Credit Card Fraud Detection

By - Swati Acharya

Problem Statement:

A credit card is one of the most used financial products to make online purchases and payments. Though the Credit cards can be a convenient way to manage your finances, they can also be risky. Credit card fraud is the unauthorized use of someone else's credit card or credit card information to make purchases or withdraw cash.
It is important that credit card companies are able to recognize fraudulent credit card transactions so that customers are not charged for items that they did not purchase.
The dataset contains transactions made by credit cards in September 2013 by European cardholders. This dataset presents transactions that occurred in two days, where we have 492 frauds out of 284,807 transactions. The dataset is highly unbalanced, the positive class (frauds) account for 0.172% of all transactions.
We have to build a classification model to predict whether a transaction is fraudulent or not.

Approach:

The approach to the problem can be divided into below parts:

1. **Load the Dataset & Data Understanding**

   First load the dataset given in a .csv format. Understand the Independent variables & Dependent Variables. As it looks that data is highly imbalanced with fraud data count is 492 and non-fraud is 284315 out of 284807. Also apart for amount & time feature all other features are PCA applied hence it is kept confidential.

   The positive class (frauds) account for only 0.172% of all transactions:

Class	0	1
Count	284315	492

2. **Data Preparation and EDA**

   Class is the **target variable** which we have to predict where 0 is normal transaction and 1 is fraudulent transaction.
   V1, V2, V3…. Amount, Time are **predictor variables.**

   Features V1, V2, … V28 are **Principal Component Analysis** applied features obtained with PCA, the only features which have not been transformed with PCA are 'Time' and 'Amount'. Feature 'Time' containsthe seconds elapsed between each transaction and the first transaction in the dataset. The feature 'Amount' is the transaction Amount.

   Since the PCA transformed variable are already Gaussian, there is **no need for normalization**.

   We can start with the **basic EDA like correlation, boxplots etc for outliers**. Check univariate, bivariate analysis.

3. **Feature Scaling**

   We can scale Amount & time columns if necessary.

4. **Data Skewness**

   - Next, we can use **transformation to handle and check the skewness in the data**. (Power Transformer, Box-Cox,Log transformation, Yeo-Johnson etc.)

5. **Feature Selection**

   - We can check for correlation & multi-collinearity between the independent variables using heat map. And other visualization plots.
   - We will drop the columns which are not required.
   - Check the relationship between Amount predictor and class target variable & same for time vs class.
   - Also we can check the data proportion ratio on the target variable for fraud vs non-fraud data.

6. **Class Imbalances:**

   - As we can see that the data is highly imbalanced with 0.17% data as fraud and 99.8% as non-fraud the model will become biased. Hence we will apply some balancing techniques. Next, we will try the below class imbalance handling techniques:

- **Random Over-Sampler** is a process which will random balance the data proportion between two binary outcomes

- **SMOTE** is a process where you can **generate new data points**, which lie vectorially between two data points that belong to the minority class.

- **ADASYN** is similar to SMOTE, with a minor change i.e. the number of synthetic samples that it will add will have a **density distribution**. The aim here is to create synthetic data for minority examples that are harder to learn, rather than the easier ones.

7. **Train test Split:**

   - We will split the data into train test split (80-20 ratio can be used).

8. **Model Selection and Model Building**:

   - We need to find which ML model works good with the imbalance data and have better result on the test data.
   - **Logistic regression** works best when the data is **linearly separable** and **needs to beinterpretable.**
   - **KNN, SVM, Random Forest** is also highly interpretable, but not preferred when we have a huge amount of data**it will consume a lot of computation.**
   - **The decision tree model** is the first choice when we want the output to be **intuitive,** butthey tend to over fit if left unchecked.
   - **XGBoost** is an extended version of **gradient boosting**, with additional features likeregularization and parallel tree learning algorithm for finding the best split.

We will start with the **Logistic Regression** model with the different value of regularization andhyper parameter tuning, then go to the **Decision Tree** & **XGBoost**.

9. Hyper Parameter Tuning:

- When the data is imbalanced or less, it is better to use **K-Fold Cross Validation** for evaluating the performance when the data set is randomly split into 'k' groups.
- **Stratified K-Fold Cross Validation** is an extension of K-Fold cross-validation; in which we rearrange the data to ensure that each fold is a good representative of all the strata of thedata.
- When you have a small data set, the computation time will be manageable to test out different hyper parameter combinations. In this scenario, it is advised to use a grid search.
- But, with large data sets, it is advised to use a randomized search because the sampling willbe random and not uniform.

In our case, we will check with **K-Fold**, **Repeated K-Fold** & **Stratified K-Fold Cross Validation** on the dataset and check the model performance.

10. Model Evaluation:

- We will use **ROC_AUC_Score** for this as AUC and ROC metric in sklearn is usedas the metric for highly imbalanced data-set, rest all fails.
- ROC have better false negative than the false positives.ROC-Curve = Plot between TPR and FPR
- -The threshold with highest value for TPR-FPR on the train set is usually the best cut-off. We **should not use the confusion matrix** as the performance metrics as well as they haveinternally defined hard threshold of 0.5.
- We also **can't completely rely on the precision, recall and F1-score** for now as they alsohave their strings attached of some threshold value.
- ROC curve takes into cognizance of all the possible threshold values.
- The **ROC curve** is used to understand the strength of the model by evaluating the performance of the model at all the **classification thresholds**.
- Because the ROC curve is measured at all thresholds, the best threshold would be one atwhich the **TPR is high and FPR is low, i.e., misclassifications are low.**

11. Cost-Benefit Analysis:

- Depending on the use case, we have to account for what we need: high precision or highrecall.
- For banks with smaller average transaction value, we would want high precision because weonly want to label relevant transactions as fraudulent. For every transaction that is flagged as fraudulent, you can add the human element to verify whether the transaction was done by calling the customer. However, when precision is low, such tasks are a burden because the human element has to be increased.
- For banks having a larger transaction value, if the recall is low, i.e., it is unable to detecttransactions that are labelled as non-fraudulent. So consider the losses if the missed transaction was a high-value fraudulent one, for e.g., a transaction of $10,000?

So here, to **save banks from high-value fraudulent transactions, we have to focus on a high recall in order to detect actual fraudulent transactions.**