Final Course Project: Predicting Credit Card Default

16.02.2021

by Tom K. Walter

"In order to increase market share, card-issuing banks in Taiwan over-issued cash and credit cards to unqualified applicants. At the same time, most cardholders, irrespective of their repayment ability, overused credit cards for consumption and accumulated heavy credit card debts. The crisis caused [a] blow to consumer finance confidence and it is a big challenge for both banks and card-holders."
I-Cheng and Che-hui, 2009.[1]

1. Dataset and Goal

For my project, I have chosen the Taiwan Credit Card Default Dataset, which was originally aggregated and analyzed by I-Cheng Yeh and Che-hui Lien in their paper *"The comparisons of data mining techniques for the predictive accuracy of probability of default of credit card clients"* in the journal Expert Systems with Applications (2009).[2] The dataset is available on the UCI Machine Learning Repository.[3] It contains 30,000 observations, which correspond to clients that have been issued credit cards, and 25 columns that provide demographic information and the payment history of each client. The demographic information include marital status, sex, age, and education level. The payment record variables track the timely payment (PAY) over the last 6 months, the amount (BILL_AMT) due for each of the 6 months, and the amount paid (PAY_AMT) for each of the 6 months. The target variable, which I have renamed into DEFAULT, indicates whether or not a client has defaulted in the month following the observed time period. Out of 30,000 credit card holders, 23,272 have continued to pay off their bill and 6,625 did default. This makes it an unbalanced dataset. See Table 1, for a more detailed breakdown of all variables.

Table 1:

Name	Description	Type
ID	index number assigned to each customer	(useless for ML)
LIMIT_BAL	amount of the given credit in New Taiwan Dollar (NT$)	continuous
SEX	gender (1= male; 2= female)	categorical
EDUCATION	level of education (1= graduate school; 2= university; 3= high school; 4= others)	categorical
MARRIAGE	marital status (1= married; 2= single; 3= others)	categorical
AGE	age in years	continuous
PAY_n	history of past payment; *n* tracks the 6 months of payment record (from April to September, 2005); the repayment status is: -1= pay duly, 1= payment delay for one month, 2= payment delay for two months, etc.	ordinal
BILL_AMT_n	amount of the bill in NT$ (*n* tracks the 6 months)	continuous
PAY_AMT_n	amount paid in NT$ (*n* tracks the 6 months)	continuous

[1] Yeh, I-Cheng, and Che-hui Lien. "The comparisons of data mining techniques for the predictive accuracy of probability of default of credit card clients." Expert Systems with Applications 36.2 (2009): 2473-2480.
[2] Ibid.
[3] Https://archive.ics.uci.edu/ml/datasets/default+of+credit+card+clients.

DEFAULT	whether or not client has defaulted (Yes= 1, No= 0)	binary target

The research task associated with the dataset is to predict whether or not clients will default on their credit card, and to gain a better understanding of credible or non-credible clients in the future. In order to complete this task, I will develop two tree-based models and two "scaled" models using the Python Sklearn library.[4] To optimize the models' performance further, I will utilize Sklearn's stratified k-fold and grid-search cross validation functions. However, before modelling, I will briefly discuss preprocessing and address the issues of the unbalanced target classes. I intend to downsample both the majority and the minority class to speed up computation and balance the dataset. Moreover, I will also engage in some exploratory analysis to ensure that the downsampled dataset accurately reflects on information provided in the original dataset (otherwise it would be useless for modelling). Lastly, after the building and optimizing the models, I will evaluate and compare their performances.

2. Preprocessing, Balancing, and EDA

2.1 Data Cleaning and Preprocessing

To prepare a dataset for ML analysis, it must be checked for duplicate and missing values.
All variables must also be sorted and encoded to be ready for the algorithm. The ID column has been dropped because it does not contain relevant information for modelling.

Given the large size of the dataset, missing and duplicate values have been dropped. 35 missing values have been dropped and only their original instances have been kept. For the categorical variables EDUCATION and MARRIAGE, I have found additional values to those mentioned in the description, as Image 1 shows. I assume that a zero stands for missing entries and thus I have dropped 68 instances where EDUCATION or MARRIAGE were equal to 0. Further, I assume that the values 5 and 6 represent further education or degree types that are just not mentioned in the description. Therefore, I have kept them in the dataset.

