**Credit Card fraud detection**

**Credit card fraud** is a wide-ranging term for [theft](https://en.wikipedia.org/wiki/Theft) and [fraud](https://en.wikipedia.org/wiki/Fraud) committed using a [payment card](https://en.wikipedia.org/wiki/Payment_card), such as a [credit card](https://en.wikipedia.org/wiki/Credit_card) or [debit card](https://en.wikipedia.org/wiki/Debit_card).

The purpose may be to obtain goods without paying, they have resulted in huge financial losses as the fraudulent transactions.

 Card fraud begins either with the theft of the physical card or with the compromise of data associated with the account, including the card account number or other information that would routinely and necessarily be available to a merchant during a legitimate transaction.

When a credit card is lost or stolen, it may be used for illegal purchases until the holder notifies the issuing bank and the bank puts a block on the account.

Most banks have free 24-hour telephone numbers to encourage prompt reporting. Still, it is possible for a thief to make unauthorized purchases on a card before the card is canceled. Without other security measures

A thief could potentially purchase thousands of dollars in merchandise or services before the cardholder or the card issuer realizes that the card has been compromised.

The mail and the Internet are major routes for fraud against merchants who sell and ship products and affect legitimate mail-order and Internet merchants.

**Problem Statement:**

The Credit Card Fraud Detection Problem includes modeling past credit card transactions with the knowledge of the ones that turned out to be fraud. This model is then used to identify whether a new transaction is fraudulent or not. Our aim here is to detect 100% of the fraudulent transactions while minimizing the incorrect fraud classifications.

**Observation:**

The data set has 31 features, 28 of which have been anonymized and are labeled V1 through V28. The remaining three features are the time and the amount of the transaction as well as whether that transaction was fraudulent or not. Before it was uploaded to Kaggle, the anonymized variables had been modified in the form of a PCA (Principal Component Analysis) for privacy. The data set contains 284,807 transactions. The mean value of all transactions is $88.35 while the largest transaction recorded in this data set amounts to $25,691.16. this data set includes all transactions recorded over the course of two days. most transactions are non-fraudulent. In fact, 99.83% of the transactions in this data set were not fraudulent while only 0.17% were fraudulent.

The data set is highly skewed, consisting of 492 frauds in a total of 284,807 observations. This resulted in only 0.172% fraud cases. This skewed set is justified by the low number of fraudulent transactions. we assume that the ‘Time’ feature has little or no significance in classifying a fraud transaction. There is no missing value in the dataset.

**Backend Stuff**

Credit Card Fraud Detection is a typical example of classification. In this process, we have focused more on analyzing the feature modeling and possible business use cases of the algorithm’s output than on the algorithm itself. We used necessary libraries like numpy for numerical computations , pandas dataframe for data analyzation, matplot library to plot the graph.We used supervised machine learning for training the data set, we imported some of the necessary libraries like keras.

First we imported the necessary libraries and imported dataset which contains nearly 284k transactions which is in the form of .csv extension.

We do check for the null values present in the dataset for accuracy as well as to prevent errors. Dependent and independent variables are declared

We filtered out and considered some of the necessary features such as amount ,v1,v2,v3,v4,v5,v6 and amount as inputs. Class is output which is a target {0:'Not Fraud', 1:'Fraud'}

# Normalizing the data

Using StandardScaler we are normalizing the dataset in order to have no bias. By applying StandardScaler we will simply transform the data such that its distribution will have a mean value 0 and standard deviation of 1.

# ****Splitting the data in train and test****

For the training dataset, we are excluding the fraudulent transactions. As our aim is to detect a fraudulent transaction, we will specifically train or data on the normal transaction so that when a fraudulent transaction occurs our autoencoder can easily predict it with high reconstruction error.

Models in Keras are defined as a sequence of layers.

1st define the input layer has the right number of inputs which is done by defining, in input dimensions our case it is 7 columns.

We can specify the number of neurons in the layer as the first argument, the initialization method as the second argument as **int** and specify the activation function using the **activation** argument.

We will use the rectifier (‘**relu**‘) activation function.

The **batch size** is the number of data in each portion. After we are done with all batches, we complete a single epoch. In order to have better accuracy, we should train data for large no. of epochs. In other words, **Epoch describes** the number of times the algorithm sees the entire data set. So, each time the algorithm has seen all samples in the dataset, an epoch has completed.

After this we are predicting the dataset as well as comparing y\_pred and y\_test for comparing accuracy is matched or not.

**Deployment part in IBM Watson Studio**

Importing the .ipynb file into the Watson studio by creating a new Watson studio project. Again inserting the dataset into the uploaded project and selecting the insert to pandas dataframe.

After that we install Watson studio services with the command !pip install watson studio. We do some of the things like saving the model with .h5 extension and converting into .tgz and inserting some credentials from service credentials.

We deploy the model and find the scoring endpoint for the model.

**Conclusion & Future Work**

Fraud detection is a complex issue that requires a substantial amount of planning before throwing machine learning algorithms at it. Nonetheless, it is also an application of data science and machine learning for the good, which makes sure that the customer’s money is safe and not easily tampered with.

Future work will include a comprehensive tuning of the Random Forest algorithm I talked about earlier. Having a data set with non-anonymized features would make this particularly interesting as outputting the feature importance would enable one to see what specific factors are most important for detecting fraudulent transactions.