PREDICTING CREDIT CARD DEFAULT

A PREPRINT

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

January 11, 2022

1 Introduction

The globally worsening pandemic has seriously increased the debt generated by credit card users, with the Bank of England declaring the highest debt observed in more than a year. As a result, it is no surprise that the need for an accurate predictive model that judges the credibility of a potential credit card client has become increasingly imperative to banks. Such circumstances demand the construction of creative modelling techniques, and this report outlines such methods and their successes on the largely explored customer default payments dataset from Taiwan in 2006.

The dataset is split into 24000 training and 6000 test data points that observe the following habits of each client and track with their actual outcome of defaulting/not defaulting:

- Amount of credit
- Gender
- Education
- Marital status
- Age
- Payment history
- Bill statements
- Previous payment amounts

The data is used to form 16 model combinations of one feature engineering technique and one sampling technique each, followed by hyperparameter tuning. The feature engineering methods (LASSO, XGBoost and manual transformation) are used to select the categories that have the most significance to the observed outcome. The sampling methods used in conjunction are SMOTE, TL, and SMOTE-TL. The cases of no feature engineering and no sampling techniques are explored as well.

To judge the predictive accuracy of each combination, the KNN, QDA, LR, LDA, DF, XGB, and NB classification methods are used, and they output their preferred models. The best individual predictions on the training data were given by QDA. The final results on the test data are created using a soft-voting technique that probabilistically determines whether a client defaults based on the three top models.

2 Data Transformation and Exploration

2.1 Data Exploration

Since we are aiming at predicting whether a person will default in payment next month or not, the EDA helps us understand what the correlation between all the attributes are, and what are the most relevant attributes to perform the prediction. We have downloaded the full dataset from the UCI website for basic attribute analysis.

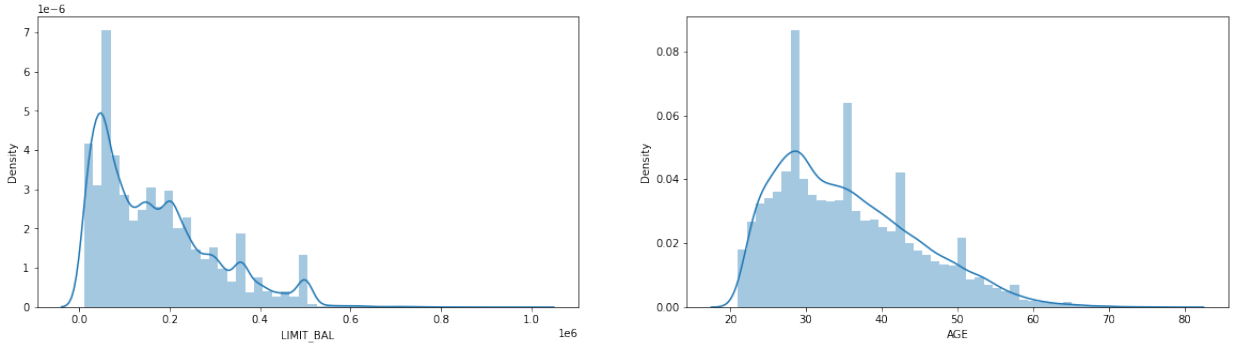

Figure 1: Skewness of the Continuous Variables

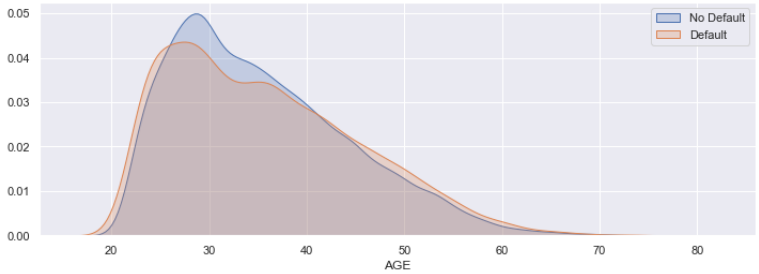

Figure 2: Distribution of Default Payment in Age Attribute

To provide an overview of the data, the following plot shows that approximately 23 % of the clients are expected to default the payment for next month.

We can firstly understand the basic profile of the clients from checking the skewness of the continuous variables, limiting balance (X1 – Amount of the giving credit) and age (X5).

