PREDICTING CREDIT CARD DEFAULT

A PREPRINT

Group Name: Group G
Department of Biochemical Engineering
University College London
London, WC1E 6BT

January 7, 2020

1 Introduction

The UCI Default of Credit Card Clients Dataset is a famous dataset that has been the subject of many machine learning papers. For our given task, we were trying to predict the 'default payment for next month', making this a binary classification problem. Our analysis starts by first transforming the data through means such as normalization and one hot encoding. Next, to get a better intuition for the dataset, we looked for linear relationships by generating a correlation heatmap between our input features and the output. It can be seen that variables such as PAY_LATE, LIMIT_BAL, PAY_DULY, PAY_AMT have high correlations while BILL_AMT has the weakest correlation with the output variable. Though simple, this analysis aided our feature selection process.

In section 3, we detail our methodology, which answers how we trained and evaluated our model. We chose to use under/oversampling to tackle the imbalanced dataset; Our metric of choice for this task was the f1 score for defaults; We created 2 new features - CASH_FLOW and PAY_PERC. The latter being a non-linear transformation of BILL_AMT; We detail Recursive Feature Elimination (RFE), a systematic way to select features. Lastly, we detail which features we chose for model training in the next section and why.

In section 4, we provide a brief explanation as to the models we have chosen. In section 5, we detail the training results of the different models. Using k-fold cross-validation, we find that logistic regression with undersampling provides the best F1 Score for defaults (the minority class), with a score of 0.525. We then performed RFE on this model, which allowed us to improve the F1 Score to 0.531, and allowed us to determine which were the most important features in the dataset for our given model.

In section 6, we ran our model on the test set, which gave us an overall accuracy of 0.79, and a F1 Score of 0.54 for defaults. We obtained slightly better metrics on the test set than on the training set.

2 Data Transformation & Exploration

2.1 Data Transformation

The credit card default dataset contains a mixture of categorical and numerical data with no empty entries. Light data cleaning is required, however, as some categorical features included classes which are undescribed in the documentation of the datasets. In such cases, these categories are mapped to the most appropriate class of the feature. For instance, as "MARRIAGE" is not defined for the values 0, 5, or 6, any data point labelled as such has its value changed to 4, which encodes the "others" class.

Prior to conducting preliminary data exploration, categorical features of the dataset first need to be encoded such that it could be processed in the same context as numerical ones. The approach we have chosen is One-Hot Encoding, which encodes the multinomial categorical features "MARRIAGE" and "EDUCATION" into three and four binary dummy variables respectively, each representing the different classes present for each feature. This encoding system, however, falls into the 'dummy variable trap' as the inclusion of every dummy variable causes a subset of features to have perfect multicollinearity, which lead to non-unique solutions for regression models. As such, it is common practice to omit one resulting dummy variable from the dataset [1].

Lastly, “PAY_x” features, which represents the payment status of a customer in a given month, is currently given as a multinomial categorical feature. In “PAY_x”, -1 indicates that a user has fully paid their bill, a 0 indicates that some minimum payment is met, and 1-8 indicate that payment is late by the given number of months. Unlike other aforementioned categorical features, “PAY_x” is distinct in that numerical data can be extracted from the latter classes if extracted from its original representation. Therefore, we have decided to transform “PAY_x” into three separate features:

- PAY_LATE_x : (binary) indicates if payment is late (1 if PAY_x >0, and 0 otherwise).
- PAY_LATE_MONTHS_x : (int) indicates the number of months late, if PAY_LATE_x = 1.
- PAY_DULY_x : (binary) indicates if payment is made fully (1 if PAY_x = -1, and 0 otherwise).

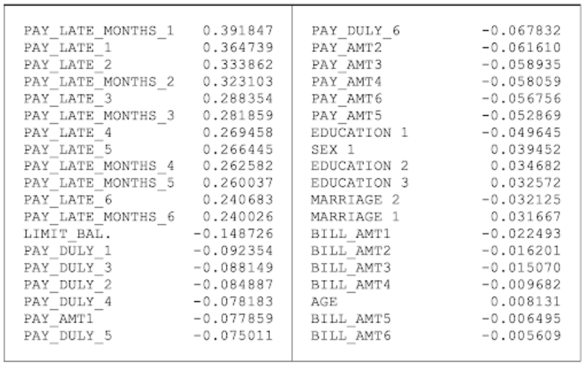
With this transformation, we have not lost any information from the original data format with the added flexibility to utilize the number of months late as a numerical feature separate from binary features of late and full payments. This will allow a more comprehensive exploration in the following section.

2.2 Data Exploration

The credit card default dataset presents a binary classification problem with imbalanced data, where the distribution of the binary output feature is heavily skewed to one class. Among the 24000 training data points, only 5370 (22.375%) are positives for default. This imbalance will be a major point of consideration in further data exploration, methodology, and result evaluation.

