University College Dublin

School Of Mathematics and Statistics

COMPARISON AND EVALUATION OF DIFFERENT MACHINE LEARNING METHODS AT PREDICTING CREDIT CARD DEFAULT

By

Sidney Harshman-Earley
Denis O'Riordan

in partial fulfilment of ACM40960 -
Projects in Maths Modelling required
for the completion of the degree
MSc in Data and Computational Sciences

Principal Supervisor: Dr Sarp Akcay
Programme Coordinator: Assoc. Professor Lennon Ó Náraigh

July 2022

We hereby certify that the submitted work is our own work, was completed while registered as candidates for the degree stated on the Title Page, and we have not obtained a degree elsewhere on the basis of the research presented in this submitted work.

—

Abstract

Research into utilising ML methods to predict credit card default has largely been limited to studies conducted on a single historic data set. As part of an applied data science competition organised by the credit card company American Express, an industrial-sized dataset with masked feature labels was released which in turn provided the foundation for further research in this field. A subset of the data was used and had feature engineering and noise-reduction applied to create a dataset of 458'913 individual customer observations and 129 features. Exploratory data analysis found the P_2 variable, in particular, displayed a strong correlation with the target variable - defaulting on credit card payments. Four ML classification models - logistic regression (LR), logistic regression (P_2 feature only), Random Forest (RF) and Neural Networks (NN) - were trained, validated and tested on independent data to predict the target. The Neural Network and Random Forest models displayed the best ability to correctly identify cases of credit card default in the test data. These were the top two performing models on the test data in terms of overall accuracy, sensitivity and AUC, with the NN model marginally outperforming the RF. Only marginal improvements were gained from the full 129 variable logistic regression model compared to the P_2-only logistic regression model. The report concludes that there certainly are benefits to financial institutions to using advanced ML methods to identify potential incidences of credit card default and a method to mitigate credit risk.

Contents

List of Figures

List of Tables

1 Introduction

A credit card is a financial instrument issued by banks and other financial institutions with a pre-set credit limit allowing customers to make cashless transactions. Each month or at different intervals, the credit card provider issues a statement with details of spending history, the interest charged, balance due, and payment deadlines. As of May 2022, there were 1.5 million active credit cards in circulation in Ireland, with daily spending exceeding €1 million.[1]. Further to that, according to research conducted by the Central Statistics Office in 2018, 12.7% of all households have credit card debt, this figure reaches 21.0% for households with two adults and one to three children under 18.[2]

A default event concerning a credit card occurs if a customer does not pay the amount due within 120 days of their latest statement date. However, the deadline can depend on the issuing institution. The most significant risk to lenders is a large and unexpected number of customers failing to meet their credit repayment obligations. This is known as credit default risk. Credit default prediction is critical to managing risk in a consumer lending business. Financial institutions are legally obligated to create models that predict credit defaults. Credit card default prediction is based on historical and underlying information of credit card customers. These models advise the bank's capital requirements, which ensure that the bank can absorb losses brought about by a significant increase in credit defaults. Credit default prediction also allows lenders to optimise lending decisions, which leads to a better customer experience and sound business economics. The use of models to predict credit card default is a typical classification problem. Current models exist to help manage risk, but is it possible to create better models that outperform those currently in use? Given that banking portfolios can be worth billions, even marginal improvements in preventing and reducing defaults can be considered significant.

The big data paradigm has revolutionised the banking industry, changing how financial institutions operate. Developments in technology have revolutionized the customer experience and granted access to an ever-growing volume of structured and unstructured information. Institutions can leverage this new information in many ways, including predicting credit default. There have been advances in Machine Learning (ML) techniques such as representation learning methods e.g. Neural Networks, ensemble learning methods such as Random Forest. These models have yet to be widely deployed because of their opaque nature i.e. Black-Box methods. From a regulation perspective, transparency and explainability are necessary when deploying credit default prediction models in practice. However, they may still provide value to financial institutions and may in the future be accepted as valid, clear and sound methods for predicting credit card default.

The aim of this project is to apply a range of machine learning techniques to predict credit card default using the historical data of credit card customers. The following report describes the process undertaken to compare a number of models. Beginning with an industrial size dataset, cleansing, formatting and analysing the data before using it to train, tune and evaluate the chosen models based on the historical data of credit card customers.

[1] https://www.centralbank.ie/statistics/data-and-analysis/credit-and-debit-card-statistics/daily-credit-and-debit-card-statistics

[2] https://www.cso.ie/en/releasesandpublications/ep/p-hfcs/householdfinanceandconsumptionsurvey2018/debtandcredit/

2 Review of Literature

There have been numerous articles and papers within the scope of using ML methods to predict credit default, this includes the prediction for credit card default.

Studies examining credit card default have been concentrated, mainly using data originally used as part of Yeh & hui Lien (2009). This data is currently freely available as the Default credit card clients Data Set on the UC Irvine Machine Learning Repository.[3] This data, collected in October 2005 is from a cash and credit card issuing bank in Taiwan. Among the 30,000 observations, 22.12% are the cardholders who defaulted on payment. This research employed a binary variable, default payment (Yes = 1, No = 0), as the response variable. This data uses 23 explanatory variables, including a mix of personal information(age, gender, marital status and education level), amount of credit given, historical bill statements, and historical payment information.

Yeh & hui Lien (2009), compared six classification algorithms - K-nearest neighbour, Logistic regression, Discriminant analysis Naïve Bayesian classifier, Artificial Neural Networks, Classification Trees. In the classification accuracy, the results show that there were few differences in error amongst the six methods. The generated probability of default by the Artificial Neural Network most closely resembled the actual probability of default which was estimated using a novel "Sorting Smoothing Method".

