**CREDIT CARD FRAUD DETECTION**

SUBMITTED BY:

NAME: Sarvesh Pandey

REGISTRATION NO.:11801771

SECTION: K18KK / GROUP-2

ROLL NO: 57

SCHOOL OF COMPUTER SCIENCE AND ENGINEERING

**LOVELY PROFESSIONAL UNIVERSITY**

**PHAGWARA, PUNJAB**

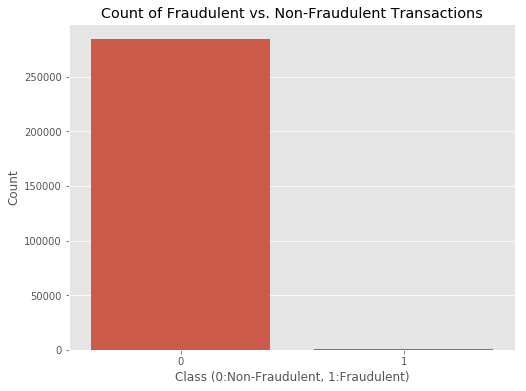
**INTRODUCTION**:

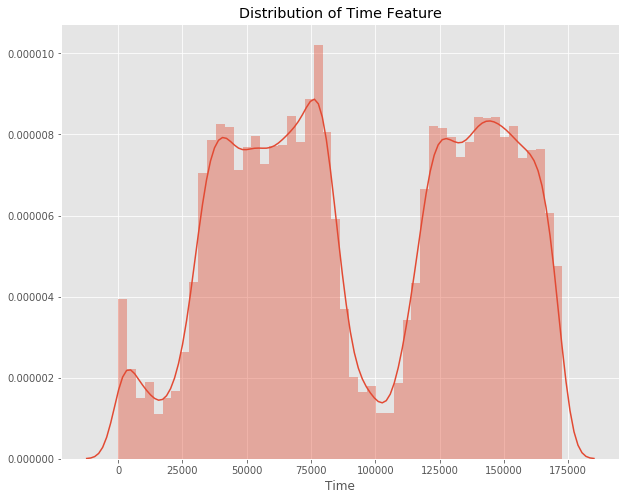
A Fraud in credit card transaction is committed when the transaction leading to purchase of a product is unauthorized and carried out by someone else than the owner of the Card. Although the cases of such fraudulent transactions can be minimized through proper prevention measures, still many of the transactions carried out today are fraudulent. So the Machine learning model, to be designed will help detect such fraudulent transactions, and help to minimize such activities in an automated manner.

Fraud detection involves monitoring the activities of the population of users to estimate and eliminate objectionable behavior. If the solution is not automated it is nearly impossible to carry out such operations by using man power.

**MOTIVATION:**

Going through various datasets on kaggle(A data analysis website), A dataset was available which focuses on Credit card detection, which consists of 31 features, 28 of which are made anonymous for hiding sensitive information of users, 3 are made visible namely **amount of transactions, time** and **class(1 for fraudulent and 0 for non fraudulent).** Plotting a Bar graph using the matplotlib library on python clearly shows the no of fraudulent transactions are very significant, which can be avoided. This led to the development of this model.





These graph show the time at which these transaction occurs which clearly shows that most of these fraudulent transactions occur during the night time.

**OBJECTIVE:**

Since, there are millions of transactions that occur on daily basis, most of them are non-fraudulent (99.8% source: kaggle) and the data available is mostly private, so most important objective of this project is to detect such fraudulent transactions which are misaligned and difficult to detect.

**SCOPE:**

The scope of project extend to wide variety of applications, since the algorithm used for the application is Isolation Forest Algorithm which provides functionalities such as classification, regression and outlier detection, it can be modified and the efficiency of the model can be increased to detect fraud in real time rather than using a dataset to do so.

**ABSTRACT**

One of the major ethical issues in Industry is Fraud. Primary Aim to identify credit cards companies and people who are borne to be a victim of these frauds. The sub-aim is to present, is to establish a method using current methods of fraud detection. Since the frauds carried out are in very large numbers, a lot of techniques were developed to keep them in check. The application Developed will help in identifying and minimizing the frauds which are being carried out by various unethical means. Although these methods are proven to be effective, sometimes many genuine cases are identified as fraudulent.