We can observe that the limit balance is more skewed between 0 to 200,000, and clients age are more skewed from 20 to 40 years old.

As we can observe, as the age increases, higher proportion of clients in the age group will default the payment.

Observing the data over the last six months, the above plot shows us the proportion of clients that will default payment next month based on repayment history. For Current month status, the earlier the payment is made lesser are the chances of those clients defaulting the payment.

These two plots show us that first, married people between age bracket of 30 and 50 and unmarried clients of age 20-30 tend to default payment, and second, woman in the age group of 20-30 has the highest tendency to default the payment compared to man in all age groups.

In Figure 3 it is observed that there is higher proportion of clients for whom the bill amount is high but payment done against the same is very low.

The full correlation matrix (not shown) revealed several linear relationships suggesting that the data set could benefit from feature engineering. The 'PAY' variables are particularly correlated, as shown in Figure 4.

2.2 Data Transformation

In general, the graphs generated from the data of all features is skewed towards one side and has a significantly long trail. For example, the limit balance is significantly in the range of 0 and 200,000 and client ages predominantly lie between 20 and 40 years of age. The lack of data towards the tail end gives us statistically less confidence in deciding the client's tendency to default in that range. For this reason, where possible, the data in a feature can be regrouped such that each group in the feature would have roughly the same size, and thereby a provide more equal ground of default prediction. This is also true for the features 'Education' and 'Marriage' which can be summarized into Boolean groups e.g. "Married" and "Other."

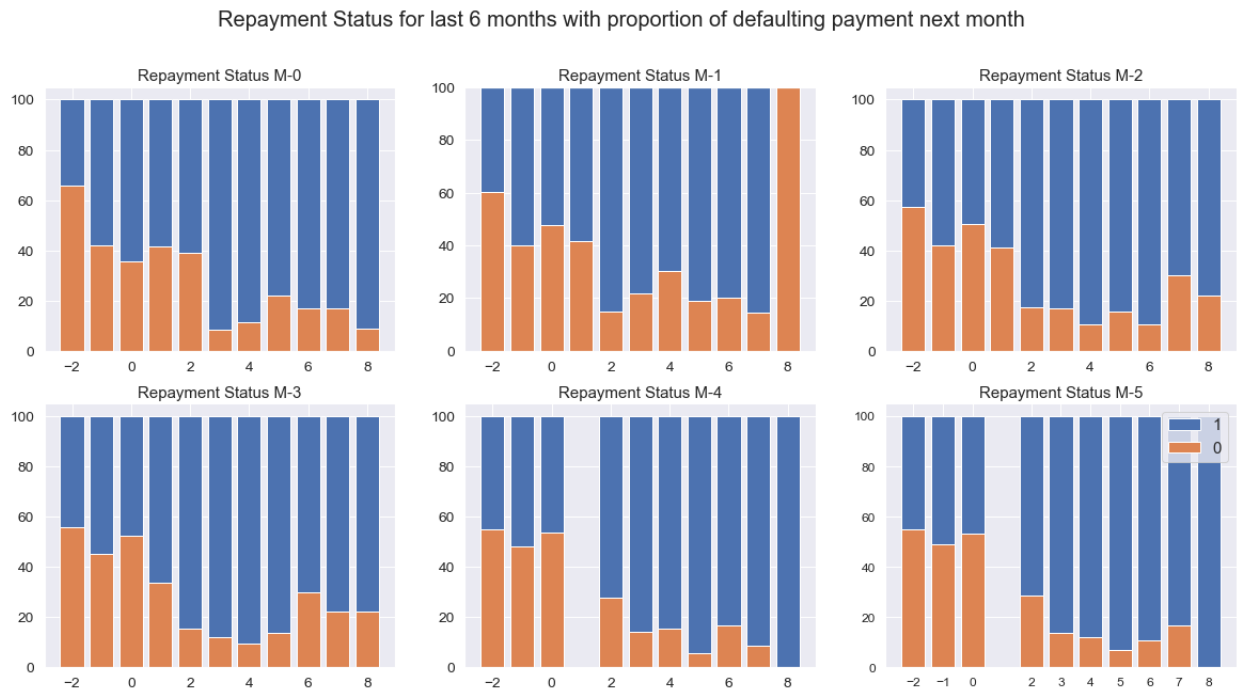

Figure 3: Repayment Statues for last 6 months with proportion of defaulting payment next month

