Case Study: **Credit Card Fraud Detection**

**Learner Name: Rudrakanta Ghosh**

**Problem Description:**

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

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

**Data Description:**

The data set includes credit card transactions made by European cardholders over a period of two days in September 2013. Out of a total of 2,84,807 transactions, 492 were fraudulent.

This data set is highly unbalanced, with the positive class (frauds) accounting for just 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 are the principal components obtained using PCA.

**Approach:**

1. **Understand Data**:Here, we need to load the data and understand the features present in it. This would help us choose the features that will be needed for final model.
2. **Perform EDA**: Since we have features that are principal components obtained using PCA, all of them are Gaussian variables. We do not need to do Z-scaling or do univariate or bivariate analysis.

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 as some of the data points causing the skewness can be outliers.

1. **Handling Data Imbalance**: In our dataset we have very a smaller number of class label as 1 as there are very a smaller number of fraudulent transactions overall. So we can’t use this dataset directly for the model building step. This is known as minority class problem.

We have below options to mitigate this:

1. Removal of majority class rows
2. Oversampling minority class
3. SMOTE: a process where we can generate new data points, which lie vectorially between two data points that belong to the minority class
4. ADASYN: This is similar to SMOTE, with a minor change i.e. the number of synthetic samples that it will add will have a density distribution

We will check pros and cons of each of these steps during model building.

1. **Train/Test Split**: We need to perform this step to check the performance of our models with unseen data. Here 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.
2. **Model Building:**

We will try with different classifications modelling technique to get the desired performance (accuracy, precision and recall). Below are some of the modelling techniques:

1. Logistic regression
2. KNN
3. Decision Trees and Random Forest
4. XGBoost
5. **Hyperparameter Tuning:**  Once we have different model execution data, we can fine-tune their hyperparameter to get the desired level of performance.

Since we have a very small volume of data, it is better to use “K-Fold Cross Validation” approach instead of “hold-out approach” (train-test-val) split of the data.

For “K-Fold Cross Validation” data, its better to use “Stratified K-Fold Cross Validation” approach as we have highly imbalanced data.

For Hyperparameter