Image 1:

```
[8]: # check categorical variables for missing values
     cat_col=['SEX', 'EDUCATION', 'MARRIAGE']
     for col in cat_col:
         print(col,':', df[col].unique())

     SEX : [2 1]
     EDUCATION : [2 1 3 5 4 6 0]
     MARRIAGE : [1 2 3 0]

[9]: # assume missing values have been entred as 0
     # how many missing values
     len(df.loc[(df['EDUCATION']==0) | (df['MARRIAGE']==0)])

[9]: 68

[10]: # drop rows where EDUCATION or MARRIAGE = 0
      df= df.loc[(df['EDUCATION'] != 0) & (df['MARRIAGE'] != 0)]
      df.shape

[10]: (29897, 24)
```

Lastly, I have transformed these 3 categorical variables into dummies. All remaining variables already had the data type int64 and were kept.

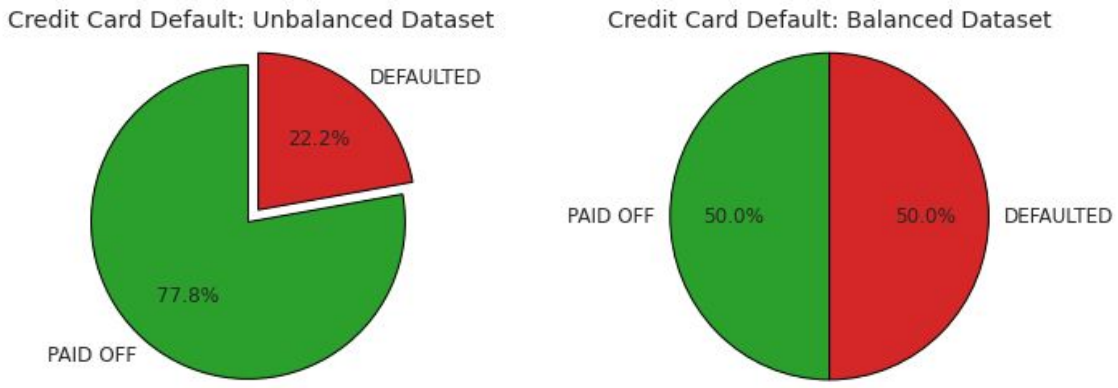
[4] Https://scikit-learn.org/stable/.

2.2 Balancing the Dataset

The classification of "rare events", e.g. disease occurrence, fraud detection, accidents, or credit card default, is a special type of classification problem as its datasets are often highly unbalanced. This means that the occurrence of a rare event are the minority of samples and the non-occurrence are the majority of samples in the dataset. The same issue is given in the Taiwan Credit Card Default dataset, where 77.8% of cardholders did not default and 22.8% did.

This is a problem because most ML algorithms for classification expect the target to be balanced. There are two common best practice methods of dealing with unbalanced classes. The first method is to balance the target classes by downsampling the majority class or upsampling the minority class. The second method is to only use algorithms that can adjust the weight given to each class. As mentioned before, I am choosing to downsample both the majority and minority class to the size of 2,000 samples per class. Beyond balancing the dataset, the major advantage of downsampling both classes is that it speeds up computation during modelling. A potential pitfall could be that the downsampled dataset no longer reflects the information given in the original dataset like the distribution of the input features.

Image 2:



2.3 Exploratory Data Analysis

To ensure that the downsampled dataset has retained the same information as the original dataset, I have compared the correlation of input features with the target as well as the distribution of LIMIT_BAL and AGE. Although one can hardly expect the same distribution between the original and downsampled dataset, similar distributions indicate that it can be used for modelling. If the distributions are too different that means the downsampled dataset does not accurately mirror the underlying relation between input and output, and cannot be used for modelling.

Image 3 shows the correlation of all input features with the target variable across both datasets. Although the size is slightly stronger, all downsampled features have a similar Pearson correlation coefficient compared to the ones in the original dataset.