Other research on predicting credit card default in subsequent years utilised this data for training and evaluating models. The following is a sample of articles available, applying a wide variety of Machine Learning methods to this classification problem. Neema & Soibam (2017), took a similar approach in the choice of methods but attempted to predict the best possible cost-effective outcome from the risk management perspective. Naive Bayes Estimator and Random Forest classifiers were introduced. This was evaluated using a cost function which gave a higher cost to defaulters classified not correctly as they are the minority in the data. It was concluded that original data with the Random Forest algorithm is the best in terms of a good balance on cost versus accuracy.

Yang & Zhang (2018) introduces two new methods used to predict credit card default. Support Vector Machine (SVM) involves using a kernel function to map the predictor data into a high-dimensional feature space where the outcome classes are easily separable. XGBoost and LightGBM, forms of gradient boosted trees algorithm were used. LightGBM and XGBoost were both deemed to have the best performance in the prediction of categorical response variables.

While other studies have used Neural Networks in predicting credit card defaults, the models used have been vague, and little detail has been given on the architecture or tuning of the model. Chou & Lo (2018) trialled a range of Networks, experimenting with two to five layers with the number of processing units of 64, 32 and 16 units. Neural Networks with three layers and 64 units recorded the highest accuracy of all configurations.

Due to a lack of credit card default-specific data pertaining to defaulting on payment, all available studies which predict credit card default utilise the Taiwan data, which is both region specific, over 15 years old and obtained at a time when credit card issuers in Taiwan faced a credit card debt crisis.[4]

[3] https://archive.ics.uci.edu/ml/datasets/default+of+credit+card+clients
[4] https://sevenpillarsinstitute.org/case-studies/taiwans-credit-card-crisis/

3 Exploratory Data Analysis

3.1 Background

Data used for this project was obtained from the Kaggle website. One of Kaggle's main features is its competition platform.[5] Kaggle allows users to organise and host data science competitions. This competition provides an industrial-scale data set to build a machine learning model. The model is used to predict credit card default using anonymised customer profile information from March 2017 to March 2018. The target binary variable i.e. default is calculated by observing 18 month's performance window after the latest credit card statement, and if the customer does not pay the due amount in 120 days after their latest statement date, it is considered a default event.

3.2 Data Structure

train_labels.csv
A list of unique customer identifiers `customer_ID` with the target label `target` indicating a default event with `target = 1` indicating a default, `target = 0` indicating no default. There are 458913 observations in this data, meaning 458913 unique AMEX customers. 74.1% of customers in the data did not default on payment, while 25.9% defaulted.

train_data.csv
This tabular data set corresponding to the training data, consisting of 190 features and over 5.5 million observations. It contains multiple statement dates per customer_ID ,and each observation corresponds to a monthly statement for a customer. The data is ordered firstly by `customer_ID`, corresponding with the train_labels.csv file, and chronologically by statement date. The dataset contains aggregated profile features for each customer at each statement date. There are 177 numeric features which are anonymised and normalized and fall into the following general categories:

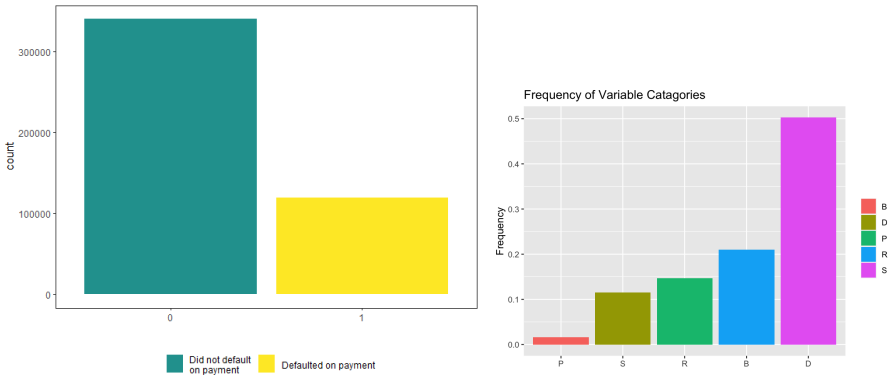

Figure 1: Frequency of Lables / Variable Catagories

Table 1: Variable Categories

Variable Name	Category
D_*	Delinquency variables
S_*	Spend variables

[5] https://www.kaggle.com/docs/competitions

Variable Name	Category
P_*	Payment variables
B_*	Balance variables
R_*	Risk variables

There are 11 categorical variables - B_30, B_38, D_114, D_116, D_117, D_120, D_126, D_63, D_64, D_66 and D_68.

The remaining two unaccounted-for features are the variable S_2 which contains the date of the statement and `customer_ID`, used to match the target label.

3.3 Correlation to Target Variable

The Pearson Correlation Coefficient measures the strength of the linear association between two variables calculated between the target variable and each feature. A value of 0 indicates that there is no association between the two variables. An absolute value greater than 0 indicates an association between the variables, either positive or negative, depicted by the sign of the value.[6]

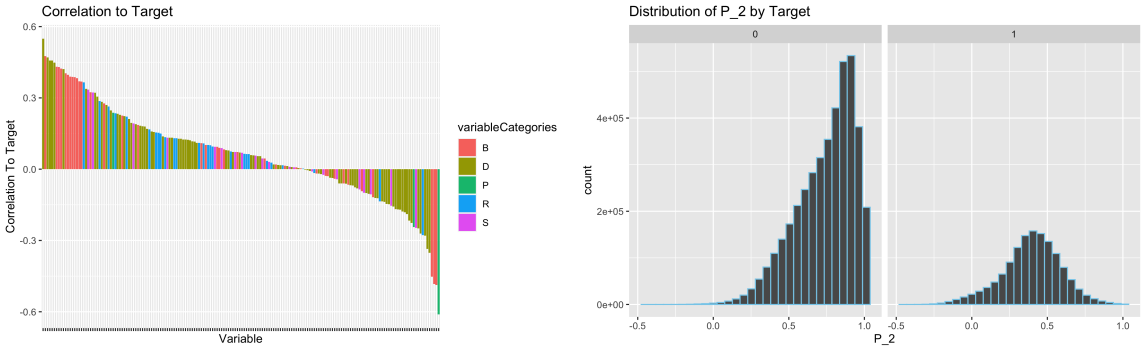

Figure 2: Relationship of features to Target Variable

Payment variable P_2 shows the most substantial linear relationship with the target variable. It is the only variable to achieve an absolute correlation value over 0.6. It is a negative value, so for the sample, as P_2 increases, the likelihood of a cardholder defaulting increases. The majority of variables have an absolute correlation value of less than 0.3, demonstrating a weak or even non-existent linear relationship. Looking at the variable categories, Balance (B_*) and (D_*) Delinquency variables are the majority of features with a correlation with the target of greater than 0.4.

