Online Fraud Detection

Major project

#import libraries and Packages:

```
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
```

#Loading Dataset:

```
df=pd.read_csv('/content/online fraud detection.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	0.0
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

```
df.columns
```

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

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 83561 entries, 0 to 83560
Data columns (total 11 columns):
 #   Column          Non-Null Count  Dtype
---  ------          --------------  -----
 0   step            83561 non-null  int64
 1   type            83561 non-null  object
 2   amount          83561 non-null  float64
 3   nameOrig        83561 non-null  object
 4   oldbalanceOrg   83560 non-null  float64
 5   newbalanceOrig  83560 non-null  float64
 6   nameDest        83560 non-null  object
 7   oldbalanceDest  83560 non-null  float64
 8   newbalanceDest  83560 non-null  float64
 9   isFraud         83560 non-null  float64
 10  isFlaggedFraud  83560 non-null  float64
dtypes: float64(7), int64(1), object(3)
memory usage: 7.0+ MB
```

```
df.isnull().sum()
```

```
step            0
type            0
amount          0
nameOrig        0
oldbalanceOrg   1
newbalanceOrig  1
nameDest        1
oldbalanceDest  1
newbalanceDest  1
isFraud         1
isFlaggedFraud  1
dtype: int64
```

```
df.shape
```

```
(83561, 11)
```

```
df['type'].unique()
```

```
array(['PAYMENT', 'TRANSFER', 'CASH_OUT', 'DEBIT', 'CASH_IN'],
      dtype=object)
```

```
type=df['type'].value_counts()
type
```

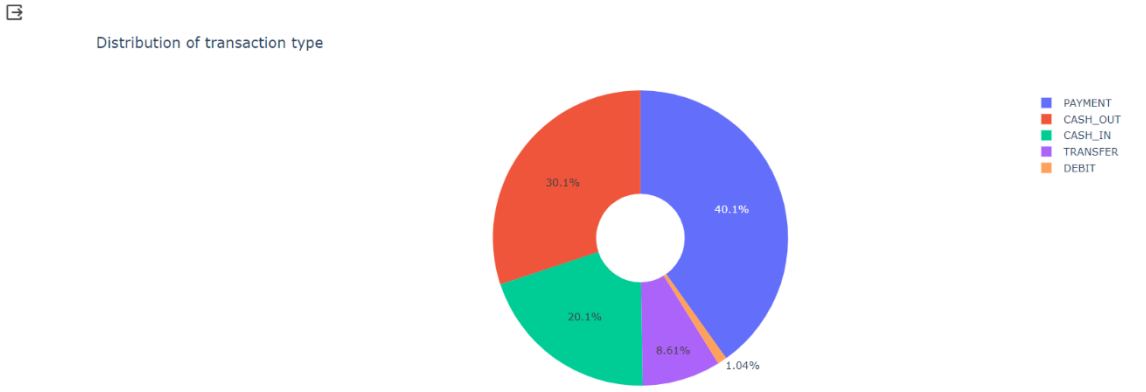
```
PAYMENT 33529
CASH_OUT 25156
CASH_IN 16818
TRANSFER 7192
DEBIT 866
Name: type, dtype: int64
```

```
transaction=type.index

quantity=type.values

import plotly.express as px
```

```
px.pie(df,values=quantity,names=transaction,hole=0.3,title="Distributio
n of transaction type")
```



```
df=df.dropna()


df.replace(to_replace=['PAYMENT', 'TRANSFER', 'CASH_OUT', 'DEBIT',
'CASH_IN'],value=[1,4,2,5,3],inplace=True)
```

```
<ipython-input-90-4443fa14fb80>:1: SettingWithCopyWarning:


A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-
docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
```

```
df['isFraud']=df['isFraud'].map({0:'N0 fraud',1:'fraud'})
df
```

```
<ipython-input-91-5055155ca92a>:1: SettingWithCopyWarning:


A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
```

	step	type	amount	nameOrig	oldbalanceOrg	newbalanceOrig	nameDest	oldbalanceDest	newbalanceDest	isFraud	isFlaggedFraud
0	1	1	9839.64	C1231006815	170136.0	160296.36	M1979787155	0.00	0.00	N0 fraud	0.0
1	1	1	1864.28	C1666544295	21249.0	19384.72	M2044282225	0.00	0.00	N0 fraud	0.0
2	1	4	181.00	C1305486145	181.0	0.00	C553264065	0.00	0.00	fraud	0.0
3	1	2	181.00	C840083671	181.0	0.00	C38997010	21182.00	0.00	fraud	0.0
4	1	1	11668.14	C2048537720	41554.0	29885.86	M1230701703	0.00	0.00	N0 fraud	0.0
...	...	...	...	...	...	...	...	...	...	...	...
83555	10	2	14895.17	C214279684	51759.0	36863.83	C1298314970	979963.09	994858.25	N0 fraud	0.0
83556	10	1	7705.70	C1834114901	96490.0	88784.30	M1214836727	0.00	0.00	N0 fraud	0.0
83557	10	2	319045.01	C1964329082	56471.0	0.00	C699133054	0.00	319045.01	N0 fraud	0.0
83558	10	3	249169.96	C1421944154	3481.0	252650.96	C790672270	38177.07	0.00	N0 fraud	0.0
83559	10	2	244279.64	C722886752	29968.0	0.00	C1492538502	25680.00	269959.64	N0 fraud	0.0

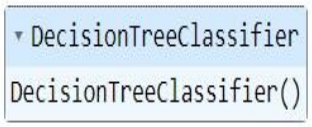
83560 rows × 11 columns

```
x=df[['type','amount','oldbalanceOrg','newbalanceOrig']]
y=df.iloc[:,-2]
y
```

```
0       N0 fraud
1       N0 fraud
2          fraud
3          fraud
4       N0 fraud
          ...
83555   N0 fraud
83556   N0 fraud
83557   N0 fraud
83558   N0 fraud
83559   N0 fraud
Name: isFraud, Length: 83560, dtype: object
```

```
from sklearn.tree import DecisionTreeClassifier
model=DecisionTreeClassifier()
```

```
xtrain,xtest,ytrain,ytest=train_test_split(x,y,test_size=0.2,random_sta
te=42)
model.fit(xtrain,ytrain)
```




```
model.score(xtest,ytest)
```

```
0.9985639061752034
```

```
model.predict([[3,9800,170136,160296]])
```

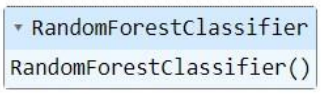
/usr/local/lib/python3.10/dist-packages/sklearn/base.py:439: UserWarning:

X does not have valid feature names, but DecisionTreeClassifier was fitted with feature names

array(['N0 fraud'], dtype=object)

```
from sklearn.ensemble import RandomForestClassifier
```

```
model=RandomForestClassifier()
xtrain,xtest,ytrain,ytest=train_test_split(x,y,test_size=0.2,random_sta
te=42)
model.fit(xtrain,ytrain)
```




```
model.score(xtest,ytest
```

```
0.9990426041168023
```

```
#conclusion:
```

```
Random Forest gives the best accuracy.
```