Online Payments Fraud Detection with Machine Learning

Sreeparna Ray

To identify online payment fraud with machine learning, we need to train a machine learning model for classifying fraudulent and non-fraudulent payments. For this, we need a dataset containing information about online payment fraud, so that we can understand what type of transactions lead to fraud.

Online Payments Fraud Detection using Python

In [9]:
```
import pandas as pd
import numpy as np
import plotly.express as px
```

In [2]:
```
data = pd.read_csv("data.csv")
```

In [3]:
```
print(data.head())
```

```
   step      type    amount     nameOrig  oldbalanceOrg  newbalanceOrig  \
0     1   PAYMENT   9839.64  C1231006815       170136.0       160296.36
1     1   PAYMENT   1864.28  C1666544295        21249.0        19384.72
2     1  TRANSFER    181.00  C1305486145          181.0            0.00
3     1  CASH_OUT    181.00   C840083671          181.0            0.00
4     1   PAYMENT  11668.14  C2048537720        41554.0        29885.86

      nameDest  oldbalanceDest  newbalanceDest  isFraud  isFlaggedFraud
0  M1979787155             0.0             0.0        0               0
1  M2044282225             0.0             0.0        0               0
2   C553264065             0.0             0.0        1               0
3    C38997010         21182.0             0.0        1               0
4  M1230701703             0.0             0.0        0               0
```

In [4]:
```
# Check whether this dataset has any null values or not
print(data.isnull().sum())
```

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

So, the above dataset does not have any null values

In [6]:
```
# Check the type of transaction mentioned in the dataset
print(data.type.value_counts())
```

```
CASH_OUT    2237500
PAYMENT     2151495
CASH_IN     1399284
TRANSFER     532909
DEBIT         41432
Name: type, dtype: int64
```

In [7]:
```
types = data["type"].value_counts()
transactions =  types.index
quantity = types.values
```

In [10]:
```
figure = px.pie(data, values=quantity, names=transactions, hole = 0.5, title = "Distribution of Transaction Ty
figure.show()
```

In [11]:
```
# Check the correlation between the features of the data with the isFraud column
correlation = data.corr()
print(correlation["isFraud"].sort_values(ascending = False))
```

```
isFraud           1.000000
amount            0.076688
isFlaggedFraud    0.044109
step              0.031578
oldbalanceOrg     0.010154
newbalanceDest    0.000535
oldbalanceDest   -0.005885
newbalanceOrig   -0.008148
Name: isFraud, dtype: float64
```

Now let's transform the categorical features into numerical. Here I will also transform the values of the isFraud column into No Fraud and Fraud labels to have a better understanding of the output.

In [12]:
```
data["type"] = data["type"].map({"CASH_OUT":1,"PAYMENT":2,"CASH_IN":3,"TRANSFER":4,"DEBIT":5})
data["isFraud"] = data["isFraud"].map({0:"No Fraud",1:"Fraud"})
```

In [13]:
```
print(data.head(20))
```

	step	type	amount	nameOrig	oldbalanceOrg	newbalanceOrig	\
0	1	2	9839.64	C1231006815	170136.00	160296.36	
1	1	2	1864.28	C1666544295	21249.00	19384.72	
2	1	4	181.00	C1305486145	181.00	0.00	
3	1	1	181.00	C840083671	181.00	0.00	
4	1	2	11668.14	C2048537720	41554.00	29885.86	
5	1	2	7817.71	C90045638	53860.00	46042.29	
6	1	2	7107.77	C154988899	183195.00	176087.23	
7	1	2	7861.64	C1912850431	176087.23	168225.59	
8	1	2	4024.36	C1265012928	2671.00	0.00	
9	1	5	5337.77	C712410124	41720.00	36382.23	
10	1	5	9644.94	C1900366749	4465.00	0.00	
11	1	2	3099.97	C249177573	20771.00	17671.03	
12	1	2	2560.74	C1648232591	5070.00	2509.26	
13	1	2	11633.76	C1716932897	10127.00	0.00	
14	1	2	4098.78	C1026483832	503264.00	499165.22	
15	1	1	229133.94	C905080434	15325.00	0.00	
16	1	2	1563.82	C761750706	450.00	0.00	
17	1	2	1157.86	C1237762639	21156.00	19998.14	
18	1	2	671.64	C2033524545	15123.00	14451.36	
19	1	4	215310.30	C1670993182	705.00	0.00	

	nameDest	oldbalanceDest	newbalanceDest	isFraud	isFlaggedFraud
0	M1979787155	0.0	0.00	No Fraud	0
1	M2044282225	0.0	0.00	No Fraud	0
2	C553264065	0.0	0.00	Fraud	0
3	C38997010	21182.0	0.00	Fraud	0
4	M1230701703	0.0	0.00	No Fraud	0
5	M573487274	0.0	0.00	No Fraud	0
6	M408069119	0.0	0.00	No Fraud	0
7	M633326333	0.0	0.00	No Fraud	0
8	M1176932104	0.0	0.00	No Fraud	0
9	C195600860	41898.0	40348.79	No Fraud	0
10	C997608398	10845.0	157982.12	No Fraud	0
11	M2096539129	0.0	0.00	No Fraud	0
12	M972865270	0.0	0.00	No Fraud	0
13	M801569151	0.0	0.00	No Fraud	0
14	M1635378213	0.0	0.00	No Fraud	0
15	C476402209	5083.0	51513.44	No Fraud	0
16	M1731217984	0.0	0.00	No Fraud	0
17	M1877062907	0.0	0.00	No Fraud	0

```
18   M473053293            0.0            0.00  No Fraud              0
19  C1100439041        22425.0            0.00  No Fraud              0
```

Online Payments Fraud Detection Model

Let train a classification model to classify fraud and non-fraud transactions.

In [14]:
```
# Before training the model, I will split the data into training and test sets
from sklearn.model_selection import train_test_split
```

In [15]:
```
x = np.array(data[["type","amount","oldbalanceOrg","newbalanceOrig"]])
y = np.array(data[["isFraud"]])
```

Let's train the online payments fraud detection model

In [17]:
```
# training a machine learning model
from sklearn.tree import DecisionTreeClassifier
xtrain, xtest, ytrain, ytest = train_test_split(x,y,test_size=0.10, random_state=42)
model = DecisionTreeClassifier()
model.fit(xtrain,ytrain)
print(model.score(xtest,ytest))
```

```
0.9997375295082843
```

Let's classify whether a transaction is a fraud or not by feeding about a transaction into the model

In [20]:
```
# prediction
# features = [type, amount, oldbalanceOrg, newbalanceOrig]
features = np.array([[4, 9000.60, 9000.60, 0.0]])
print(model.predict(features))
```

```
['Fraud']
```

So this is how we can detect online payments fraud with machine learning using Python.