Image 3:

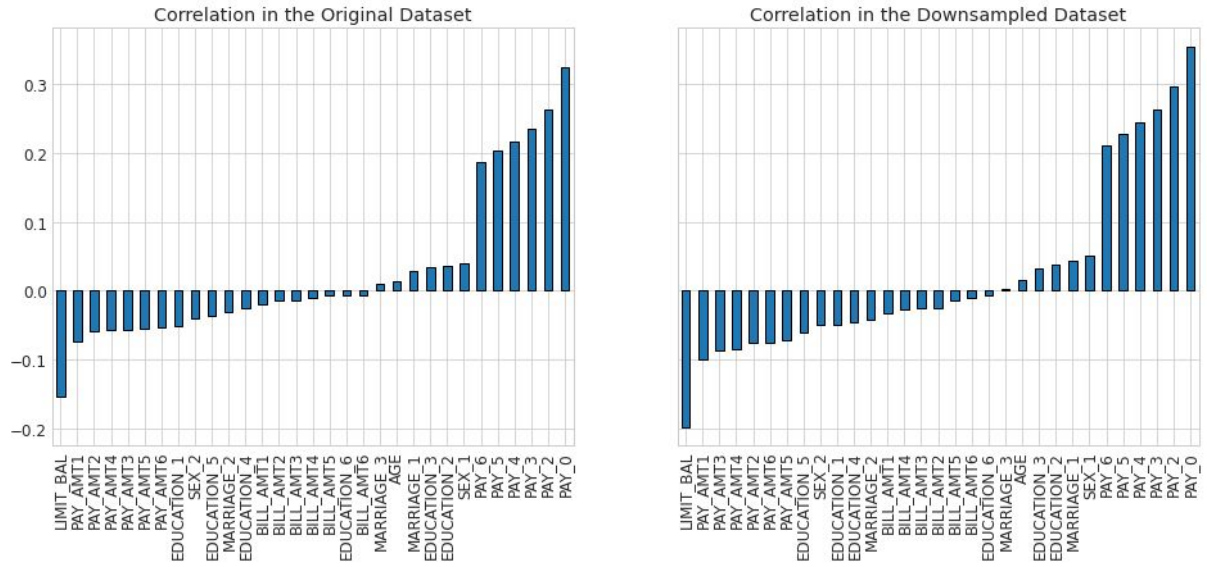
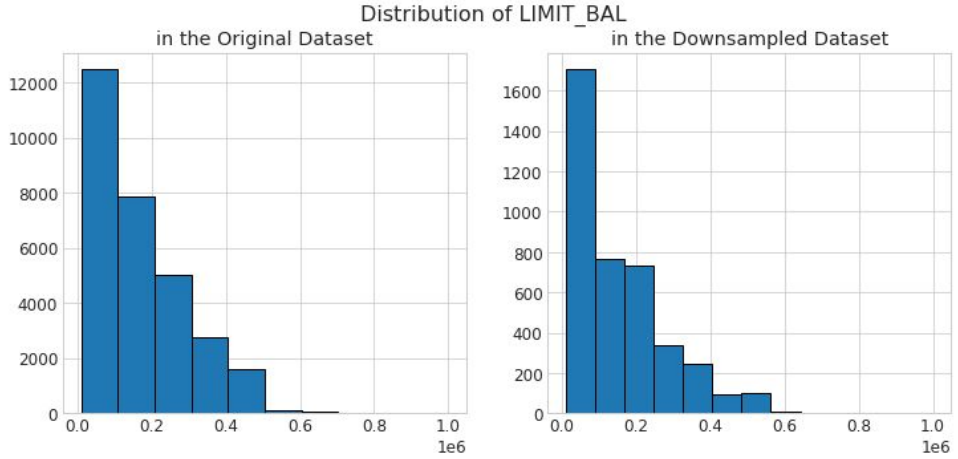


Image 4 shows a distribution histogram of the credit amount given to each cardholder. Both distributions are skewed to the right, which means that most credit card holders had a low credit limit relative to all cardholders. Here again, there is only a minor discrepancy between original and downsampled dataset.

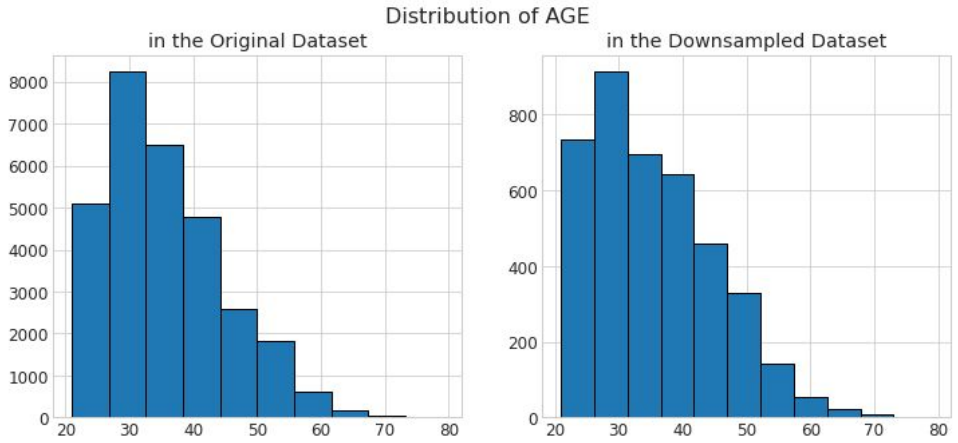
Image 4:



The distribution of the cardholders' age is displayed in Image 5. Although AGE is a bit more normally distributed than LIMIT_BAL, it is also right skewed in both the original and the downsampled dataset. A cursory observation confirms that roughly 90% of cardholders are between the age of 20 and 50 in both datasets.

In my in-depth analysis, I have checked more features, but these two illustrate the point. Therefore, I can conclude that the downsampled dataset reflects the original dataset well enough to be used for modelling.

Image 5:



3. Modelling and Optimizing Tree Models

The tree-based models that I have built and optimized to predict credit card default are a single decision tree and a random forest classifier. For the optimization of all models, I have created a stratified k-fold split (k=4) of the dataset to prevent overfitting and implemented grid-search cross validation for optimizing hyperparameters. A stratified split means that the 50:50 balance in my downsampled set will be retained in each training and testing subset. The scoring criterion for the grid-search optimization has been set to 'f1' to obtain the best balance of the classes after prediction.
For the decision tree model, the only hyperparameter used was:

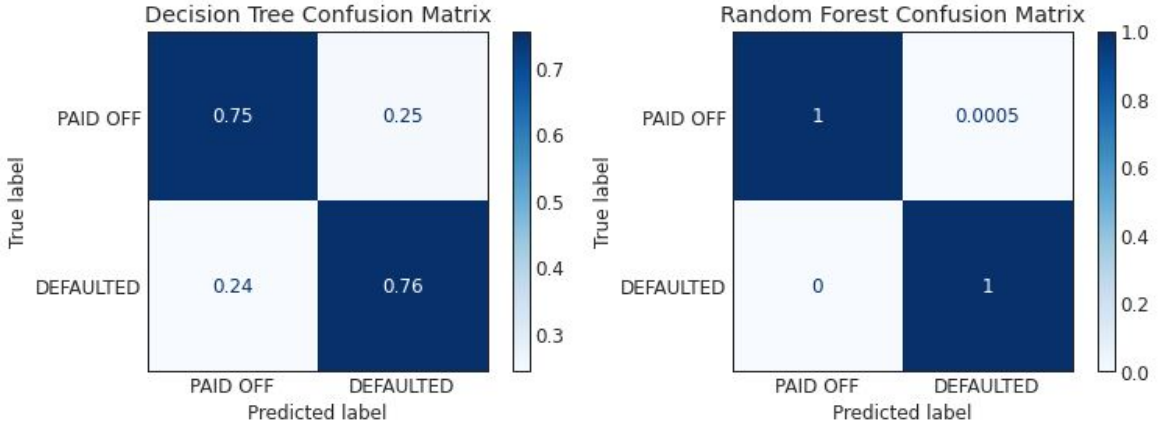
- cost complexity pruning (ccp_alpha) in range [0.0001, 0.001, 0.01, 0.1].

For the random forest model, the hyperparameters were:

- the number of estimators (n_estimators) in range [10, 50, 100, 200, 300],
- and the maximum number of features (max_features) in range [2, 3, 4, 5, 6, 'sqrt', 'log2'].

Image 6 shows the normalized confusion matrices for the optimized decision tree and random forest classifiers (for more information on model performance and best hyperparameters, see Table 2). While a single tree has a better-than-random-guessing performance, a forest of trees has much better classification performance.