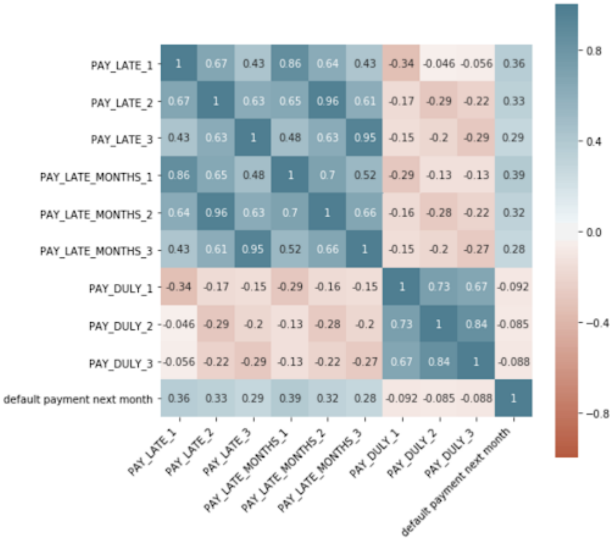
Due to the transformations described in the previous section, the new train and test dataset now contains 26 input features. The following explorations were conducted to evaluate the importance of these features.

2.2.1 *Correlation Analysis*



```
PAY_LATE_MONTHS_1   0.391847  PAY_DULY_6       -0.067832
PAY_LATE_1          0.364739  PAY_AMT2         -0.061610
PAY_LATE_2          0.333862  PAY_AMT3         -0.058935
PAY_LATE_MONTHS_2   0.323103  PAY_AMT4         -0.058059
PAY_LATE_3          0.288354  PAY_AMT6         -0.056756
PAY_LATE_MONTHS_3   0.281859  PAY_AMT5         -0.052869
PAY_LATE_4          0.269458  EDUCATION 1      -0.049645
PAY_LATE_5          0.266445  SEX 1             0.039452
PAY_LATE_MONTHS_4   0.262582  EDUCATION 2       0.034682
PAY_LATE_MONTHS_5   0.260037  EDUCATION 3       0.032572
PAY_LATE_6          0.240683  MARRIAGE 2       -0.032125
PAY_LATE_MONTHS_6   0.240026  MARRIAGE 1        0.031667
LIMIT_BAL.         -0.148726  BILL_AMT1        -0.022493
PAY_DULY_1         -0.092354  BILL_AMT2        -0.016201
PAY_DULY_3         -0.088149  BILL_AMT3        -0.015070
PAY_DULY_2         -0.084887  BILL_AMT4        -0.009682
PAY_DULY_4         -0.078183  AGE               0.008131
PAY_AMT1           -0.077859  BILL_AMT5        -0.006495
PAY_DULY_5         -0.075011  BILL_AMT6        -0.005609
```

Figure 1: Linear Correlation Analysis against "default payment next month



	PAY_LATE_1	PAY_LATE_2	PAY_LATE_3	PAY_LATE_MONTHS_1	PAY_LATE_MONTHS_2	PAY_LATE_MONTHS_3	PAY_DULY_1	PAY_DULY_2	PAY_DULY_3	default payment next month
PAY_LATE_1	1	0.67	0.43	0.86	0.64	0.43	-0.34	-0.046	-0.056	0.36
PAY_LATE_2	0.67	1	0.63	0.65	0.96	0.61	-0.17	-0.29	-0.22	0.33
PAY_LATE_3	0.43	0.63	1	0.48	0.63	0.95	-0.15	-0.2	-0.29	0.29
PAY_LATE_MONTHS_1	0.86	0.65	0.48	1	0.7	0.52	-0.29	-0.13	-0.13	0.39
PAY_LATE_MONTHS_2	0.64	0.96	0.63	0.7	1	0.66	-0.16	-0.28	-0.22	0.32
PAY_LATE_MONTHS_3	0.43	0.61	0.95	0.52	0.66	1	-0.15	-0.2	-0.27	0.28
PAY_DULY_1	-0.34	-0.17	-0.15	-0.29	-0.16	-0.15	1	0.73	0.67	-0.092
PAY_DULY_2	-0.046	-0.29	-0.2	-0.13	-0.28	-0.2	0.73	1	0.84	-0.085
PAY_DULY_3	-0.056	-0.22	-0.29	-0.13	-0.22	-0.27	0.67	0.84	1	-0.088
default payment next month	0.36	0.33	0.29	0.39	0.32	0.28	-0.092	-0.085	-0.088	1

Figure 2: Correlation Heatmap against "default payment next month

We conducted a basic correlation analysis to evaluate the linear significance of each input feature to the output feature. As shown on Figure 1, based on this metric, the lateness of previous payments seems to correlate significantly with defaults, more so than other features. This is followed by other financial features in LIMIT_BAL, PAY_DULY, and PAY_AMT. Generally, it seems that more recent payments are better predictors than older ones. Even with one-hot encoding, the given categorical features are still relatively poor predictors. Given that most financial features had among the highest correlation scores, BILL_AMT performed the worst out of all predictors. It is important to note, however, that correlation does not capture non-linear relationships between the input and output features. Hence, experimentation on engineering the BILL_AMT feature will be conducted in the following sections.