Exploring the P_2 variable in more detail, the extent of its relationship with the target is clear. The distribution of this variable, split by target value, depicts a left-skew peaking at 0.9 for those not in default and an approximately normal distribution with a mean of 0.5 for those who default. The vast majority of observations with a value of P_2 greater than 0.5 are not in default. There are approximately even numbers who are in default and not in default with a value of P_2 less than 0.5. This could indicate the feature is related to or a function of a type of credit-worthiness or credit-score variable. This variable is now used as a baseline for predictions of models created. A simple logistic regression model with P_2 as the sole

[6]https://statistics.laerd.com/statistical-guides/pearson-correlation-coefficient-statistical-guide.php

predictor may be valuable as a baseline for measuring more complex models against. It is trained, tuned and evaluated in an identical format to the full logistic regression model.

4 Feature Engineering

Dimension Reduction
Several feature engineering steps were performed independently on the predictor variables. Feature engineering was done to create a more manageable-sized data set with uninformative features removed.

Most ML algorithms either suffer or are inoperable with missing data. Imputation methods must be utilised, which can bias the column's information if the proportion of missing data is too significant A threshold of 10% was chosen of which columns would be removed. Of variables beyond that threshold, 77% had greater than half of its data missing.

Features dominated by a single value or small range of values do not hold much information and would not contribute to an effective classification model. When creating data models, such variables are uninformative and should be considered to be removed. Choosing a conservative variance threshold of 0.001, variables where 99.9% of the values are similar, are removed.

Highly intercorrelated features mainly affect the interpretation of a regression model as then the regression coefficients are not unique and have influences from other features. More broadly, multi-collinear features increase dimensionality, which increases the computational intensity of ML algorithms without adding extra information which could be used to predict the target. Columns with a pairwise correlation in excess of a high correlation threshold of 0.9 are considered for removal.

These three filters reduced the number of features in the dataset from 189 to 129.

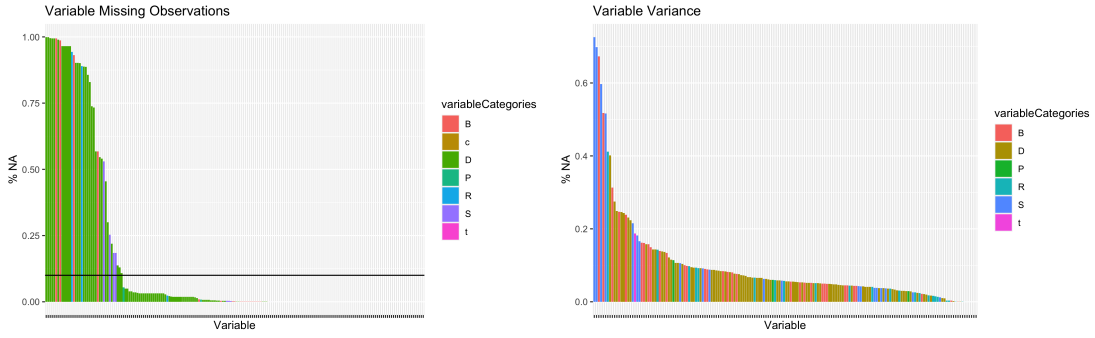

Figure 3: Proportion Missing, and Variance of Feature Columns

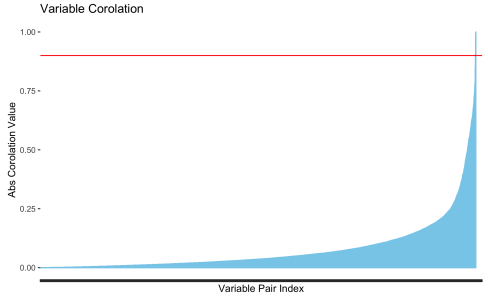

Figure 4: Pairwise Correlation of Feature Variable

Noise Removal

As part of anonymising the data, noise was injected into a number of rational variables. This can be observed by plotting the histogram of variables. Viewing the variables S6, D111 and B16, we can see that there are pillars of high frequencies with no observations between them. Each variable has a different interval for which values are seen, but the commonality is the width of each pillar which is 0.01. This can be easily rationalized by trying to anonymise the standard interval for US currency ($0.01) but has been applied across a number of discrete variables to make them continuous.

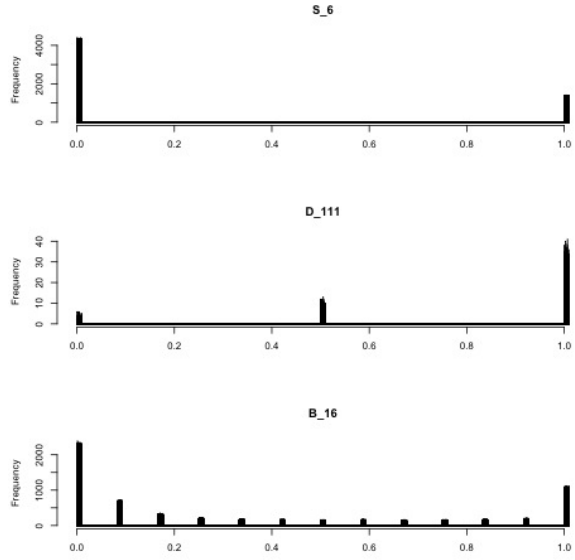

Figure 5: Sample of features with noise added

A one-sided Kolmogorov-Smirnov goodness-of-fit test is implemented to identify these intervals where noise occurs. The Kolmogorov-Smirnov test is a nonparametric test that measures agreement between the cumulative distribution function of observed data to a specified distribution (Dimitrova et al. 2020). In this case, we are comparing to a uniform $U(0, 0.01)$ distribution for each test.

