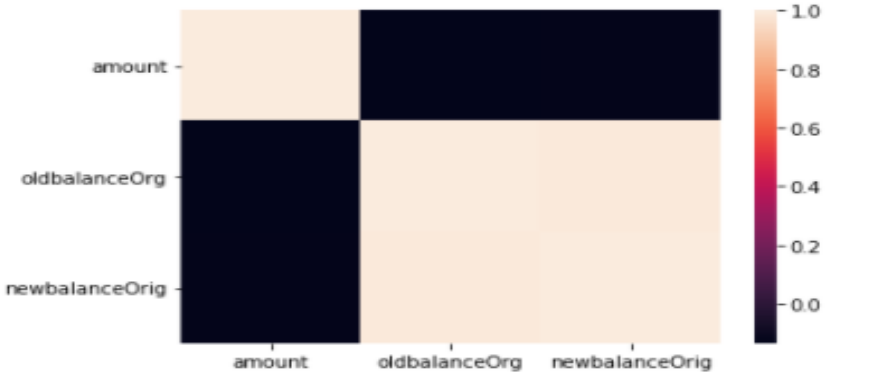



```
[6]  numerical_colss=['amount','oldbalanceOrg','newbalanceOrig']
     df[numerical_colss].corr()
```

	amount	oldbalanceOrg	newbalanceOrig
amount	1.000000	-0.134047	-0.128058
oldbalanceOrg	-0.134047	1.000000	0.993814
newbalanceOrig	-0.128058	0.993814	1.000000

```
[7]  sns.heatmap(df[numerical_colss].corr())
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7fcd1c9d3d68>
```



Plot pairwise relationships in a dataset.

```
sns.pairplot(data=df[df.columns[1:]],diag_kws={'edgecolor':'k','bins':25},plot_kws={'edgecolor':'k'})
plt.show()
```