**METHODOLOGY**

The approach used in the program uses the latest machine learning algorithm called outliers to detect fraudulent cases. First a dataset is obtained from kaggle the details of which are mentioned in introduction. The dataset is now processed using graph tools. Since the dataset contains more than 25,000 sets of data we have to perform processing on only a part of data due to limitation of technology . The data is processed by a set of algorithms from modules. The data is fit into model and the following outlier detection modules are applied on it:

Local outlier Factor

Isolation Forest Algorithm

**CODE SNIPPET:**

#!/usr/bin/env python  
# coding: utf-8  
  
# In[3]:  
  
  
# import the necessary packages  
import numpy as np  
import pandas as pd  
import matplotlib.pyplot as plt  
import seaborn as sns  
  
  
# In[4]:  
  
  
# Load the dataset from the csv file using pandas  
data = pd.read\_csv('creditcard.csv')  
  
  
# In[4]:  
  
  
# Start exploring the dataset  
print(data.columns)  
  
  
# In[5]:  
  
  
# Print the shape of the data  
data = data.sample(frac=0.1, random\_state = 1)  
print(data.shape)  
print(data.describe())  
  
# V1 - V28 are the results of a PCA Dimensionality reduction to protect user identities and sensitive features  
  
  
# In[6]:  
  
  
# Plot histograms of each parameter   
data.hist(figsize = (20, 20))  
plt.show()  
  
  
# In[7]:  
  
  
# Determine number of fraud cases in dataset  
  
Fraud = data[data['Class'] == 1]  
Valid = data[data['Class'] == 0]  
  
outlier\_fraction = len(Fraud)/float(len(Valid))  
print(outlier\_fraction)  
  
print('Fraud Cases: {}'.format(len(data[data['Class'] == 1])))  
print('Valid Transactions: {}'.format(len(data[data['Class'] == 0])))  
  
  
# In[8]:  
  
  
# Correlation matrix  
corrmat = data.corr()  
fig = plt.figure(figsize = (12, 9))  
  
sns.heatmap(corrmat, vmax = .8, square = True)  
plt.show()  
  
  
# In[9]:  
  
  
# Get all the columns from the dataFrame  
columns = data.columns.tolist()  
  
# Filter the columns to remove data we do not want  
columns = [c for c in columns if c not in ["Class"]]  
  
# Store the variable we'll be predicting on  
target = "Class"  
  
X = data[columns]  
Y = data[target]  
  
# Print shapes  
print(X.shape)  
print(Y.shape)  
  
  
# In[11]:  
  
  
from sklearn.metrics import classification\_report, accuracy\_score  
from sklearn.ensemble import IsolationForest  
from sklearn.neighbors import LocalOutlierFactor  
  
# define random states  
state = 1  
  
# define outlier detection tools to be compared  
classifiers = {  
 "Isolation Forest": IsolationForest(max\_samples=len(X),  
 contamination=outlier\_fraction,  
 random\_state=state),  
 "Local Outlier Factor": LocalOutlierFactor(  
 n\_neighbors=20,  
 contamination=outlier\_fraction)}  
  
  
# In[15]:  
  
  
# Fit the model  
plt.figure(figsize=(9, 7))  
n\_outliers = len(Fraud)  
  
  
for i, (clf\_name, clf) in enumerate(classifiers.items()):  
   
 # fit the data and tag outliers  
 if clf\_name == "Local Outlier Factor":  
 y\_pred = clf.fit\_predict(X)  
 scores\_pred = clf.negative\_outlier\_factor\_  
 else:  
 clf.fit(X)  
 scores\_pred = clf.decision\_function(X)  
 y\_pred = clf.predict(X)  
   
 # Reshape the prediction values to 0 for valid, 1 for fraud.   
 y\_pred[y\_pred == 1] = 0  
 y\_pred[y\_pred == -1] = 1  
   
 n\_errors = (y\_pred != Y).sum()  
   
 # Run classification metrics  
 print('{}: {}'.format(clf\_name, n\_errors))  
 print(accuracy\_score(Y, y\_pred))  
 print(classification\_report(Y, y\_pred))

**LIBRARIES USED:**

Sklearn

Numpy

Scipy

matplotlib

**REFERENCES:**

<https://www.researchgate.net/publication/336800562_Credit_Card_Fraud_Detection_using_Machine_Learning_and_Data_Science>

<https://towardsdatascience.com/detecting-credit-card-fraud-using-machine-learning-a3d83423d3b8>

<https://www.geeksforgeeks.org/ml-credit-card-fraud-detection/>