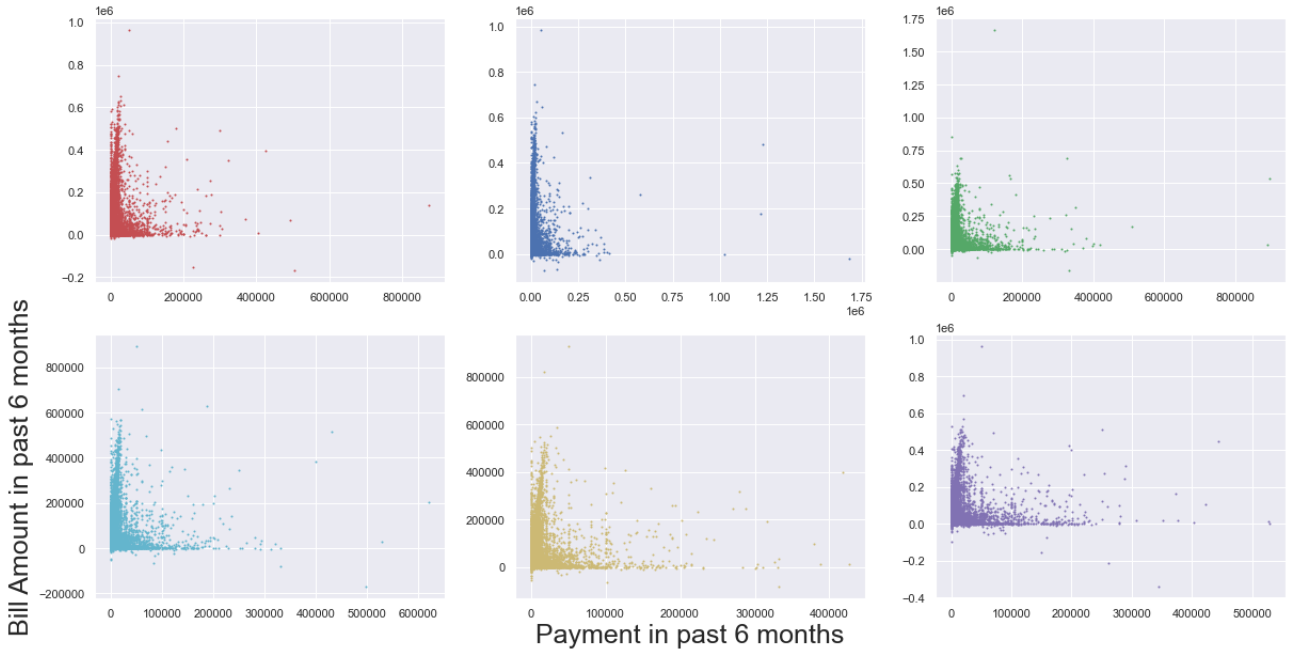

Figure 4: Bill vs. Payment Analysis in the Past 6 Months

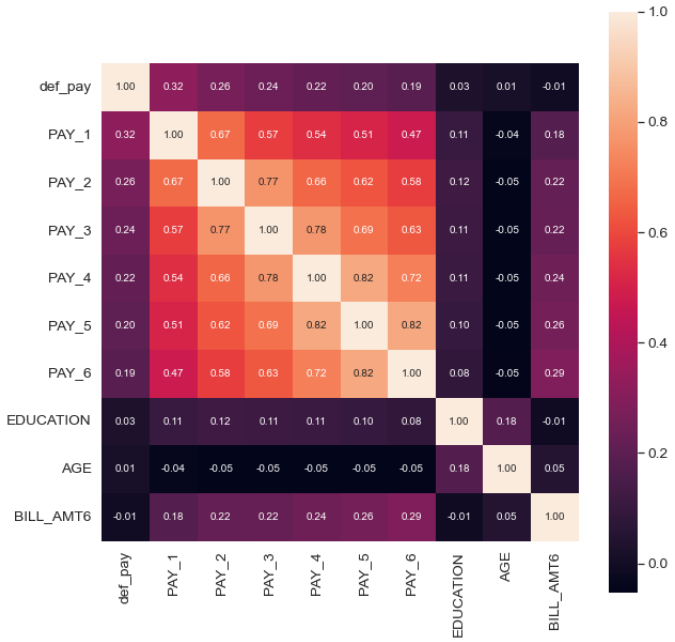

Figure 5: Payment Correlation

In other cases, one feature alone may not be sufficient in providing a clear tendency (or lack of) to default; instead, two or more features combined provide a more obvious pattern of defaulting. One such case is that married clients in their 30s to 50s and unmarried clients of 20s and 30s pose a notably higher probability to default payment . The correlation heat map exhibits the feature interdependency clearly; and this is apparent across the bill amounts across months. These are typically consistent, exhibiting similarities of at least 80% across months. The correlations of the bills are also greatly consistent with the credit limit the clients have (28-30%). For this reason, not only can the client's bills over six months be averaged into one feature without great loss of generality, but this new averaged feature can also coalesced with the limit balance. However, this extent of correlation is not true for all variables, e.g. the payment amounts across six months are mostly correlated less 80%, the lowest being under 50%, making it inappropriate to merge these features.

Finally, any remaining features that cannot be regrouped or merged to create a better-performing, positively-correlated feature to defaulting can be negated – this is true for the client's sex and average returns.

Although the intuitive transformation of data statistically improved the correlations to default, the original data is able to produce a rawer result based on the true and complete values. For this reason, only one feature engineering method uses the manual transformation data described above, while the other two (XGBoost and LASSO) work with the given data.

3 Methodology

3.1 Background Reading

The very real problem of credit card default has prompted dozens of journal and internet articles, many using the very same UCI Taiwanese credit card default data set.

A standard approach typically contains the following elements: data visualisation, sampling, feature engineering, model validation, and finally model prediction (Jenny Wang article). In general, feature engineering is useful for avoiding the 'curse of multi-collinearity' and high dimensionality. Wang uses an interesting approach of featured engineering, grouping correlated features in a strategic way; for example, the "age" attribute is used to calculate which generation the client belongs to, i.e., "Generation X", "Millennials", etc.