2.2.2 Dimensionality Reduction

Given the high dimensionality of the dataset, dimensionality reduction must be applied in order to visualise the distribution of defaults for preliminary exploration. Principal Component Analysis (PCA) was applied on continuous numerical features PAY_LATE_MONTHS, PAY_AMT, BILL_AMT, and LIMIT_BAL, where the first and second principal components were visualised with a scatter plot.

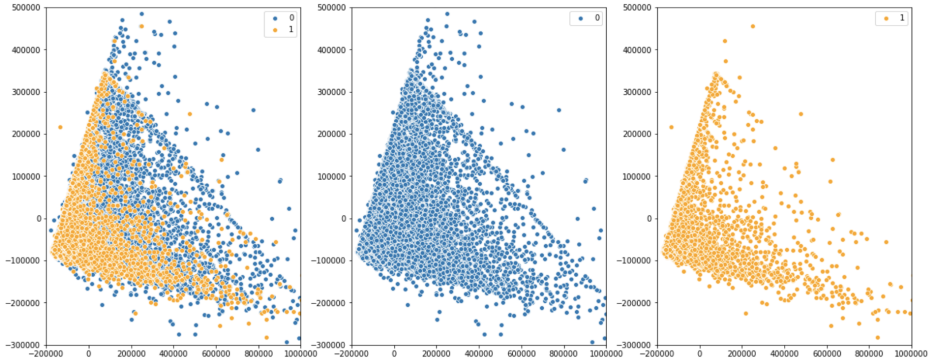


Figure 3: Preliminary exploration PCA (a = left, b = middle, c = right)

From Figure 3, defaults in the training dataset do not appear to be separable in lower dimensions. The contrast of data point distribution between Figure 3b and 3c suggests that while there is a large portion of non-defaulters that are distinguishable from defaulters, the reverse is not true. It may be reasonable to expect, therefore, that a model which captures a large portion of the defaulters (hence high recall) may not do so with high precision.

2.2.3 Relationship of Payment and Bill Columns

From the correlation analysis, we observe the importance of 'payment' columns, including our breakdowns of PAY[1-6] and PAY_AMT[1-6]. However, while it may be intuitively sound from a domain science perspective to assume that credit bill (BILL_AMT[1-6]) would have some sort of relation with the potential for default, such hypothesis is not evidenced by the correlation analysis. Hence, we investigate further on the relationship between these columns.

Month	**1**	**2**	**3**	**4**	**5**	**6**
PAY	-1	-1	-1	-1	0	-1
PAY_AMT	15586	344	2340	4702	339	330
BILL_AMT	1199	15586	344	2340	6702	339

Table 1: Payment columns of customer ID #145

It can be observed that the payment made by a customer in a given month x, corresponds to a bill that customer received in month x+1. In the case that the bill is fully covered for that month, the payment status for month x+1 will be labelled with '-1', otherwise, if payment is made that is less than the given bill, the payment status is marked as '0'. This is observed once in the above example, where PAY_AMT4 not covering BILL_AMT5 means that PAY5 is labelled as '0'. Hence, any feature engineering which attempts to relate PAY_AMT and BILL_AMT, which is conducted in this study, will have to take into account this one month offset.

3 Methodology Overview

3.1 Background Research

3.1.1 Data Scaling/Normalisation/Min-Max Scaler

Feature scaling (also known as data normalization) is the method used to standardize the range of features of data. We standardize data so as to eliminate the units of measurement for data, enabling us to more easily compare data from different places. There are many methods for data normalization including min-max normalization, z-score normalization and normalization by decimal scaling. We tried both min-max normalization and z-score normalization through the standard scaler function. The standard scaler standardise features by removing the mean and scaling to unit variance, where The standard score of a sample x is calculated as: z = (x - u) / s.

The min-max normalization performs a linear transformation on the original data. Suppose that mina and maxa are the minimum and the maximum values for attribute A. Min-max normalization maps a value v of A to v' in the range [new-mina , new-maxa] by computing: v'= ((v-mina) / (maxa – mina)) * (new-maxa – newmina) + new-mina. Standardization & normalization of these methods were tested on the models [2].

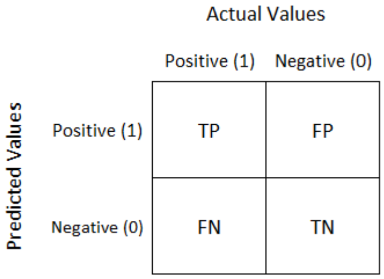
3.1.2 Under/Oversampling

Under/Oversampling are popular methods to deal with imbalanced datasets. The rationale is that when data is unbalanced, standard machine learning algorithms that maximise overall accuracy tend to classify all observations as majority class instances [3]. This then leads to low recall on the minority class. For our dataset, we'll use random undersampling and oversampling (specifically SMOTE) as our sampling techniques. These techniques have the effect of modifying the decision function of the classifier, as can be seen here [4]. That being said, there is yet a theoretical framework that explains how sampling techniques affect the overall learning process [3], and whether the modification in the decision boundary is a desired one. To tackle this issue, we'll compare our classifiers trained on the on undersampled, oversampled and original data.

