Credit Card Default Payments

Georgios Markos Chatziloizos

1 Introduction

The credit card default payment problem is extremely important for financial institutions. In our case, it is a binary classification task that aims to forecast if a credit card user will default on their payment or not. We will use both simple and complex models, but also try to find out which are the most crucial features of our dataset. The dataset is an imbalanced one, so, there will be used techniques in order to mitigate the risk of our models predicting only the majority class.

2 Exploratory Data Analysis

For the Exploratory Data Analysis, we have created boxplots, histograms and the correlation matrix among the features of the dataset. Also, pandas-profiling provides us with a full report of each combination and histogram possible (DataAnalysis.html). More graphs with matplotlib and seaborn are presented at the GitHub link: `https://github.com/GeoMarX/CreditDefault`.

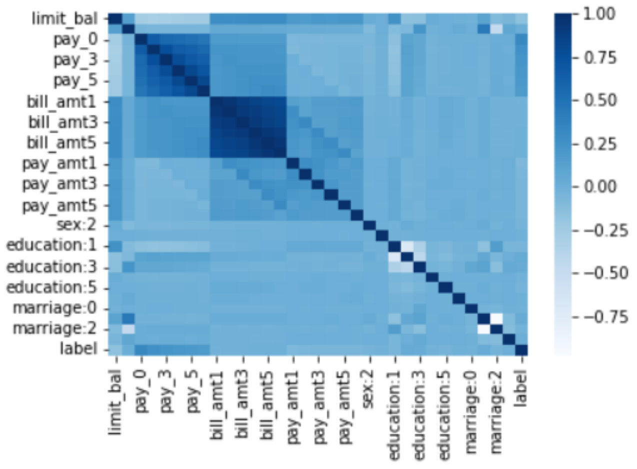

Figure 1 shows that the bill_amt features are strongly correlated with each other and also the pay features are also correlated with each other.

Figure 1: Correlation matrix of the dataset

3 Pre-Processing & Treating the Imbalance

Initially, the sex:1 feature is dropped as it contains the same information as sex:2 (if sex:1 is 0 then sex:2 is 1 and vice versa).

We utilized the SMOTE technique, which generates new samples by randomly selecting samples from the minority class that are "close" to them.

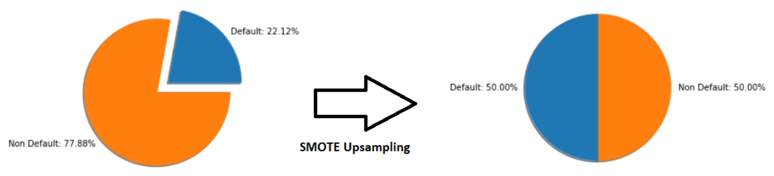

Figure 2: Labels percentage of the training dataset before and after SMOTE Upsampling

The SMOTE method mitigates the impact of the imbalanced datasets problem and it is only used at the training dataset as we want to preserve the imbalance of the test dataset. Why? Because we want the real data that our model will actually encounter in the real world, to be reflected in the test dataset (with the same ratios) and the model to be less biased. To accomplish that, we used the StratifiedShuffleSplit and not the train_test_split as the latter would lead to a biased model, in case that all the data of the minority class are included only in the training or even only in the test set.

4 Predictive models

4.1 Models

Metrics: Firstly, in order to decide the performance of models when working with imbalanced datasets, accuracy is not a sufficient metric. For this project, we are going to also use precision, recall, F1 score and AUC.

Models: In order to predict the binary outcome, we employed the following predictive models:

- Decision Trees with max depth 5.
- GaussianNB, a classifier based on Bayes' theorem.
- MLP, an ANN with 15 hidden layers.
- XGB, gradient boosting algorithm that uses decision trees with max depth 5.
- SVC, this classifier performs really well in high-dimensional spaces and maximizes the margin between decision boundary and support vectors.
- Random Forest, an Ensemble learning method that creates multiple decision trees and averages their predictions.
- K-NN, a label is given to a data point based on the majority class of its k=7 nearest neighbors.

4.2 Explainability

For the Random Forest algorithm, we used an explainability library (SHAP) to see which features and at what values affect the model.