The hypothesis being tested is:
H_0 : The sample comes from the specified theoretical distribution
H_A : The sample does not come from the specified theoretical distribution

The test statistic is the absolute value of the largest distance between the two distributions functions.

$$D = sup_x|F_n(X) - F(X)|$$

where:

- n is the sample size
- $F_n(X)$ is the observed cumulative frequency distribution of a random sample of n observations
- $F(X)$ is the theoretical cumulative distribution function

This test statistic is evaluated against the K-S tables to give a p-value. To evaluate if noise $[0, 0.01]$ had been added to the data, the bounds for each noise pillar are found iteratively by

taking the smallest value in the dataset and adding 0.01 to it to create an upper bound. All values within this pillar are first scaled to the interval $[0, 0.01]$ by taking the remainder modulo 0.01. The scaled values are sampled using the Kolmogorov-Smirnov test against the $U(0, 0.01)$ distribution. These values are then omitted, and the next minimum data point is selected.

There are two values of interest:
1) The interval between observed values with noise removed.
Each minimum value recorded is rounded to the nearest rational number, with the smallest accepted value being 0.01. Limiting the lower bound of acceptable rational numbers removes the possibility of overlap between pillars. We accept the interval as the most observed interval across all pillars, as values are assumed uniformly discrete.

2) The acceptance criteria for if noise has been added to the data.
The acceptance criteria is the percentage of total pillars that do not reject the null hypothesis. As we are observing the null hypothesis, on each test with $p > 0.05$ we can only not reject that the observed values come from a uniform distribution. As such, a high threshold of 0.95 for the acceptance criteria is chosen to ensure variables selected are highly likely to have had noise added.

Variables that were identified as having noise added are converted to an ordinal array of integers representing the underlying rational number. This is done for each variable by first selecting the interval between values, An arbitrarily small number (1×10^{-8}) is added to avoid machine precision errors. Each value in the column is divided by this augmented interval. The floor of the values is obtained and then converted to an integer. It was found using the K_S goodness of fit test that 42 variables had noise added. Each of these received noise reduction as outlined above and were converted to ordinal integer form.

Final Dataset
This cleansed data was merged with the default labels data. Due to computational limitations, the complete data set could not reasonably be used in any full capacity to train computationally intensive ML algorithms. To reduce the length of the data, the most recent observation by date (variable S_2) per customer_id was retained, and all preceding month's data was dropped. This reduced the data to its final form of 458'913 individual customer observations and 129 features which were used for further analysis.

5 Methods

5.1 Models

1. Logistic Regression (LR)
Logistic regression is one of the most simple classification algorithms and is used to model a binary categorical variable given a collection of numerical and categorical predictors, Harrell (2015). The model applies the logistic function to the linear regression model, which is used to model the probability of a binary event occurring.
The model is trained using a three-fold cross-validation method over ten replications.

$$p_i = Pr(Y_i = 1|x_i) = \frac{exp(w_0 + x_i^T w)}{1 + exp(w_0 + x_i^T w)}$$

2. Random Forest (RF)
Random Forests, Breiman (2001), are a very powerful ensemble classification method. Ensemble learning is an aggregation of predictions made by multiple classifiers with the goal of improving accuracy. The method uses Classification Trees and bootstrapping extensively. A random forest is, effectively, a random collection of Classification Trees estimated on random subsets of the data.
A Classification Tree is an iterative process of splitting the data into partitions based on values of the observations and then splitting it up further on each of the branches. The classifier is trained in order to produce pure groups of observations or 'buckets' by minimising the entropy or spread of the target variable in each bucket. The majority value of the target variable in a bucket is then used for predictions. The main disadvantage is that Classification Trees suffer from over-fitting and bias, and using a single Classification Tree would present an over-simplistic model.
In a Random Forest, each Classification Tree is grown independently, and each individual Classification Tree has an equal vote as to what the outcome is. Random forests provide an improvement by means of a minor tweak that reduces the dependence among the trees. The main idea is to use only a random subset of the predictor variables at each split of the classification tree fitting step. This is by default $\sqrt{N}$ where N is the number of features but can be specified by the user. If we build classifiers on subsets of the variables, then they will behave more independently than if we build them on all of the data. This increases diversity, and averaging results across independent classifiers will be more stable than averaging results on dependent ones. The main parameter of the Random Forests to be tuned is the number of variables used at each split and is tuned to prevent overfitting and improve efficiency.
The model is trained using a three-fold cross-validation method over ten replications. A grid of values $(20, 30, 40, 50)$ for the number of variables used at each split of the RF is evaluated and the optimal value is chosen.

3. Deep Neural Network (DNN)
Neural networks are machine learning methods that simulate the biological learning mechanism of the brain (Goodfellow et al. (2016), Zhang et al. (2021)). Neural networks learn representation features encompassing complex relations between inputs and output. Representation learning is the use of Machine Learning to not only learn the mapping from representation features to output but also to learn the representation itself. Deep Learning

methods introduce representations that are expressed in terms of other simpler representations, thus enabling to derive complex concepts out of simpler ones. In practice, a neural network is a series of layers made up of nodes with subsequent layers separated by activation functions.