3.1.3 Evaluation Metrics Choice

Taking into account the goal of the project, it is natural to assume that the output of the model can be categorised into True/False Positives (default) and True/False Negatives (no-default). The highlight of the prediction activity would be to decrease *real-loss* arising from the customers defaulting (false negatives) when predicted no-default and also *opportunity-loss* by considering false positives (FP) to potentially default while they aren't going to. This led to the choice of representation by **recall** and **precision** rather than accuracy as real-loss and opportunity-loss are the first and second most misclassification costs in credit card defaults respectively.

F1 Scores arises as it is very often used to present machine learning algorithm results for binary decision problems. Log-loss is also an interesting approach, giving more weight into accuracy, hence in our case is less prioritised. F1 Score will be used not only because it provides a simple one-parametric value in evaluation steps, but also value precision and recall in its metric evaluation evenly. More information will be needed in terms of penalties and misclassification costs to be able to determine a more accurate evaluation metric.



		Actual Values	
		Positive (1)	Negative (0)
Predicted Values	Positive (1)	TP	FP
	Negative (0)	FN	TN

Figure 4: Confusion Matrix for Binary Classification

$$F_1 = \left(\frac{2}{\text{recall}^{-1} + \text{precision}^{-1}}\right) = 2 \cdot \frac{\text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}}$$

3.2 Feature Engineering

Feature Engineering was used to highlight or supplement existing data, in order to highlight relationships in the dataset, as well as use extract meaningful information from existing data.

Based on domain knowledge [5] and intuition, we engineered 2 new groups of attributes from existing data to highlight the relationships between PAY_AMNT and BILL_AMT due to the high correlation between these variables. Based on domain knowledge on credit cards and loans, we believe BILL_AMT to be a plausible indicator for default. However, due to the low correlation between BILL_AMT and default, we wanted to engineer a feature which would make BILL_AMT useful to our model. Using these 2 variables, we formed CASH_FLOW and PAY_PERC. CASH_FLOW details the difference between the bill amount and the amount paid by the user - intuitively, the higher this difference is on average, the more likely he is to default on his payment. The second is PAY_PERC which details the percentage of the bill which was paid in a month. PAY_PERC is therefore a non-linear transformation in relation to BILL_AMT. Using the 2 new features, we wanted to conduct analysis on whether a linear or non-linear combination was more appropriate to model the interaction between PAY and BILL_AMT. Based on initial correlation analysis using the 2 engineered features, we found that PAY_PERC has a higher correlation with default than CASH_FLOW. We hypothesize that PAY_PERC will perform better than CASH_FLOW as an indicator for default. We then conducted feature analysis and selection to validate the selection of these attributes. Through correlation analysis of the engineered features, we found that log(PAY_PERC) presents a significant increase in correlation with default than just PAY_PERC. Intuitively, this measures the difference in magnitude between what a customer owes the bank and what he was willing and able to pay. Thus, we decided to use log(PAY_AMT/BILL_AMT) as our non-linear combination of the 2 features, labeled as PAY_PERC. The full correlation ranking of these engineered features are included in section 4 of the attached notebook.

3.3 Feature Selection

Based on the engineered and transformed data, we conducted a few preliminary tests to determine feature importance and aid feature selection.

The basic idea of feature selection is to remove features which add noise rather than accuracy to the model to reduce overfitting in the final model, and add features which might increase the strength of the model. The inclusion of unimportant features might result in a model which wrongly takes these features into account during training, but does not actually scale to other datasets. This results in the false inclusion of noisy data in the dataset. Thus, the reduction of features can improve the strength of the model. Our goal is thus to find and eliminate these features from the dataset. Reduction of features also means that less data is needed to re-train the model, which makes maintaining it less expensive. [6]

3.3.1 Coefficient Analysis and Feature Importance

Coefficient analysis and feature importance is one method to find the most important features in a dataset. By analysing the coefficients assigned by various algorithms, we can get an intuition for which features carry the most weight, or have the highest impact on the model. This is applicable to regression algorithms and random forests, where the weight (or coefficient) of each variable can be found. Using the weights assigned by each model, we were able to create a ranking of each feature's importance for each model we used, thus giving us candidate features for elimination.

3.3.2 Recursive Feature Elimination

Recursive Feature Elimination (RFE) is a feature selection process which builds on the idea of coefficient analysis and feature importance. Based on a model, it iteratively removes features with the lowest weights and tests for the accuracy of a resultant model without these low-weightage features. It then uses these accuracy scores to produce a set 'best' variables with k number of features, where k is a user-defined number, which optimises for accuracy on the selected model. Using this process, we are able to use the information obtained from feature importance and coefficients to quickly find features that reduce the accuracy of each model, allowing us to eliminate poor features in an efficient manner, as well as to identify features which are consistently important, allowing us to include these features in our model. Using RFE, we wanted to find the optimal number of selected features which would result in the highest F1 Score.

