The introduction of online payment systems has helped a lot in the ease of payments. But, at the same time, it increased in payment frauds. Online payment frauds can happen with anyone using any payment system, especially while making payments using a credit card. That is why detecting online payment fraud is very important for credit card companies to ensure that the customers are not getting charged for the products and services they never paid. If you want to learn how to detect online payment frauds, this article is for you. In this article, I will take you through the task of online payments fraud detection with machine learning using Python.

Online Payments Fraud Detection with Machine Learning

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. For this task, I collected a dataset from Kaggle, which contains historical information about fraudulent transactions which can be used to detect fraud in online payments. Below are all the columns from the dataset I'm using here:

step: represents a unit of time where 1 step equals 1 hour

type: type of online transaction

amount: the amount of the transaction

nameOrig: customer starting the transaction

oldbalanceOrg: balance before the transaction

newbalanceOrig: balance after the transaction

nameDest: recipient of the transaction

oldbalanceDest: initial balance of recipient before the transaction

newbalanceDest: the new balance of recipient after the transaction

isFraud: fraud transaction

I hope you now know about the data I am using for the online payment fraud detection task. Now in the section below, I'll explain how we can use machine learning to detect online payment fraud using Python.

In [1]:
```
#Online Payments Fraud Detection using Python
#I will start this task by importing the necessary Python libraries and the dataset we need for this task:
import pandas as pd
import numpy as np
data = pd.read_csv("credit_card.csv")
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
```

Now, let's have a look at whether this dataset has any null values or not:

In [3]:
```
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
```

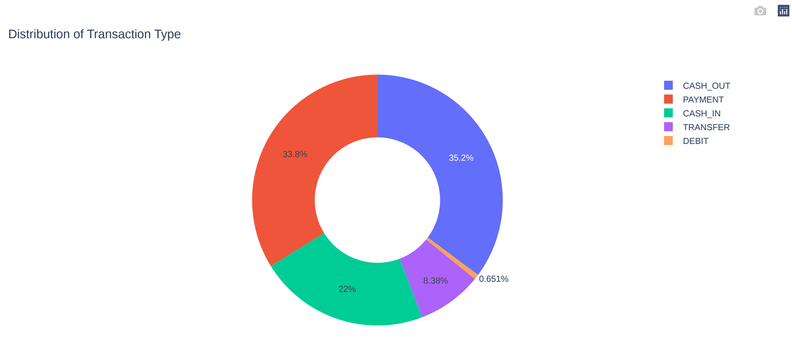
So this dataset does not have any null values. Before moving forward, now, let's have a look at the type of transaction mentioned in the dataset:

In [4]:
```
# Exploring transaction type
print(data.type.value_counts())
```

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

In [2]:
```
type = data["type"].value_counts()
transactions = type.index
quantity = type.values

import plotly.express as px
figure = px.pie(data,
                values=quantity,
                names=transactions,hole = 0.5,
                title="Distribution of Transaction Type")
figure.show()
```



Now let's have a look at the correlation between the features of the data with the isFraud column:

In [6]:
```
# Checking correlation
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 [7]:
```
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"})
print(data.head())
```

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

      nameDest  oldbalanceDest  newbalanceDest   isFraud  isFlaggedFraud
0  M1979787155             0.0             0.0  No Fraud               0
1  M2044282225             0.0             0.0  No Fraud               0
2   C553264065             0.0             0.0     Fraud               0
3    C38997010         21182.0             0.0     Fraud               0
4  M1230701703             0.0             0.0  No Fraud               0
```

Online Payments Fraud Detection Model

Now let's train a classification model to classify fraud and non-fraud transactions. Before training the model, I will split the data into training and test sets:

In [8]:
```
# splitting the data
from sklearn.model_selection import train_test_split
x = np.array(data[["type", "amount", "oldbalanceOrg", "newbalanceOrig"]])
y = np.array(data[["isFraud"]])
```

Now let's train the online payments fraud detection model:

In [9]:
```
# 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))
```

```
0.9997422445470576
```

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

In [11]:
```
# prediction
#features = [type, amount, oldbalanceOrg, newbalanceOrig]
features = np.array([[4, 9000.60, 9000.60, 0.0]])
print(model.predict(features))
```

```
['Fraud']
```

Summary

So this is how we can detect online payments fraud with machine learning using Python. Detecting online payment frauds is one of the applications of data science in finance.