A number of steps are taken in pre-processing of the data. The scaling of numeric variables is important as to not saturate hidden neurons and to balance the learning rate of all neurons. Neural Networks cannot process categorical variables as inputs, so the categorical variables are one-hot-encoded to create a set of dummy features, each with a { 0, 1 } set of values.[7]

The model used is a fully connected deep neural network with three layers. ReLU (Rectified Linear Unit) activation is implemented for hidden layers. As the labels are binary, binary classification is used therefore, the sigmoid activation is applied to the output layer. The Adam optimizer with a learning rate of 0.0001 is chosen for its computational efficiency and wide adoption in DNN models. During tuning, a number of other parameters are grid searched through to give results of a number of different models:

- **Width Size:** The flag Equal Widths indicates whether the neural network will have equal widths of neurons at all layers, each the width of the input layer. If false, each subsequent layer is half the width of the previous.
- **Dropout:** The first and second layers of neurons have an optional dropout layer. When active, the dropout rate is either a lower value of0.25 or a higher value of 0.5. Dropout layers randomly deactivate neurons with a frequency of the specified rate during training to limit dependence on any subset.
- **Regularization:** l2 regularization is available on each layer, adding the squared magnitude of coefficients as a penalty to the loss function. This is a feature to help generalise the model and stop overfitting. Values of 0, 0.001 and 0.0001 are used to tune.
- **Normalization:** Batch normalisation is available on each layer. When activated output of each layer is scaled and centred before being passed to subsequent layers.

5.2 Evaluation

The models are compared on a number of metrics based on the correspondence of predicted labels for the test data versus observed labels of the test data.

		Observed Value	
		0	1
Predicted Value	0	True Negative (TN)	False Negative (FN)
	1	False Positive (FP)	True Positive (TP)

Figure 6: Cross tabulations between predicted and actual outcomes

Sensitivity, otherwise known as the True Positive Rate (TPR), quantifies the proportion those who defaulted were predicted to have defaulted. It is calculated by TP/(TP + FN). Specificity, otherwise known as the True Negative Rate (TNR) quantifies the proportion those who did not default were were predicted to not have defaulted. It is calculated by TN/(TN + FP).

[7] https://www.kaggle.com/code/dansbecker/using-categorical-data-with-one-hot-encoding/notebook

Accuracy is the overall classification accuracy of a results set and is given by (TP + TN)/(TP + TN + FP + FN).

As the binary classification models can produce their predictions in the form of probabilities, varying the classification threshold - probability threshold to which default label is applied - affects the predictions and therefore the performance of each model. Further evaluation and comparison can be performed by comparing the performance of a classification model at all classification thresholds. The receiver operating characteristic - ROC curve plots Sensitivity versus 1 - Specificity over. The curve illustrates the diagnostic ability of a binary classifier as the classification threshold is varied. The ROC determines how often will a randomly chosen 1 outcome have a higher probability of being predicted to be a 1 outcome than a randomly chosen true 0. The larger the area under the curve - AUC, the better is the discrimination. A perfect classifier would have AUC = 1. A classifier not better than random guessing would have AUC = 0.5 and the corresponding ROC curve would resemble a plot of the identity function ($y = x$).[8]

A fourth metric was also used to compare models. The evaluation metric used for the Kaggle competition [9] is the mean of two measures of rank ordering - Normalized Gini Coefficient,G, and default rate captured at 4% , D. $M = 0.5 \times (G + D)$. G is an extension of the AUC and is calculated as $G = 2 \times AUC - 1$. The Default rate captured at 4% is the Sensitivity for a classification threshold set at 4% of weighted total sample count.

[8] https://developers.google.com/machine-learning/crash-course/classification/roc-and-auc
[9] https://www.kaggle.com/competitions/amex-default-prediction/overview/evaluation

6 Results

6.1 Model Tuning Results

The training and validation process yielded the following four models which were used to predict the test observations.

- Logistic regression model of all features using a classification threshold of 0.223. This will be referred to as the "full LR" model.
- Logistic regression model using P_2 feature as the only predictor variable using a threshold of 0.208. This will be referred to as the "reduced LR" or "simple LR" model.
- The optimal Random Forest model best on repeated cross-validation was found to use a random sample of 20 variables at each split and generating 100 trees.
- Deep Neural Network consisting of three hidden layers, with 160 units in each layer. A dropout of 0.5 and the first two hidden layers. L_2 regularization at each layer with a λ of 0.001 and batch normalization at each layer. The Adam optimizer with default learning rate of 0.0001.

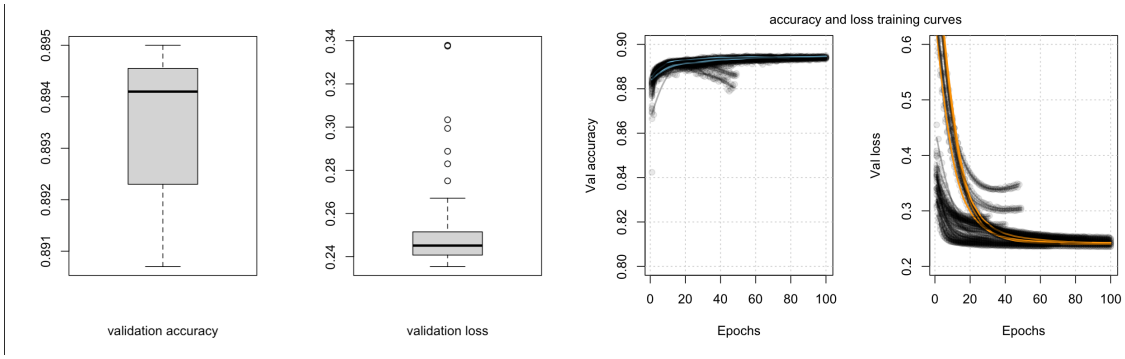

Figure 7: Neural Network Turning Results

6.2 Model Comparrison

Full table of result metrics can be seen in Appendix A.