3.4 Final Methodology

We have settled on a two-step methodology, using cross-validation to conduct both model selection and feature selection before performing predictions on the test set.

3.4.1 Model Selection

Preliminary trials have been done on various models, in which we observe that using every input feature in training causes some models to perform extremely poorly. Hence, to adjust for this issue, models are trained only a subset of features selected based on the correlation analysis done in section 2. These features are PAY_LATE_MONTHS[1-6], PAY_DULY[1-6], 'LIMIT_BAL', and PAY_PERC[1-5]. The latter are our previously described engineered feature which achieved higher correlation than both PAY_AMT[1-6] and BILL_AMT[1-6]. We will then compare the performance of five models, each of which are described in the following section. Each model will be trained with three approaches – using the processed data as described in the data transformation section, using random undersampling, and using SMOTE.

3.4.2 Feature Selection

Recursive Feature Elimination presents a more systematic approach to eliminate noisy features compared to a basic correlation analysis. Once we have settled on a model and a sampling method, we will use RFE on the selected model to determine features which act as the best predictor of defaults. Given that RFE requires the number of selected features to be specified, we will hence also find the optimal number of features by iteratively applying RFE using 1 to 42 selected features, the latter being equivalent to using every input feature. We will then perform cross validation using features selected by each iteration of RFE, and choose the most optimal number of selected features again based on the F1 Score of '1' classification for 'default payment next month'. Features which provides the best results will be selected to perform the final prediction on the test dataset.

4 Model Training & Validation

To compare the performance of different approaches, models are evaluated using a 10-fold cross validation on the training set. For the purpose of fair comparison, the training dataset was first shuffled randomly, however the same shuffle random state is used for all cross validations in this study. Precision, recall, accuracy, and classification count is computed for each cross validation fold and is averaged across 10 folds, these will act as the main evaluation metrics for each approach.

The 5 models we chose were logistic regression, KNN, Naive Bayes, Random Forest and QDA. We included logistic regression because it is often the 'go-to' model in classification due to its simplicity. Naive Bayes was then included because it's the generative pair to logistic regression. "Random Forest and KNN were included as past literature [11] has shown that these models perform well on this dataset.. QDA was included because it allows us to model a non-linear discriminant boundary without too much loss of interpretability.

Logistic Regression

Logistic Regression models the probabilities for classification problems with two possible outcomes. Instead of fitting a straight line or hyperplane, the logistic regression model uses the logistic function to squeeze the output of a linear equation between 0 and 1. The logistic function is defined as:

$$\text{logistic}(\eta) = \frac{1}{1 + exp(-\eta)}$$

KNN

The basic idea of k-Nearest Neighbors (KNN) is to determine the category of a given customer based on the categories of the k customers that are nearest to it (by euclidean distance) in the feature space [7].

Naive Bayes

One approach to classification is to assign to a given customer d the class c = arg maxc P(c | d). We derive the Naive Bayes (NB) classifier by first observing that by Bayes' rule,

$$P(c \mid d) = \frac{P(c)P(d \mid c)}{P(d)},$$

where P(d) plays no role in selecting c$*$. To estimate the term P(d | c), Naive Bayes decomposes it by assuming the f_i's are conditionally independent given d's class: [8].

$$P_{\text{NB}}(c \mid d) := \frac{P(c) \left(\prod_{i=1}^{m} P(f_i \mid c)^{n_i(d)} \right)}{P(d)}.$$

Random Forest

The random forest classifier consists of a combination of tree classifiers via bootstrap re-sampling, where each classifier is generated using a random vector sampled independently from the input vector, and each tree casts a unit vote for the most popular class to classify an input vector [9]. Random Forest is able to detect non-linear behaviors creating different trees to adjust itself to different trends of resource exhaustion.

Quadratic Discriminant Analysis

Quadratic discriminant analysis (QDA) is widely used parametric methods that assume that class distributions are multivariate Gaussians. The theory also assumes knowledge of population parameters (means, covariance and priors for every class). If this information is not available, maximum-likelihood estimates can be used, although in this case the Bayesian optimality properties are no longer valid [10].

5 Results

5.1 Model Selection

As described in the methodology, our main measure model performance evaluation is the F1 Score of defaults. For the purpose of succinctness in data presentation, each model is only shown with its best performing sampling method.

Model and Sampling	Class	Count	Accuracy	Precision	Recall	F1 Score
Logistic Regression	non-defaulters	1795	0.7742	0.8679	0.8363	0.8518
(undersampling)	defaulters	605		0.4958	0.5585	0.5251
Gaussian Naive Bayes	non-defaulters	1775	0.7604	0.8627	0.8221	0.8419
(SMOTE)	defaulters	625		0.4699	0.5461	0.5049
K-Nearest Neighbors	non-defaulters	1543	0.6920	0.8642	0.7157	0.7829
(Undersampling)	defaulters	857		0.3820	0.6098	0.4695
Quadratic Discriminant Analysis	non-defaulters	1729	0.7531	0.8672	0.8051	0.8350
(no sampling)	defaulters	671		0.4586	0.5726	0.5091
Random Forest Classifier	non-defaulters	1572	0.7023	0.8653	0.7301	0.7919
(undersampling)	defaulters	828		0.3927	0.6057	0.4763