Over and/or undersampling also tend to improve classifier performance on imbalanced datasets by generating more instances of the under-represented class, or by removing instances of the over-represented class, respectively. There is no clear ideal sampling method as it varies with a number of factors, including model choice. For example, Alam et al. report that oversampling consistently out-performs for their gradient boosted decision tree (GBDT) model, however Mqadi et al. make a case for undersampling, including the fact that it does not 'clone sensitive financial data.' The

same is true for feature engineering – certain models, such as random forest, can be used to regularize the feature space (Dominquez, 2021) and this data can be used with other models, not just random forest. However, there are drawbacks to sparsity-inducing regularisers such as LASSO as they cannot be used in conjunction with kernel methods (Hosseini, 2021).

Likewise, the metric used for 'success' varies. Accuracy is widely acknowledged as insufficient for imbalanced datasets though it is still sometimes the main metric reported. Recall tends to be preferred but this is balanced with metrics such as the F-score or Area Under the Curve (AUC) to ensure a non-lopsided performance (Alam et al. 2020).

In terms of classifiers, a typical methodology trains, validates and examines several before settling on a final predictor; Wang (2021) examines Logistic, K-Nearest Neighbor, Random Forest, Extra Trees, and Naive Bayes and finds Random Forest to be the best within the context of their method. Beyond the 'standard' classification models, several authors implement augmented models

Hybrid classifiers are also presented in the literature; Alam et al. combine gradient boosting with the decision tree framework and report performance superior to those in Table 1, at least in terms of accuracy.

Deep learning is the subject of much attention in the machine learning community, and its application to this classification task is no surprise. However, deep leaning should not be used for the sake of it. Hsu et al. examine neural networks and deep neural networks for a variety of activation functions and compare with ensemble-learning methods, including boosting. They find that boosting tends to out-perform the neural network approach and suggest the difficulty of hyperparameter tuning as well as the 'relatively small sample.'

Finally, voting systems are sometimes used to create external ensemble models. A recent paper (Dávid et al. 2021) finds that there is no clear 'best' voting method, however they do argue for a general superiority of a Borda system, in which each voter ranks a candidate such that the one elected is most accepted by all instead of simply preferred by a majority.