Image 6:



4. Modelling and Optimizing 'Scaled' Models

Furthermore, I also built and optimized two 'scaled' models, i.e. a k-nearest neighbor and support vector machine classifier. I have called them scaled because they necessitate scaling of the data before modelling, which tree-based models do not. The downsampled dataset has been standard scaled as well as 4-fold stratified split for cross validation. For the k-nearest neighbor model, the hyperparameter used was:

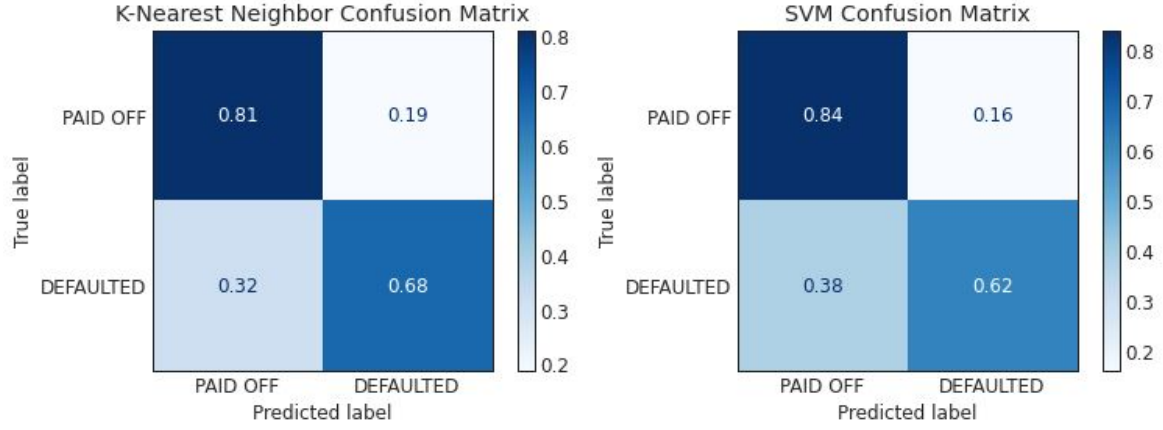
- the number of neighbors (n_neighbors) in range [3, 5, 7, 9, 11, 13, 15].

As the target is binary, I have only selected odd numbers that when a new instance is classified there is always an easy majority vote among the neighbors (i.e. without weighting).
For the support vector classifier, the hyperparameters used were:

- the regularization penalty (C) in range [0.001, 0.01, 0.1, 1, 10],
- and the kernel trick (kernel) out of ['linear', 'rbf'].

As Image 7 shows, the optimized k-nearest neighbor and support vector machine classifiers have similar classification performances despite their vastly different approaches.

Image 7:



5. Final Evaluation and Conclusion

The 4 models developed for this project of predicting credit card default have attained differing classification performance scores as summarized in Table 2. Usually, downsampling leads to a higher recall but a lower precision score as it gives more weight to the minority class. Downsampling both classes has alleviated this effect for the tree-based model, where the precision and recall are very close to one another. However, the effect has been inverted for the scaled models, which means they are still giving more weight to the majority class. This is why the F1-score is important when the balance of classes is an issue as it describes the harmonic mean of precision and recall.

Since the decision tree and k-nearest neighbors classifiers are relatively simple models, it is not unusual that they have attained lower F1-scores. However, it is surprising that the support vector machine performed worst on all scoring metrics compared to the other models. The best model for predicting both credit card default and pay-off across all metrics is the random forest model. It is also interesting to observe what balance the random forest has struck among its hyperparameters. While it needs only 2 features, but 300 estimators to fit them, this may suggest that random forest is still overfitting despite cross validation.

Nonetheless, given the clear advantage of random forest classifier, I would recommend using random forest to analyze the rest of the original dataset and build a model for production. For future analysis, it may also be helpful to have features on the amount of monthly income and the source of income (i.e. job) which each credit cardholder has. But, as I-Cheng and Che-hui wrote, this is probably missing because banks over-issued credit cards without fully checking the credibility of applicants. To prevent another credit card debt crisis, checking the credibility of applicants with machine learning is a crucial step, but in turn, the accuracy of these models should be continuously monitored too.

Table 2:

	Decision Tree	**Random Forest**	**K-Nearest Neighbor**	**Support Vector Machine**
Accuracy	0.7533	0.9998	0.7453	0.7298
Precision	0.7521	0.9995	0.7830	0.7936
Recall	0.7555	1.0000	0.6785	0.6210
F1-Score	0.7538	0.9998	0.7270	0.6968
Best Hyper-Parameters	ccp_alpha= 0.001	max_features= 2 n_estimators= 300	n_neighbors= 7 metric= minkowsky	C= 1.0 kernel= rbf