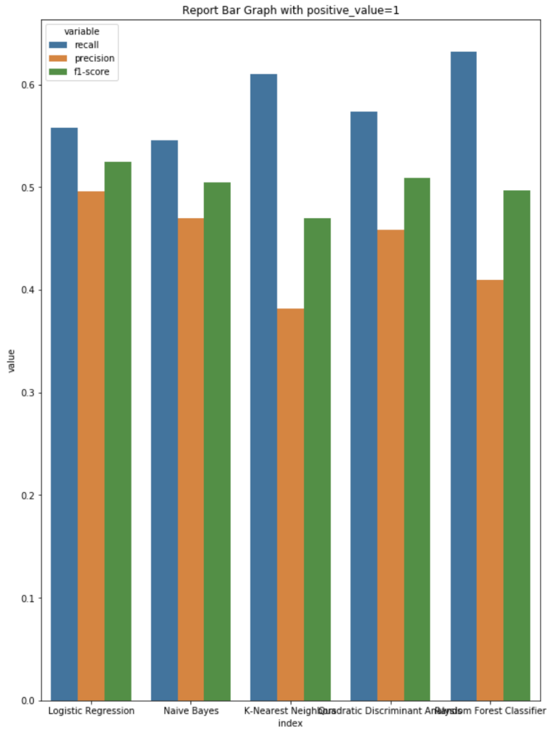


Figure 5: Model Comparisons for Standard Metrics and F Score

It can be seen that for most models, either undersampling or SMOTE resulted in the best F1 Score for defaults, with the exception of QDA. Similar to many other empirical studies [3], under/oversampling has allowed us to accurately generate more predictions for the minority class. The improvement for undersampling can also be motivated from a more theoretical perspective. In an unbalanced problem, it is often realistic to assume that many observations of the majority class are redundant, and that by removing some of them at random, the data distribution will not change significantly [3]. After undersampling, our F1 Score for the majority class decreased slightly, which means that while some relevant data were removed, a sizable portion of it was redundant. The decrease is not significant, and the corresponding increase in f1 for defaults more beneficial and far outweighs it.

Comparing the outcome of the different models with its respective best performing sampling method, most models performed within close margins. While the models presented in Figure 3 are narrowed down on the highest F1 Scores for defaulters alone, Logistic Regression managed to perform best across all metrics – on F1 Scores for both defaulters and non-defaulters, as well as overall model accuracy. This, however, is not the case when omitted results are considered, and hence it cannot be concluded that Logistic Regression is a definitive model for this particular classification problem.

For instance, Random Forest Classifier without undersampling gave better performance for non-defaulter F1 Score (0.8758) and accuracy (0.7954), however at the cost of substantially lower defaulter F1 Score (0.4200). This difference in sampling approach and evaluation explains why previous studies such as Ajat et.al [11] considered Random Forests as one of their best performing models, as their main evaluation metrics were the recall and precision of non-defaulters. Even within this study, KNN and Random Forests performed best considering only the recall of defaulters. Hence, these models would be best if the study is more concerned with identifying as many true defaulters as possible at the cost of precision. Overall, differing motivations would lend themselves to different most appropriate models, for the purpose of this study, we will proceed with Logistic Regression with random undersampling.

5.2 RFE Feature Selection

Recursive Feature Elimination was then performed iteratively in different number of features using Logistic Regression with random undersampling. Cross-validation results are presented in section 5.1.

	support	precision	recall	F1 Score
Non-defaulters	1782	0.87	0.83	0.85
Defaulters	613	0.49	0.57	0.53
Macro avg	2400	0.78	0.70	0.69
Weighted avg	2400	0.78	0.77	0.78
Accuracy		0.77		

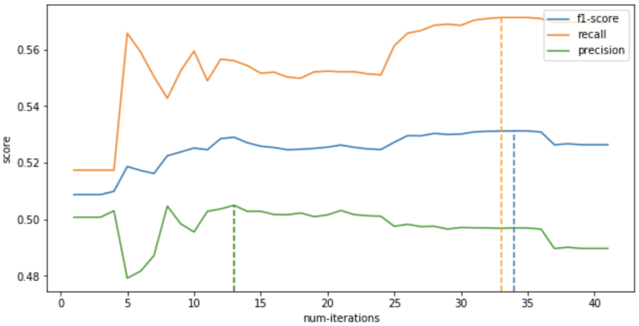


Figure 6: Cross-validation result of RFE with 34 features (Left) and Classification scores varying with number of RFE features (Right)

The highest performance was achieved using 34 features, in which we obtain a F1 Score of 0.531 for defaulters, an improvement to the 0.525 we obtained by naively selecting features based on correlation scores.. The highest rank features based on RFE are the following:

['AGE', 'BILL_AMT1', 'BILL_AMT3', 'PAY_AMT1', 'PAY_AMT2', 'PAY_AMT3', 'PAY_AMT4', 'PAY_AMT5', 'PAY_AMT6', 'SEX 1', 'EDUCATION 1', 'EDUCATION 2', 'EDUCATION 3', 'MARRIAGE 1', 'MARRIAGE 2', 'PAY_LATE_MONTHS_1', 'PAY_DULY_1', 'PAY_LATE_MONTHS_2', 'PAY_DULY_2', 'PAY_LATE_MONTHS_3', 'PAY_DULY_3', 'PAY_LATE_MONTHS_4', 'PAY_DULY_4', 'PAY_LATE_MONTHS_5', 'PAY_DULY_5', 'PAY_LATE_MONTHS_6', 'PAY_DULY_6', 'PAY_PERC_1', 'PAY_PERC_2', 'PAY_PERC_3', 'PAY_PERC_4', 'PAY_PERC_5', 'CASH_FLOW_1', 'CASH_FLOW_2']

Features which are omitted in this prediction are the remainder of 'BILL_AMT[2,4,5,6]', 'CASH_FLOW[3,4,5], and despite its earlier high correlation score, 'LIMIT_BAL'. It was expected as per previous discussions that BILL_AMT would not perform well as a predictor in its current state, as we saw low linear correlation between it and defaulters in section 2 of this report. Naturally then, it is expected that our engineered feature CASH_FLOW, being a linear function on BILL_AMT (and PAY_AMT), would also perform poorly.

However, we have demonstrated that our other engineered feature, 'PAY_PERC[1-6]' were among the higher ranked feature based on this RFE analysis, outperforming both its constituent features PAY_AMT and BILL_AMT (the RFE ranking of these features is presented in section 6 of the attached Jupyter Notebook). It seems evident, then, while performing poorly in its original form, BILL_AMT can potentially provide good predictive performance for defaulters if it is processed non-linearly.

6 Final Predictions on Test Set

Based on the results obtained, the final prediction using Logistic Regression with random undersampling, using features listed in section 5.2. The following result was obtained:

	support	precision	recall	F1 Score
Non-defaulters	4734	0.88	0.85	0.87
Defaulters	1266	0.51	0.57	0.54
Macro avg	6000	0.69	0.71	0.70
Weighted avg	6000	0.80	0.79	0.80
Accuracy		0.79		

	Predicted non-defaulters	Predicted defaulters
True non-defaulters	4035	699
True defaulters	544	722

We observe that scores obtained on the test set prediction are slightly higher than expected compared to results achieved in the model validation phase in the previous section. In this study, we have considered multiple models and features before landing on this final approach, with the goal of maximising both the precision and recall (hence, F1 Score) of defaulting customers. By motivating our approaches to maximise these metrics of success, we landed on a model that would most effectively discern potential defaulters, while also not compromising much on the precision and recall of non-defaulters, as well as overall accuracy.

The evaluation metric we have selected for this final prediction is evidently a challenging metric to maximise for. As discussed in our data exploration section, the resulting PCA of the train dataset in Figure 3 presents a scatterplot where defaulters seem indiscernible from non-defaulters (at least in two dimensions). Hence, we hypothesized that a model which captures a large portion of the defaulters may not do so with high precision. This was evidenced to be the case in this investigation, as throughout the study higher recall models comes with the price of lower precision. To have further insight into the performance of our final prediction, therefore, we conduct on the test dataset the same PCA analysis we used to explore the train dataset.