Table 1: Some published classification models for the UCI Dataset, adapted from (Alam et al., 2020)

Year	Method	Results
2009	K-nearest neighbor	A = 82%
2017	Decision tree	A = 80%
2018	Random forest	A = 59%
2018	Bagging	A = 73%
2019	Logistic Model Tree	A = 69%
2019	Gradient Boosting Tree	A = 82%
2020	Gradient Boosted Decision Tree	A = 89%

3.2 Data pre-processing and partitioning

The dataset used is generated from the information of credit card clients in Taiwan. It is initially made up of 24 attributes and 30,000 records (tuples). The attributes contain only real and integer characteristics, and there are no missing values in the dataset. Our data has additionally been split into training and test data using an 80:20 split. A 70:30 split has traditionally been used in past literature, but our split should still allow for accurate error estimation.

The 'id' column is removed from the dataset as it serves no purpose in this study. The dataset is also split so that our target variable (default) is its vector.

A min-max scaler is used to scale the dataset because several attributes significantly differ between their minimum and maximum values. Leaving the data unscaled can lead to variables having a disproportionate impact on the predictive models. The min-max scaler is the most commonly used normalisation scaler across the literature.

3.3 Data Resampling

The data contains an imbalanced number of instances of the response classes; there are far fewer instances of default than not. We have decided to resample as seen in other studies (Alam et al., 2020; Wang, 2020; Shantanu and Soibam, 2017) to add bias to balance the dataset, which may improve the training of predictive models. Various sampling techniques will be used on the training data and their results compared. These methods include:

- SMOTE (over-sampling)

This oversampling technique involves adding instances of the minority class to augment a more balanced dataset. The SMOTE method generates new instances of the minority class using the Euclidean distance between a random minority instance and its k-nearest neighbours.

- Tomek Links (under-sampling)
  This undersampling technique involves removing instances from the majority class to augment a more balanced dataset. Since instances are removed, this technique comes at the cost of some information. The Tomek method finds the majority class instances with the smallest Euclidean distance from minority class instances to remove, thus, creating more contrast between the classes.
- SMOTE-TomekLinks
  This method combines the two techniques above to achieve a balanced dataset.
- No sampling

3.4 Feature Engineering

In this study, we also compare several feature selection and engineering methods to determine how these may impact the accuracy of each prediction model. The feature selection models in use are:

- No feature selection (use all attributes)
- Lasso feature selection
- Lasso regression is used to zero out insignificant variables.
- XGboost feature importance
- A gradient boosting model maps feature importance and identifies essential features.
- Manual Engineering

3.5 Predictive Modelling

Seven predictive models are used, including K-Nearest Neighbours, QDA, Linear Regression, LDA, Naïve Bayes, Random Forrest, and Gradient Boosting (XGBoost).

Each model will be trained using a combination of sampling and feature selection methods. The training and prediction will be made in a 5-fold cross-validation setup.

As a result, the dataset will be partitioned into five folds; four folds are used to train a model, and the fifth fold is used as a validation set to score the model. The cross-validation function iteratively changes the fold becoming the validation set such that the model is trained and validated five times. We will use the average score from this 5-fold cross-validation as the score for each model/sampling/feature engineering combination.

3.6 Scoring

There are several methods of scoring a predictive model evidenced by the literature – we will compute the following scores for each model: accuracy, precision, recall and F1-score. While the generic approach uses accuracy to score a model, this is not an effective measure for unbalanced datasets. Instead, we will primarily depend on the F1 score to rank our models. The F1-score finds the harmonic mean of precision (fraction of identified positives which are true positives) and recall (fraction of test-positives that were correctly identified), punishing extreme values.

The best combination of sampling and feature engineering will be determined for each model based on the F1-score.

3.7 Optimisation

Using the best feature selection for each of the 7 models, based on the initial F1-score, we will then tune the hyperparameters of each model to maximise their F1-scores in a 5-fold cross validation as previously described.

3.8 Model Selection and Combination

Once the final, feature selected and tuned models are output, they are ranked by F1-score. Using a soft voting system, we combine the best models. Combination models return the average output of the input models. Combinations of two to seven models are created in order of the best performing individual models. Each of these combinations will also be scored.

