Team RUNTIME TERROR
DATA IMPORT AND VISUALISATION

Libraries Imported:

In [1]:
```
import numpy as np
import pandas as pd
from pandas.plotting import scatter_matrix
import matplotlib.pyplot as plt
import matplotlib.lines as mlines
import seaborn as sns
from sklearn.model_selection import train_test_split, learning_curve
from sklearn.metrics import average_precision_score
from sklearn.linear_model import LogisticRegression
from sklearn.tree import DecisionTreeClassifier
from sklearn.neighbors import KNeighborsClassifier
from sklearn.discriminant_analysis import LinearDiscriminantAnalysis
from sklearn.naive_bayes import GaussianNB
from sklearn.svm import SVC
```

In [2]:
```
dataset = pd.read_csv('C:/Users/yashb/Downloads/PS_20174392719_1491204439457_log.csv')
print(dataset.shape)
```

```
(6362620, 11)
```

Describing the Database:

In [3]:
```
print(dataset.describe())
```

```
               step        amount  oldbalanceOrg  newbalanceOrig  \
count  6.362620e+06  6.362620e+06   6.362620e+06    6.362620e+06
mean   2.433972e+02  1.798619e+05   8.338831e+05    8.551137e+05
std    1.423320e+02  6.038582e+05   2.888243e+06    2.924049e+06
min    1.000000e+00  0.000000e+00   0.000000e+00    0.000000e+00
25%    1.560000e+02  1.338957e+04   0.000000e+00    0.000000e+00
50%    2.390000e+02  7.487194e+04   1.420800e+04    0.000000e+00
75%    3.350000e+02  2.087215e+05   1.073152e+05    1.442584e+05
max    7.430000e+02  9.244552e+07   5.958504e+07    4.958504e+07

       oldbalanceDest  newbalanceDest       isFraud  isFlaggedFraud
count    6.362620e+06    6.362620e+06  6.362620e+06    6.362620e+06
mean     1.100702e+06    1.224996e+06  1.290820e-03    2.514687e-06
std      3.399180e+06    3.674129e+06  3.590480e-02    1.585775e-03
min      0.000000e+00    0.000000e+00  0.000000e+00    0.000000e+00
25%      0.000000e+00    0.000000e+00  0.000000e+00    0.000000e+00
50%      1.327057e+05    2.146614e+05  0.000000e+00    0.000000e+00
75%      9.430367e+05    1.111909e+06  0.000000e+00    0.000000e+00
max      3.560159e+08    3.561793e+08  1.000000e+00    1.000000e+00
```

Data Cleaning and Pruning :

Checking whether the dataset has any NULL values:

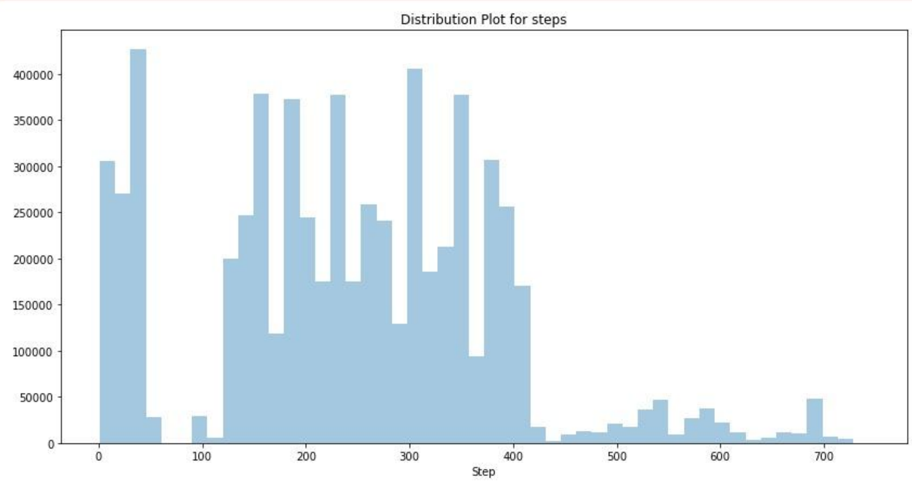
```
In [5]: dataset.isnull().values.any()

Out[5]: False
```

Getting the count of unique steps:

```
In [9]: steps = dataset['step'].value_counts().nunique()
        print(steps)

        428
```