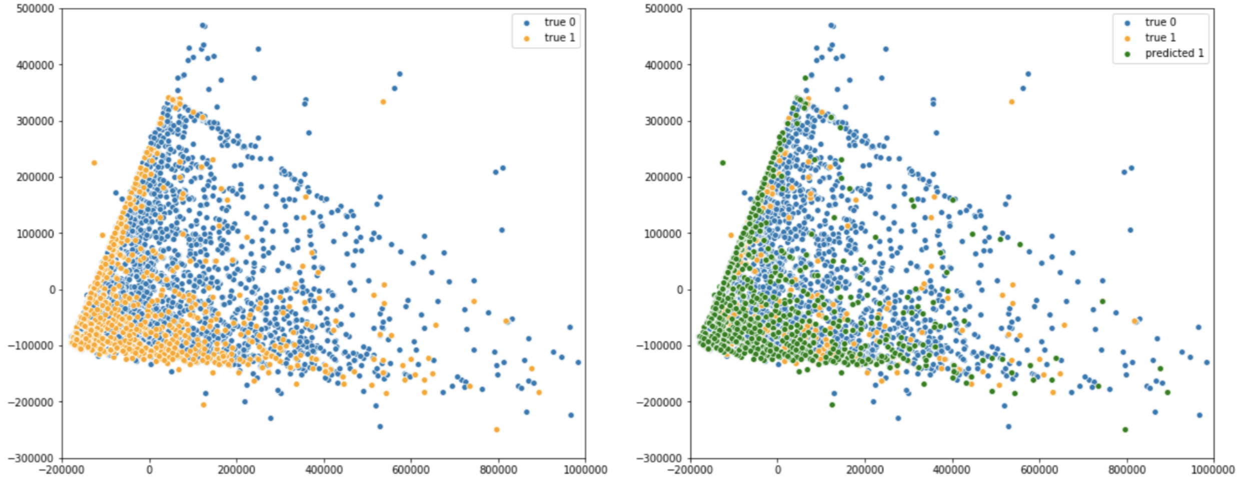


Figure 7: PCA on Test Dataset

It would seem that our final prediction model well captures the nature of most defaulters, however it undoubtedly did so while making false positives, misclassifying non-defaulters which also adopt similar characteristics. We can also observe that while the majority of 'clustered' defaulters are correctly identified, there are a number of defaulters which are scattered amongst clusters of non-defaulters this PCA scatterplot which went unidentified. Therefore, many of our false negatives arise from consumers which exhibit outlier characteristics. We would come to expect, that our model, and possibly most others, will not be able to identify most of these cases.

7 Conclusion

Our goal during this project was to provide a systematic analysis of the UCI dataset and provide a classifier that was robust at predicting defaults and non-defaults. The systematic analysis was done through the correlation heatmap and principal component analysis (PCA). The former allowed us to capture linear relationships which were important for the modelling work later on. From PCA we can observe that lots of data points that correspond to defaults look similar to non-default data.

Through feature engineering and correlation analysis and some trial and error, we were able to identify a subset of important features to be used in model training, namely PAY_LATE_MONTHS[1-6], PAY_DULY[1-6], 'LIMIT_BAL', and PAY_PERC[1-5]. It is important to note that our models performed worse when we simply included all features for training, meaning some of them were irrelevant and just added to noise. We were also successful in our use of sampling methods to increase the F1 Score of our models. Once a final model was selected (logistic regression), we used Recursive Feature Elimination as a more systematic approach to select features. This allowed us to pick a subset of 34 features while improving our F1 Score. It also allowed us to observe that our naive feature selection was quite good, in that the features we selected were generally in concordance with those selected by RFE. Furthermore, RFE validated the fact that our engineered feature PAY_PERC was a useful one. Using this subset of features, we were able to obtain an F1 Score of 0.87 and 0.54 on non-defaults and defaults, respectively. This score is marginally higher than those obtained during k-fold cross validation.

However, our F1 Score for defaults is still relatively low, in that it can't practically be used in real life to predict customer defaults. This partly stems from the nature of the data, in that many default cases look similar to non-default cases. It is also because we have comparatively fewer defaults compared to non-defaults. If we had more data for defaults, we might be able to create a better classifier.

Our low F1 Scores could also stem from our model choice. Our final model, out of the 5 we rigorously tested, was logistic regression. While interpretable and simple, logistic regression is limited method as it is still only a linear classifier. In the future, we could explore classifiers such as ANNs and SVMs as these models are better at capturing non-linear relationships.

References

[1] D. Suits, "Use of Dummy Variables in Regression Equations", *Journal of the American Statistical Association*, vol. 52, no. 280, pp. 548-551, 1957. Available: 10.1080/01621459.1957.10501412

[2] L. Shalabi, Z. Shaaban and B. Kasasbeh, "Data Mining: A Preprocessing Engine", Journal of Computer Science, vol. 2, no. 9, pp. 735-739, 2006. Available: 10.3844/jcssp.2006.735.739

[3] A. Dal Pozzolo, O. Caelen and G. Bontempi, "When is Undersampling Effective in Unbalanced Classification Tasks?", *Machine Learning and Knowledge Discovery in Databases*, pp. 200-215, 2015.

[4] "Under-sampling — imbalanced-learn 0.5.0 documentation", Imbalanced-learn.readthedocs.io. [Online]. Available: https://imbalanced-learn.readthedocs.io/en/stable/under_sampling.html.

[5] Y. Ma, "Prediction of Default Probability of Credit-Card Bills", Open Journal of Business and Management, vol. 08, no. 01, pp. 231-244, 2020. Available: 10.4236/ojbm.2020.81014

[6] J. Reunanen. Overfitting in making comparisons between variable selection methods. *Journal of Machine Learning Research*, 3:1371–1382, 2003.

[7] V. Bijalwan1, Machine learning approach for text and document mining, *arXiv*, 2004

[8] B. Pang, L. Lee and S. Vaithyanathan, "Thumbs up?", *Proceedings of the ACL-02 conference on Empirical methods in natural language processing - EMNLP '02*, 2002. Available: 10.3115/1118693.1118704

[9] L. Breiman, *Machine Learning*, vol. 45, no. 1, pp. 5-32, 2001. Available: 10.1023/a:1010933404324

[10] J. Alonso, L. Belanche and D. Avresky, "Predicting Software Anomalies Using Machine Learning Techniques", *2011 IEEE 10th International Symposium on Network Computing and Applications*, 2011

[11] Ajay, V. Ajay, S. Jacob, "Prediction of Credit-Card Defaulters: A Comparative Study on Performance of Classifiers" *International Journal of Computer Applications*, vol. 145 – no.7, 2016