# Step 1: Import Libraries and Necessary Installation

Load the necessary libraries and do required installation.

Text, letter

Description automatically generated

**Step 2: Read file.**

Load the transaction and identity datasets. Check the wall time. We check memory usage using the info method in Pandas.

**Step 3: Merging both the datasets**

We merge datasets transaction (train\_tr) and identity (train\_id) into one dataset so that we can perform the analysis and related data handling activities on one dataset. We did left outer join as every transaction id in the transaction dataset may not have equivalent record in the identity dataset and we do not want to miss any transaction records.

**Step 4: Exploratory Data Analysis (EDA)**

We analyzed the shape, feature size and statistics for the merged dataset. The percentage of Fraud (only 3.5%) vs Normal transactions in the dataset helped us to understand the imbalance in the dataset. The statistics for Normal transactions in the Dataset and the statistics for Fraud transactions in the Dataset shows that we cannot identify a fraud transaction only by amount. We checked for duplicates and outliers. We separated out numerical and categorical variables so that we can process them accordingly during Feature Engineering. We found correlation between the attributes using Correlation Matrix. V1 to V11 has high correlation with D6 to D14 as well as with other V attributes. The C\* (i.e., C1, C2, ...) attributes are more correlated within themselves with respect to D\* (i.e., D1, D2, ...) attributes internal correlation. Also C\* attributes are not much correlated with D\* attributes. V107 is highly correlated with most of the other V\* attributes. dst1 is highly correlated with a large number of V\* columns.

**Step 7: Feature Engineering**

As part of feature engineering, we checked for duplicates and Null values. There were no duplicates to handle. We filled NaN values with zeroes as there are many columns having Null value, droping the columns with Null values will delete a lot of data where we will be missing other important non-NaN attributes for the same recordset. Most of the ML algorithms perform better and produce better accuracy with numerical variables. Hence, we did Label Encoding to convert categorical variables into numerical variables. Chi square test does not apply to negative values. That’s why we checked for any negative values and replaced the negative values with zeroes.

**Step 8: Feature Selection**

In Feature selection we **reduced the number of input variables and selected features of high importance.** We have used Chi square Test for feature selection. First, we have separated the target (isFraud) variable as a separate dataframe (Y\_data). Copied the original dataset excluding the target variable into a separate dataframe(df\_input). This helped us to keep the original dataset intact, so that we can refer to it whenever required. Then we have used the SelectKBest algorithm with Chi square function to score the attributes using df\_input and Y\_data. Based on descending order of the scores, we selected the best 150 features to be utilized for next step of model training. We merged the Y\_data (the target isFraud variable) to these 150 features to prepare the dataset (df\_selected\_features\_io) which will be subjected to Train-test Split.

**Step 9: Split the dataset into train and test.**

We split the merged dataset (df\_selected\_features\_io) containing 151 columns into train and test, i.e 80% of the dataset is used for training the model and 20% of the dataset is used for testing the model. Then we separated the output (Y) variable (‘isFraud’) from input (X) variable for both train and test datasets.

**Step 10: Model Selection**

We checked the Accuracy and F1 scores for some ML models and selected the model with highest Accuracy and F1 score.

1. Simple Random Forest Model

Roc Auc Score: 62%

F1 score: 0.37

1. Decision Tree

Roc AUC Score: 97%

F1 Score: 0.22

1. K-Nearest Neighbors

Roc AUC Score: 96%

F1 Score: 0.04

1. Logistic Regression

Roc AUC Score: 96%

F1 Score: 0.002

1. Random Forest

Roc AUC Score: 97%

F1 Score: 0.27

1. XGBoost

Roc AUC Score: 98%

F1 Score: 0.68

The F1 score was an important differentiator in this model selection, because F1 score being harmonic mean of Precision and Recall, handles the imbalance of the dataset while calculating the metric. This metrics shows that XGBoost is having a high F1-score along with greater accuracy with respect to other models.

**Step 11: Balancing the unbalanced data**

We balanced our imbalanced Train dataset using SMOTE. In the original dataset, we noticed that it contains only 3.5% fraudulent transactions which represents the minority class. Using SMOTE data augmentation methodology, helps to synthesize examples of the minority class without adding any new information.

**Step 12: Model Training after balancing the data**

After applying the SMOTE, we re-executed the XGBoost model and captured the performance metrics. We noticed the Accuracy score of the XGBoost model remained 98% and the F1 score of the XGBoost model is 0.6684412895593019, which is still the highest among the other

Models tested for the purpose. The ROC Curve shows that the model has higher chance of predicting the probability that an online transaction is fraudulent based on the AUC score of 0.92 which is much higher compared to no-skill line of 0.5.