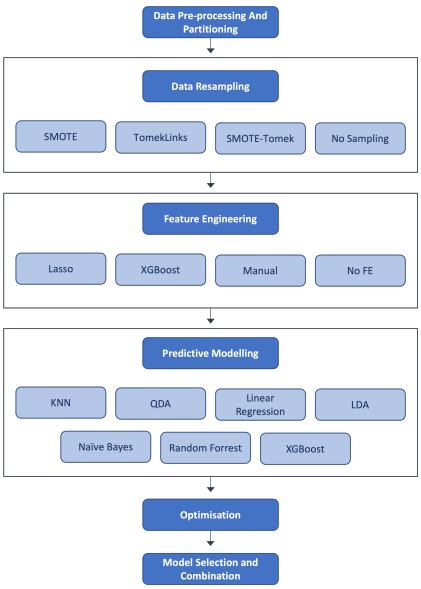

Figure 6: Block Flow Diagram of Methodology

3.9 Approach Amendments

Removed SVM – we initially included an SVM model, but this was removed as it was too computationally intensive and required too much time to process. It also was not compatible with lasso feature selection.

Randomised rather than Grid search – the randomised search function was more efficient and less computationally intense than the grid search approach we initially took.

Standard to Min-max – better deals with outliers and used more frequently in literature

Removed Hard voting – we initially had a hard-voting system for model combination, but this was removed as it performed poorly.

4 Results

Here we present the performance of our methodology on the training data.

Table 2: Preference Space for Naive Bayes

	Feature Engineering Method			
Sampling Method	LASSO	XGBoost	Manual	None
SMOTE-TL	0.63	0.66	0.64	0.69
SMOTE	0.62	0.65	0.60	0.66
TL	0.46	0.53	0.51	0.52
NONE	0.44	0.49	0.50	0.50

A 4 x 4 preference space was generated for each of the 7 models with the f1-score as the relevant metric. For example, Table 2 shows that NB 'prefers' SMOTE-TL as its Sampling Method and XGBoost for its Feature Engineering method. Table 3 shows the preference of each model. Notably, SMOTE-TL is the unanimous choice of sampling - this is further examined in Figure 7. There is an almost even split between 'Manual' and XGBoost (which keeps 10 features with the criterion being their importance should be > 0.01) for Feature Engineering.

Table 3: Model Preferences

Model	Sampling Method	Feature Engineering
KNN	SMOTE-TL	Manual
QDA	SMOTE-TL	Manual
LR	SMOTE-TL	Manual
LDA	SMOTE-TL	Manual
RDF	SMOTE-TL	XGBoost
XGB	SMOTE-TL	XGBoost
NB	SMOTE-TL	XGBoost

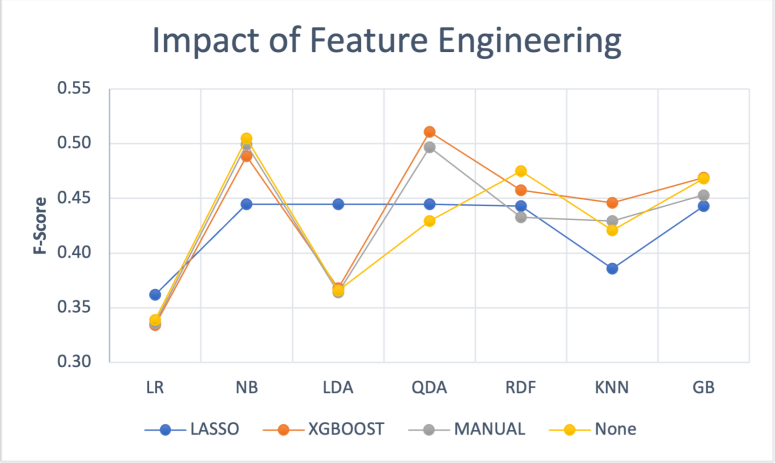

Figure 7: Impact of Feature Engineering

To isolate the effect of feature engineering, sampling was held constant (as none) and each model was fitted to the relevant training data and then the 5-fold cross validation was performed.. All feature engineering methods has a similar impact on LR, NB and LDA except for LASSO, which was visibly worse on NB but better on LR and LDA. On QDA manual and XGBoost feature selection returned similar F-scores, which were noticeably better than the F-scores given by Lasso and None. On RDF and GB, all feature selection methods give similar F-scores, with no feature engineering marginally better. Lastly, on KNN XGBoost gives marginally better results than manual and no feature engineering, however LASSO is notably worse. This is potentially because LASSO only keeps the PAY2 attribute which is may not be sufficient by itself to predict default for most models.