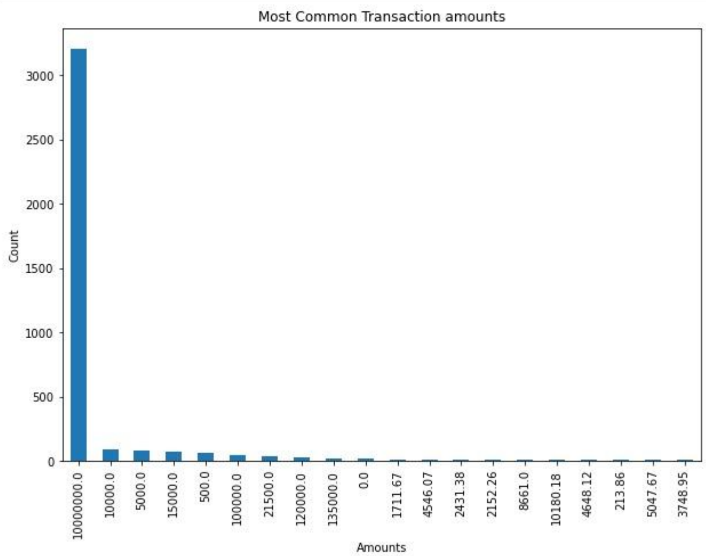
Bar Graph for steps distribution:

```
In [26]: plt.rcParams['figure.figsize'] =(14, 7)
         sns.distplot(dataset.step, kde = False)
         plt.title('Distribution Plot for steps')
         plt.xlabel('Step')
         plt.show();

C:\Users\yashb\anaconda3\lib\site-packages\seaborn\distributions.py:2551: FutureWarning: `distplot` is a deprecated function an
d will be removed in a future version. Please adapt your code to use either `displot` (a figure-level function with similar fle
xibility) or `histplot` (an axes-level function for histograms).
  warnings.warn(msg, FutureWarning)
```



Common Transaction Amounts:

In [29]:
```
plt.rcParams['figure.figsize'] = (10, 7)
dataset['amount'].value_counts().head(20).plot.bar()
plt.title('Most Common Transaction amounts')
plt.xlabel('Amounts')
plt.ylabel('Count')
plt.show()
```



Frauds Detected by the system and money lost:

In [6]:
```
oc_isFraud = dataset['isFraud'].value_counts()
oc_isFraud
```

Out[6]:
```
0    6354407
1       8213
Name: isFraud, dtype: int64
```

In [7]:
```
oc_isFlaggedFraud= dataset['isFlaggedFraud'].value_counts()
oc_isFlaggedFraud
```

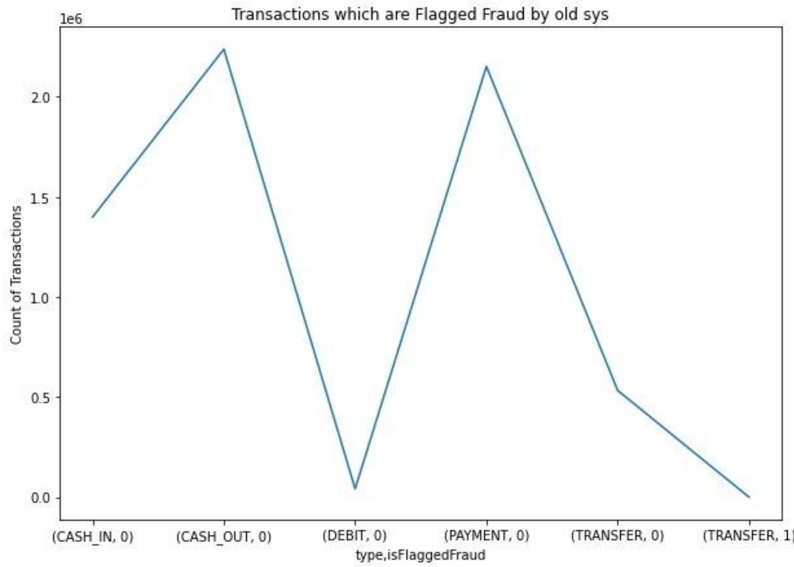
Out[7]:
```
0    6362604
1         16
Name: isFlaggedFraud, dtype: int64
```

In [8]:
```
fraud_type= dataset.groupby("type")["isFraud"].count()
fraud_type
```

Out[8]:
```
type
CASH_IN     1399284
CASH_OUT    2237500
DEBIT         41432
PAYMENT     2151495
TRANSFER     532909
Name: isFraud, dtype: int64
```