In terms of accuracy, NN and RF both slightly outperform LR and achieve an accuracy that is 0.07 greater than the accuracy produced by the reduced LR model. That pattern of results is repeated in the Sensitivity. Of all the customers who defaulted in the test set, the Neural Network and Random Forest correctly identified over 90% of them. The full LR model correctly identified 85% while the simple LR identified under 80%. The full LR model achieves the highest specificity on the test data out of the four models with over 0.91 recorded. The simple LR model records the second highest specificity with 0.89 while both the Neural and Random Forrest recorded specificities of 0.78.

As the Sensitivity has a positive effect in the calculation of the AMEX metric, it is only logical that NN and RF would fare better than the two LR models. With a value fo 0.71, the NN model is the only one to record a AMEX metric greater than 0.7. The RF model records an metric value of 0.69 while the full and simple LR models achieved values of 0.61 and 0.58 respectively.

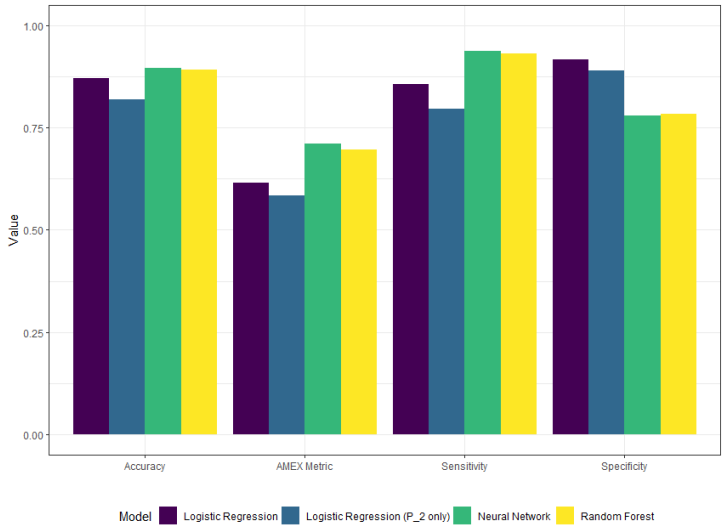

Figure 8: Bar Chart of Results by Model

On initial inspection, the four ROC curves produced by the model appear quite, however the ROC curves for the simple LR model and the RF model don't extend as close to the top-left of the plot as much as the full LR and the NN models. This is confirmed when comparing the Areas Under the Curve (AUC). The full LR and NN models recorded an AUC of over 0.95 while the other two models only recorded AUC's of 0.92.

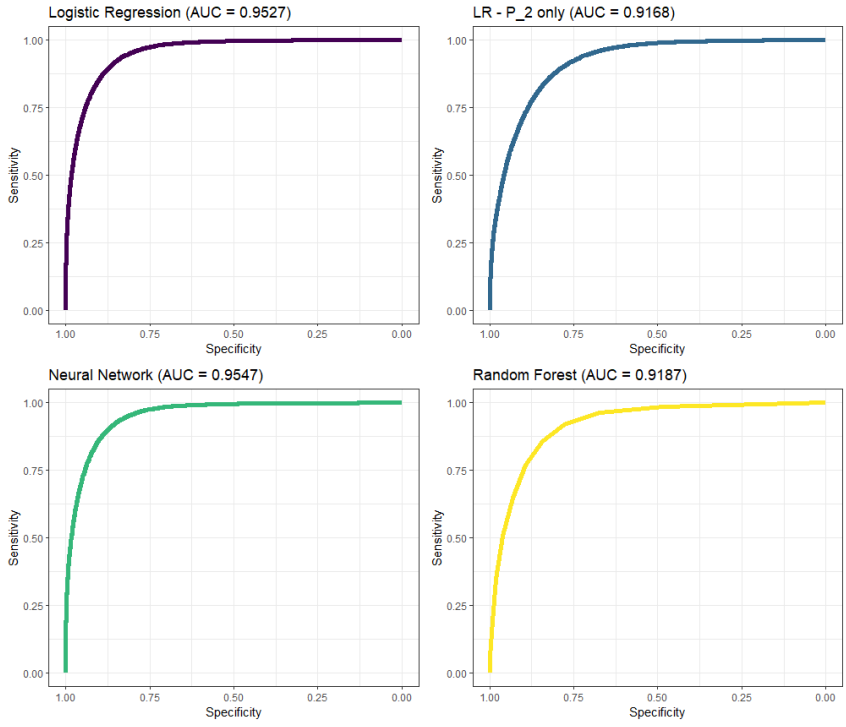

Figure 9: Receiver Operating Characteristic (ROC) curve for each Model.

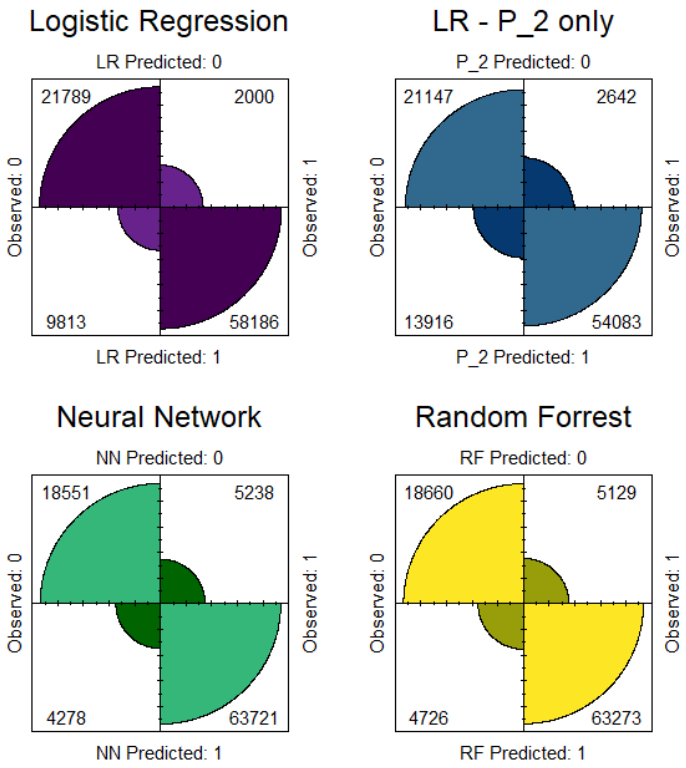

Figure 10: FourFold Plot of Test Data Results

The fourfold plots of the four models produced show largely similar shapes. Upon reading the corresponding section labels clear distinctions can be made. The neural network has correctly identified the most defaults. While Logistic regression has identified the most non-defaults.