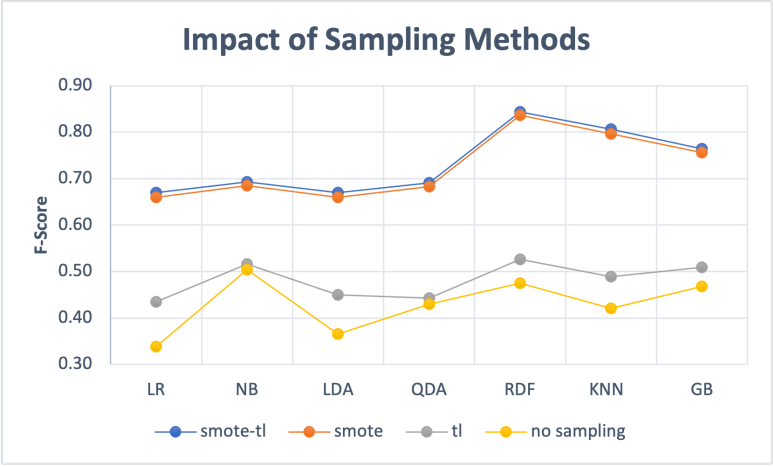

Figure 8: Impact of Sampling Methods

Similarly, to isolate the effect of sampling method, feature engineering was held constant (as none) and the same process as above was performed. The results show that over-sampling is significantly superior to under-sampling for all models examined, which is consistent with the results of Alam *et. al* (2020).

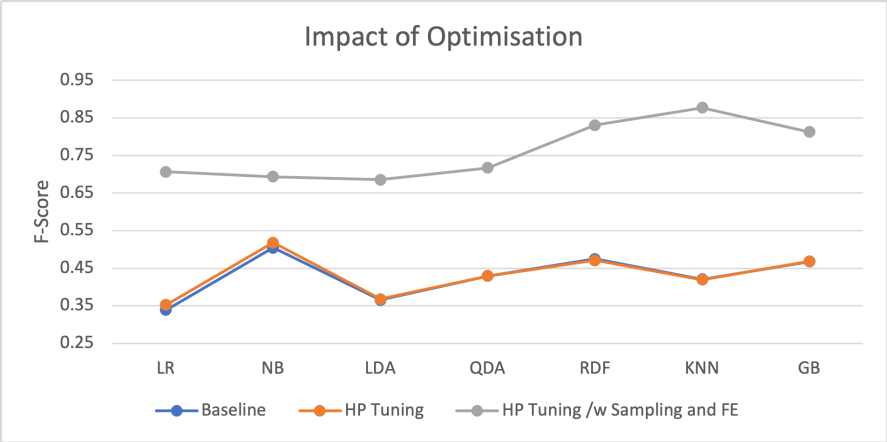

Figure 9: Impact of Optimisation

To isolate the effect of hyperparameter tuning three sets of the 7 models were run in a similar fashion as above. The first with baseline models, the second with hyperparameter tuning alone and the third with hyperparameter tuning and the model's preference of sampling and feature engineering. The results shown in Figure 7 suggest that hyperparameter tuning by itself does not contribute significantly to improved model performance. This is perhaps because the default parameters are already well suited to the data set. In constrast, the 'correct' sampling and feature engineering significantly improve performance. For example, KNN nearly doubles its F-score and least improvement (NB) is still 0.2 points higher. Interestingly, the preferred sampling and feature engineering cause the model performance order to change; for example, NB performs best as baseline, but KNN performs better when both NB and KNN have their preference.

5 Final Predictions on Test Set

Table 4: Base Results on Test Data (with MinMax Scaling)

Scoring Method	KNN	QDA	LR	LDA	NB	RDF	GB
Accuracy	0.81	0.54	0.82	0.82	0.76	0.83	0.83
Recall	0.34	0.8	0.21	0.24	0.62	0.37	0.35
Precision	0.57	0.29	0.75	0.72	0.45	0.67	0.71
F-score	0.42	0.42	0.33	0.36	0.52	0.47	0.47

Table 5: Results after optimisation (each method trained on its choice of sampling and feature selection method)

Scoring Method	KNN	QDA	LR	LDA	NB	RDF	GB
Accuracy	0.7	0.75	0.77	0.77	0.4	0.79	0.82
Recall	0.47	0.64	0.59	0.58	0.89	0.48	0.41
Precision	0.35	0.43	0.46	0.46	0.25	0.51	0.62
F-score	0.4	0.52	0.52	0.52	0.39	0.49	0.49

QDA, LDA and LR all achieved the top F1-score of 0.52. If recall is used as the secondary metric then QDA is the best overall model.

