**Problem Statement**

The aim of this project is to predict fraudulent credit card transactions using machine learning models.

The data set for this project was obtained from Kaggle. It contains thousands of individual transactions that took place over a course of two days and their respective labels.

Download Data: <https://www.kaggle.com/mlg-ulb/creditcardfraud>

This data set is highly unbalanced, with the positive class (frauds) accounting for only 0.172% of the total transactions. The data set has also been modified with principal component analysis (PCA) to maintain confidentiality. Apart from ‘time’ and ‘amount’, all the other features (V1, V2, V3, up to V28) are the principal components obtained using PCA. The feature 'time' contains the seconds elapsed between the first transaction in the data set and the subsequent transactions. The feature 'amount' is the transaction amount. The feature 'class' represents class labelling, and it takes the value of 1 in the cases of fraud and 0 in others.

The project pipeline can be briefly summarized in the following four steps:

Data understanding: Here, we need to load the data and understand the features present in it. This would help to choose the features that we will need for r final model.

Exploratory data analytics (EDA): Usually, in this step, we need to perform univariate and bivariate analyses of the data, followed by feature transformations, if necessary. For the current data set, because Gaussian variables are used, we do not need to perform Z-scaling. However, we can check if there is any skewness in the data and try to mitigate it, as it might cause problems during the model building phase.

if the values of any independent feature are skewed, depending on the model, skewness may affect model assumptions or may impair the interpretation of feature importance.

Train/Test split: Now, we can perform to check the performance of our models with unseen data. Here, for validation, we can use the k-fold cross-validation method. We need to choose an appropriate k value so that the minority class is correctly represented in the test folds.

Model building / hyperparameter tuning: This is the final step at which we can try different models and fine-tune their hyperparameters until we get the desired level of performance.