7 Discussion

7.1 Context of Results

The results of this study are mainly in-line with results previously seen. While being evaluated on slightly different metrics, the results of this study share a similar outcome to the original study on the Taiwan Credit Card default data, Yeh & hui Lien (2009). That paper found the Neural Network to be the optimal classifier out of a range which included logistic regression and single classifier trees. In a follow-on paper, Neema & Soibam (2017), the Random Forest classifier performed better in comparing models based on a cost control perspective, assigning a higher cost to defaulters classified not correctly. Chou & Lo (2018) tuned a NN to predict default on the Taiwan credit card data. It was found that the NN with the most hidden units was the optimally tuned classifier, a result shared in this study. In summary, the results of this analysis performed on the AMEX data generally confirmed the results from the analysis of the Taiwan credit card default dataset.

A significant takeaway from the results is the closeness in results between a relatively basic logistic regression model and the more advanced and intensive Neural Networks and Random Forest models. Accuracy was only improved by 0.02. A more substantial improvement of 0.08, was made in Sensitivity which is a more important statistic for financial institutions looking to predict credit card default in its customers.
Arguably, a more interesting revelation is the closeness in results between the simple logistic regression model, utilising only one variable and the full regression model, which had an extra 128 predictor variables. Of the five metrics recorded - Accuracy, AMEX Metric, AUC, Sensitivity and Specificity - the maximum improvement gained from the additional variables is 0.06, with the AMEX metric only improving by 0.03. As the feature labels are masked, the description and function of this variable are unknown, but the importance of the P_2 variable in predicting credit card default is evident.

7.2 Limitations & Future Research

The main limitation of this study was the size of the data and lack of computational resources. It meant only a small portion of the data could feasibly be used to conduct the modelling, such as computationally heavy algorithms, e.g. Random Forest. The decision was made to take the most recent statement values for each customer rather than take the full data of the first n customers as it allowed the largest amount of distinct default observations to use to train and predict. Using this subset reduced the amount of data available by a factor of 12 and removed the opportunity to harness the benefits of the time series element in the predictive models. Incorporating the time element of the data into analysis would allow the use of more advanced forms of neural networks going forward. Hsu et al. (2019) proposed using a Recurrent Neural Network (RNN) feature extractor with a Gated Recurrent Unit (GRU) on credit card payment history to leverage the time dependencies embedded in these dynamic features of historical credit card data. Recurrent Neural Networks (RNN) are specifically designed to use recursive architecture to extract patterns from input sequences (Goodfellow et al. 2016). They have been proven helpful in applications that heavily rely on time-variant features. As a result, it is natural to consider RNN models as feature extractors for customer behaviour that often appear as sequences in financial data.

Another option to retain a time-dependent element to the data is to consider creating lag features. Lag features take an indirect approach to using the benefits of time-dependent data.

Lag features reshape the data using summary statistics of a customers set of observations. Each feature would become a set of features. For example, P_2 could become four features P_2_first, P_2_last, P_2_mean, P_2_sd and represent the first, last, mean and standard deviation of P_2 for a credit card holder. Again the number of observations will be reduced as there will only be one row per credit card customer. However, this will increase the width of the data by a factor of $l - 1$ where l is the number of lag features calculated.

The choice of models employed for this project was relatively arbitrary. A selection of three well-known models which were seen to perform this task in previous studies using the Taiwan data were chosen. One primary method, one ensemble learning and one deep learning model were comparatively evaluated. This, unfortunately does not allow us to benchmark the performance against internal models used by AMEX or other banks. This benchmark would allow analysis on whether any of the models employed in this study could provide value and risk mitigation. A small sample of potential models were used in this project. Models used in studies on other data that may improve performance independently or in tandem to other models. include K-Nearest Neighbour,Linear Discriminant Analysis,Logistic Regression, XGBoost and LightGBM.

Other methods of evaluating and comparing the models could be employed in future. Neema & Soibam (2017) used a cost function when comparing methods to predict customers credit card default. A higher penalty or cost was given to defaulters classified incorrectly. A range of cost factors were used to identify a model with the best accuracy in predicting a defaulter in a cost-effective manner.

One of the main obstacles for contextualising the results is the lack of information of the variables included as part of the data. The only detail of the features was the category to which it belonged.[10] Due to this, little inference can be made on the variables and the main outcome of this study is the performance of the models. In particular, it would be difficult to comment on the Delinquency and Risk variables as there is no further information to what they constitute and is impossible to infer any further information from them. Accessing the feature labels would open up avenues to other meaningful analyses of the data, mainly feature importance. The observed values for an important variable are related to the classification values of the target variable. Important variables are those for which the classification performance will drop significantly if they are removed or altered.

8 Conclusion

Over the past years, a strand of literature has attempted to shed light on how advanced ML algorithms such as Neural Networks and Random Forest can be employed to predict various forms of credit risk. This report confirms the additional performance of more simplistic models such as logistic regression. Credit Card issuing institutions have the potential to use models which are better able to identify the complex patterns and relationships between the target variable and the predictor variables. This, in turn, will reduce expenses brought about by default. These models can be used to single out customers at risk of default and provide early intervention or to assess the credit risk of customers before granting them credit cards.