For voting, each the models were sorted from best to worst performance and then combined iteratively such that the first combination was models 1 and 2, the second 1, 2 and, 3 and so forth. For each combination the voting model was trained on each of the 4 feature engineering training data sets. The best result overall is shown in Figure 10

Best Prediction	Voting type	Feature Selection	Feature Selection
USING ALL 7 MODELs	Soft	LASSO	0.45

Figure 10: Best Result after Voting

Table 6: Full Factorial Investigation of Preference Space for Naive Bayes

Scoring Method	SMOTE-TL LASSO	SMOTE-TL XG-BOOST	SMOTE-TL MAN-UAL	SMOTE-TL NONE	SMOTE LASSO	SMOTE XG-BOOST	SMOTE MAN-UAL	SMOTE NONE	TL LASSO	TL XG-BOOST	TL MAN-UAL	TL NONE	NO SAM-PLING LASSO	NO SAM-PLING XG-BOOST	NO SAM-PLING MAN-UAL	NO SAM-PLING NONE
Accuracy	0.79	0.78	0.77	0.41	0.79	0.78	0.78	0.42	0.83	0.81	0.78	0.79	0.83	0.81	0.79	0.8
Recall	0.51	0.58	0.53	0.89	0.51	0.57	0.54	0.89	0.33	0.48	0.52	0.56	0.33	0.44	0.25	0.53
Precision	0.51	0.48	0.46	0.25	0.51	0.48	0.48	0.25	0.72	0.56	0.47	0.5	0.72	0.57	0.5	0.52
F-score	0.51	0.52	0.5	0.39	0.51	0.53	0.51	0.39	0.45	0.52	0.49	0.53	0.45	0.5	0.33	0.53

As mentioned, it was assumed that whichever sampling-FE combination the model preferred when cross-validated on the training data would stay the same on the test data. This also prevented running hyper-parameter tuning 16x7 times which would have been computationally taxing. To validate this assumption, one model (NB) was chosen and a full factorial search was performed; i.e., hyper-parameter tuning was done for each combination in the preference space and then predictions were made on the test data. Table 6 shows that NB achieves the best F1-score with SMOTE-TL as sampling and XGBoost feature engineering, which is what was predicted on the training data (see Table 3) suggesting that the assumption made was a fair one.

6 Conclusion

In conclusion, we conducted a rigorous search of the 'preference space' of feature engineering and sampling method for 7 models. Each model was tuned with their preference. The best performance on the test data was an F1-score of 0.52 (for individual model) and 0.45 for voting. These results are slightly worse than expected based on the literature review; this is potentially due to slightly different data splits as well different models. One possible solution to improve accuracy is development of voting such as that proposed by (David *et al.*).

References

[1] S. Hamori, M. Kawai, T. Kume, Y. Murakami, and C.Watanabe Ensemble learning or deep learning? Application to default risk analysis,. *J. Risk Financial Manage., vol. 11, no. 1, p. 12,*, 2018.

[2] T.-C. Hsu, S.-T. Liou, Y.-P. Wang, Y.-S. Huang, and Che-Lin 'Enhanced recurrent neural network for combining static and dynamic features for credit card default prediction,. *Speech Signal Process. (ICASSP)*, 2019.

[3] Alam, T., Shaukat, K., Hameed, I., Luo, S., Sarwar, M., Shabbir, S., Li, J. and Khushi, M. An Investigation of Credit Card Default Prediction in the Imbalanced Datasets *IEEE Access, 8, pp.201173-201198*, 2020.

[4] Wang, J., Will You Be Able to Make Your Credit Card Payment?. *Medium*, 2020.

[5] Burka, Dávid, et al Voting: A Machine Learning Approach. *European Journal of Operational Research, 2021, pp. European journal of operational research*, 2021.

[6] Hosseini D., Kernel Methods Voting: A Machine Learning Approach. *Lecture for BENG0095*, 2021.

[7] Dominguez, M, Kernel Methods Predicting Credit Card Defaults with Machine Learning. *Medium*, 2021 .