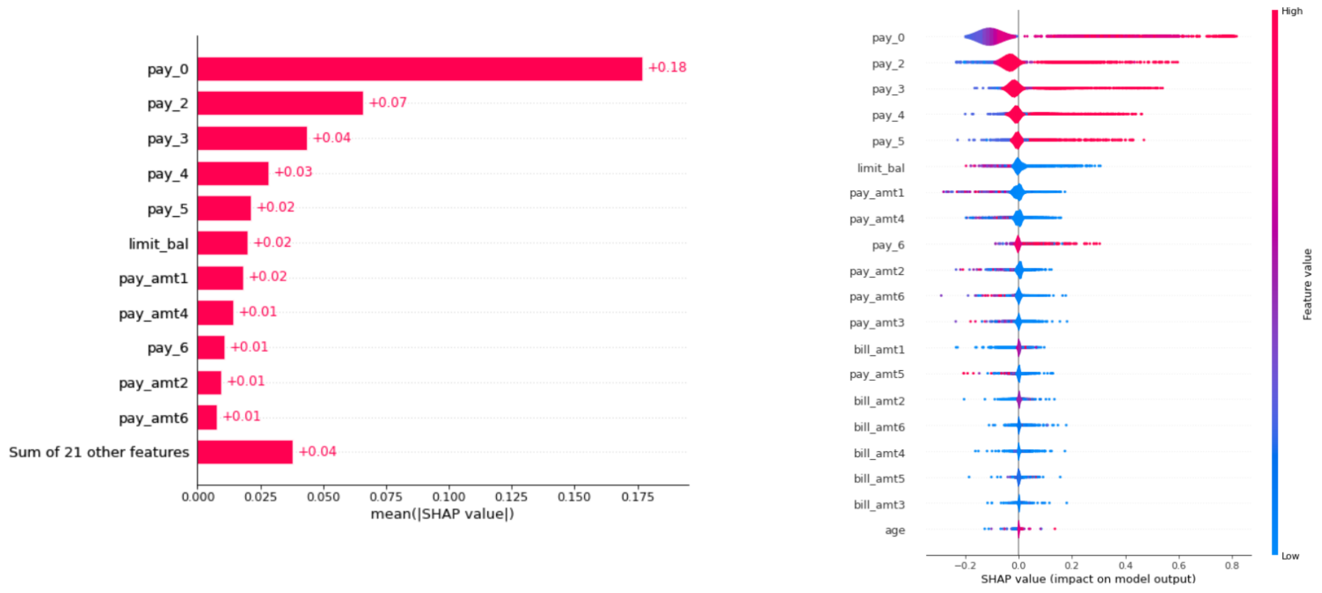

Figure 3: Shap Values

The features on the left of Figure 3 are sorted in absolute values from the highest to the lowest effect on the prediction, so it does not take into account whether the feature affects the prediction in a positive or negative way. So, the model is affected mostly by pay_0 and pay_2 . On the right, the features are also sorted by their effect on the prediction, but we can also see how higher and lower values of the feature will affect the result. Higher values at pay_0 and pay_2 indicate that the person will default.

4.3 A More Complex Deep Learning Model

This DL model is a much smaller type of MixNet with 5 bottlenecks having as kernels 3x3 and 5x5. At the end of each bottleneck there is a Squeeze-and-Excitation (SE) block. For the SE blocks the ratio is 8, because with larger ratio such as 16, the squeeze filters were very few and thus, important information was lost during the process. Also, the swish activation function was employed over ReLU as it usually has better performance. The parameters of the model were 92,873.

The Adam optimizer was used and the ExponentialDecay as a learning scheduler in order to gradually decrease the learning rate as the epochs progress, but also for the model to converge effectively. Finally, the binary crossentropy was used as the loss function.

5 Results

Model	Precision	Recall	F1 Score	Accuracy	AUC
Decision Trees	0.49	0.51	0.50	0.78	0.75
GaussianNB	0.23	0.95	0.38	0.30	0.72
MLP	0.46	0.61	0.52	0.76	0.76
XGB	0.62	0.41	0.49	0.81	0.75
SVC	0.49	0.54	0.52	0.77	0.73
Random Forest	0.51	0.55	0.53	0.78	0.76
KNN	0.35	0.61	0.44	0.66	0.69
Deep Learning Model	0.43	0.63	0.50	0.73	0.76

Table 1: ML and DL Models Performance for our Metrics

The results show that the XGB has the highest precision and the Random Forest has the highest F1 score. The GaussianNB has the highest recall, but underperforming in each and every other metric! Moreover, XGB and Decision Trees have the highest accuracy (0.81 and 0.78, respectively). The AUC score ranges from 0.69 to 0.76.

For the Deep learning model, it didn't have astonishing results but maybe with better GPUs (and more time on my end to tune hyperparameters, more blocks, etc etc) , we could achieve better results. But for now, simpler (model) is better!

6 Conclusion

To sum up, we had to deal with an imbalanced dataset where we used the StratifiedShuffleSplit in order to preserve the imbalanced ratio at the test set and the SMOTE algorithm for upsampling the training dataset in such way that our labels are split 50-50. These techniques are proven to provide us with better results when using ML/DL models. For the models in this project, the metrics were not only accuracy, but also, recall, precision, F1 score and AUC as they can provide us with better understanding when dealing with this type of datasets.