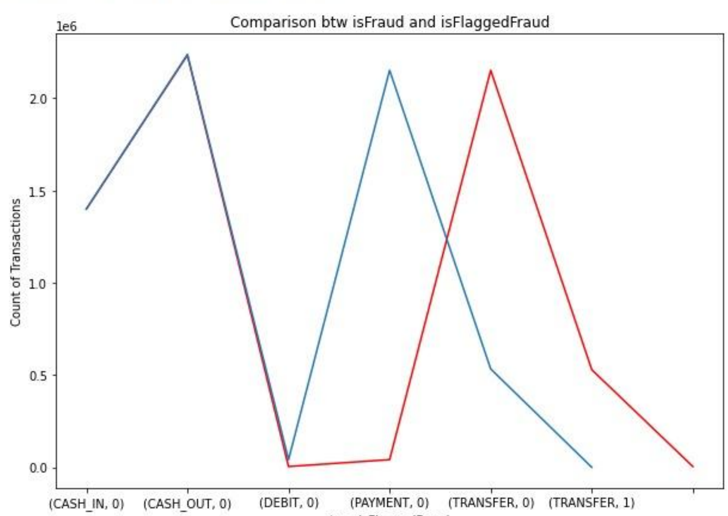
Transactions flagged fraud by the old system:

```
In [12]: #isFlaggedFraud (conf)
         ax = dataset.groupby(['type', 'isFlaggedFraud'])['amount'].size().plot(kind='line')
         ax.set_title("Transactions which are Flagged Fraud by old sys")
         ax.set_ylabel("Count of Transactions")

Out[12]: Text(0, 0.5, 'Count of Transactions')
```



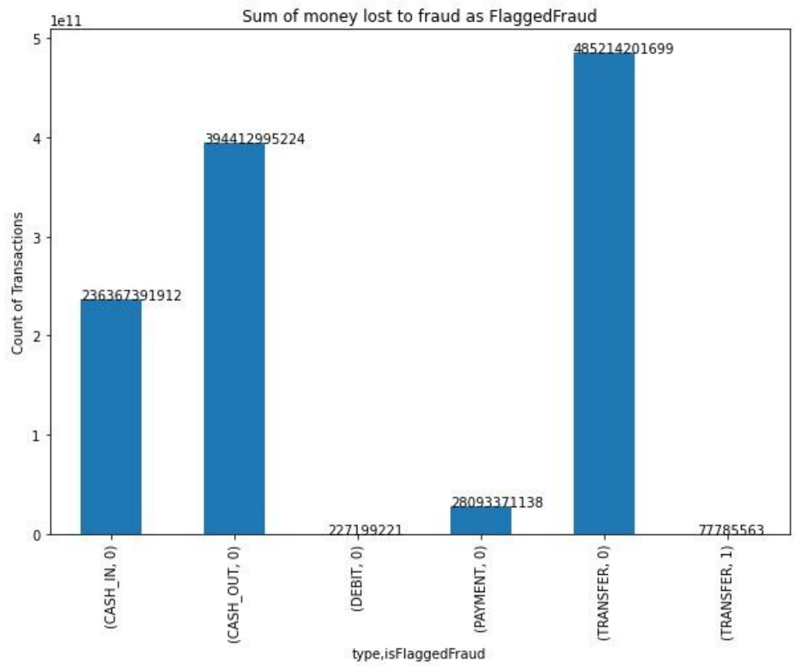
Comparison between isFraud and isFlaggedFraud:

```
In [13]: #comparison btw isFraud and isFlaggedFraud
         ax = dataset.groupby(['type', 'isFraud'])['amount'].size().plot(kind='line', color='r')
         ax = dataset.groupby(['type', 'isFlaggedFraud'])['amount'].size().plot(kind='line')
         ax.set_title("Comparison btw isFraud and isFlaggedFraud")
         ax.set_ylabel("Count of Transactions")

Out[13]: Text(0, 0.5, 'Count of Transactions')
```



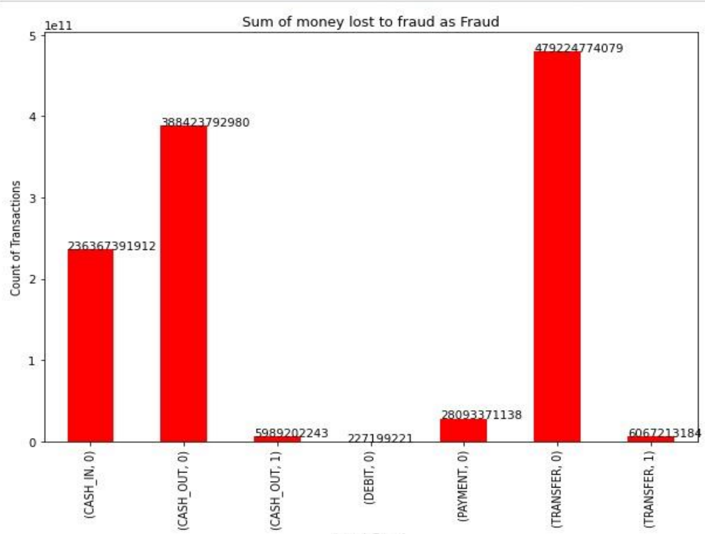
Sum of money lost to fraud as flagged fraud:

In [20]:
```
#Sum of money lost to fraud as FlaggedFraud
ax = dataset.groupby(['type', 'isFlaggedFraud'])['amount'].sum().plot(kind='bar')
ax.set_title("Sum of money lost to fraud as FlaggedFraud")
ax.set_ylabel("Count of Transactions")
for q in ax.patches:
    ax.annotate(str(format(int(q.get_height()))),(q.get_x(), q.get_height()))
```



Sum of money lost to fraud as Fraud:

In [22]:
```
#Sum of money lost to fraud as Fraud
ax = dataset.groupby(['type', 'isFraud'])['amount'].sum().plot(kind='bar', color='r')
ax.set_title("Sum of money lost to fraud as Fraud")
ax.set_ylabel("Count of Transactions")
for q in ax.patches:
    ax.annotate(str(format(int(q.get_height()))),(q.get_x(), q.get_height()))
```



Maximum and minimum transaction:

In [24]:
```
print("Minimum Transaction :", dataset.loc[dataset.isFlaggedFraud == 1].amount.min())
print("Maximum Transaction :", dataset.loc[dataset.isFlaggedFraud == 1].amount.max())
```

```
Minimum Transaction : 353874.22
Maximum Transaction : 10000000.0
```

In [30]:
```
dataTransfer = dataset.loc[dataset['type'] == 'TRANSFER']
dataTransfer = pd.DataFrame(dataTransfer)
dataTransfer.head(20)
```

Out[30]:

	step	type	amount	nameOrig	oldbalanceOrg	newbalanceOrig	nameDest	oldbalanceDest	newbalanceDest	isFraud	isFlaggedFraud
2	1	TRANSFER	181.00	C1305486145	181.00	0.0	C553264065	0.00	0.00	1	0
19	1	TRANSFER	215310.30	C1670993182	705.00	0.0	C1100439041	22425.00	0.00	0	0
24	1	TRANSFER	311685.89	C1984094095	10835.00	0.0	C932583850	6267.00	2719172.89	0	0
58	1	TRANSFER	62610.80	C1976401987	79114.00	16503.2	C1937962514	517.00	8383.29	0	0
78	1	TRANSFER	42712.39	C283039401	10363.39	0.0	C1330106945	57901.66	24044.18	0	0
79	1	TRANSFER	77957.68	C207471778	0.00	0.0	C1761291320	94900.00	22233.65	0	0
80	1	TRANSFER	17231.46	C1243171897	0.00	0.0	C783286238	24672.00	0.00	0	0
81	1	TRANSFER	78766.03	C1376151044	0.00	0.0	C1749186397	103772.00	277515.05	0	0
82	1	TRANSFER	224606.64	C873175411	0.00	0.0	C766572210	354678.92	0.00	0	0
83	1	TRANSFER	125872.53	C1443967876	0.00	0.0	C392292416	348512.00	3420103.09	0	0
84	1	TRANSFER	379856.23	C1449772539	0.00	0.0	C1590550415	900180.00	19169204.93	0	0
85	1	TRANSFER	1505626.01	C926859124	0.00	0.0	C665576141	29031.00	5515763.34	0	0
86	1	TRANSFER	554026.99	C1603696865	0.00	0.0	C766572210	579285.56	0.00	0	0
87	1	TRANSFER	147543.10	C12905860	0.00	0.0	C1359044626	223220.00	16518.36	0	0
88	1	TRANSFER	761507.39	C412788346	0.00	0.0	C1590550415	1280036.23	19169204.93	0	0
89	1	TRANSFER	1429051.47	C1520267010	0.00	0.0	C1590550415	2041543.62	19169204.93	0	0
90	1	TRANSFER	358831.92	C908084672	0.00	0.0	C392292416	474384.53	3420103.09	0	0
91	1	TRANSFER	367768.40	C288306765	0.00	0.0	C1359044626	370763.10	16518.36	0	0
92	1	TRANSFER	209711.11	C1556867940	0.00	0.0	C1509514333	399214.71	2415.16	0	0
93	1	TRANSFER	583848.46	C1839168128	0.00	0.0	C1286084959	667778.00	2107778.11	0	0