[10] Variable Categories table

9 Bibliography

Breiman, L. (2001), 'Random forests', *Mach. Learn.* **45**(1), 5–32.
URL: *https://doi.org/10.1023/A:1010933404324*

Chou, T. & Lo, M. (2018), 'Predicting credit card defaults with deep learning and other machine learning models', *International Journal of Computer Theory and Engineering* **10**, 105–110.

Dimitrova, D. S., Kaishev, V. K. & Tan, S. (2020), 'Computing the kolmogorov-smirnov distribution when the underlying cdf is purely discrete, mixed, or continuous', *Journal of Statistical Software* **95**(10), 1–42.
URL: *https://www.jstatsoft.org/index.php/jss/article/view/v095i10*

Goodfellow, I., Bengio, Y. & Courville, A. (2016), *Deep Learning*, MIT Press. http://www.deeplearningbook.org.

Harrell, F. E. (2015), *Regression modeling strategies : with applications to linear models, logistic and ordinal regression, and survival analysis / Frank E. Harrell, Jr.*, Springer series in statistics, second edition. edn, Springer, Cham.

Hsu, T.-C., Liou, S.-T., Wang, Y.-P., Huang, Y.-S. & Che-Lin (2019), Enhanced recurrent neural network for combining static and dynamic features for credit card default prediction, *in* 'ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)', pp. 1572–1576.

Neema, S. & Soibam, B. (2017), The comparison of machine learning methods to achieve most cost-effective prediction for credit card default.

Yang, S. & Zhang, H. (2018), 'Comparison of several data mining methods in credit card default prediction', *Intell. Inf. Manag.* **10**(05), 115–122.

Yeh, I.-C. & hui Lien, C. (2009), 'The comparisons of data mining techniques for the predictive accuracy of probability of default of credit card clients', *Expert Systems with Applications* **36**(2, Part 1), 2473–2480.
URL: *https://www.sciencedirect.com/science/article/pii/S0957417407006719*

Zhang, A., Lipton, Z. C., Li, M. & Smola, A. J. (2021), 'Dive into deep learning', *arXiv preprint arXiv:2106.11342* .

10 Appendices

10.1 Appendix A: additional tables

Table 2: Test data results

Model	Measure	Value
Neural Network	Specificity	0.7798142
Logistic Regression	Specificity	0.9159275
Logistic Regression (P_2 only)	Specificity	0.8889403
Random Forest	Specificity	0.7843961
Neural Network	Sensitivity	0.9370873
Logistic Regression	Sensitivity	0.8556891
Logistic Regression (P_2 only)	Sensitivity	0.7953499
Random Forest	Sensitivity	0.9304990
Neural Network	Accuracy	0.8963263
Logistic Regression	Accuracy	0.8713013
Logistic Regression (P_2 only)	Accuracy	0.8196060
Random Forest	Accuracy	0.8926330
Neural Network	AMEX Metric	0.7118359
Logistic Regression	AMEX Metric	0.6160811
Logistic Regression (P_2 only)	AMEX Metric	0.5847342
Random Forest	AMEX Metric	0.6964640
Logistic Regression	AUC	0.9527000
Logistic Regression (P_2 only)	AUC	0.9168000
Neural Network	AUC	0.9547000
Random Forest	AUC	0.9187000

10.2 Appendix B: additional figures

10.3 Appendix C: AMEX_Metric Code

```
amex_metric = function(target, pred){

  df = data.frame(target, pred)
    # Default rate captured at 4%
    top_four_percent_captured = function(target, pred){
    df = data.frame(target, pred)
    # Descending predictions (pred == 1 first)
    df = df  %>% arrange(-pred)
    # define weight (negative labels are given a weight of 20 to adjust for down sampling)
    df[,'weight'] = ifelse(df[,'target'] == 0, 20, 1)
    # get rows under 4% cutoff
    pctCut = as.integer(sum(0.04*df[,'weight']))
    pctCut = df[cumsum(df[,'weight']) <= pctCut,]
    # return
    sum(pctCut['target'] == 1)/sum(df['target'] == 1)
  }

  weighted_gini = function(target, pred){
    df = data.frame(target, pred)
    df = df  %>% arrange(-pred)
    # define weight (negative labels are given a weight of 20 to adjust for down sampling)
    df[,'weight'] = ifelse(df[,'target'] == 0, 20, 1)
    df[,'random'] = cumsum(df[,'weight'] / sum(df[,'weight']))
    total_pos = sum(df[,'target'] * df[,'weight'])
    df[,'cum_pos_found'] = cumsum(df[,'target'] * df[,'weight'])
    #Define lorentz and Gini variables
    df[,'lorentz'] = df[,'cum_pos_found'] / total_pos
    df[,'gini'] = df[,'weight'] * (df[,'lorentz'] - df[,'random'])
    # return
    sum(df[,'gini'])
  }

  normalized_weighted_gini = function(target, pred){
    weighted_gini(target, pred) / weighted_gini(target, target)
  }

  G = normalized_weighted_gini(target, pred)
  D = top_four_percent_captured(target, pred)

  # Return val
  0.5 * (G + D)

}
```

11 Abbreviations

Table 3: List of Abbrevivations

Abbreviation	Meaning
AI	Artificial Intelligence
AMEX	American Express
AUC	Area Under the Curve
DNN	Deep Neural Network
EDA	Exploratory Data Analysis
GRU	Gated Recurrent Unit
K-S	Kolmogorov-Smirnov
LR	Logistic Regression
ML	Machine Learning
NN	Neural Network
RF	Random Forest
RNN	Recurrent Neural Network
ROC	Receiver Operating Characteristic
SVM	Support Vector Machines
TNR	True Negative Rate
TPR	True Positive Rate