Machine Learning: Influences on Credit Card Approval

Bakhtiar Haseeb

May 13, 2020

Abstract

I provide information on what influences credit card applicants to be approved or denied. Often, applicants are accepted and denied using risk assessment depending on delinquent payments and defaults. This dataset includes more interesting predictors such as whether the person owns their house, is self-employed, number of kids, etc. I utilized various machine learning algorithms such as logistic regression, k-nearest neighbors (KNN) Ridge/Lasso regression, Bagging/Random Forests, Gradient Boosting/eXtreme gradient boosting (XGB), support vector machines (SVM), and artificial neural networks. The artificial neural network produced the highest accuracy rates while the results from the gradient boosting models were not far off. The findings across these different models agree that average monthly expenditure and derogatory reports on the applicants' credit report are the most influencing factors on the credit applications. Other factors such as active credit cards, and owning a home seemed to help predict the outcome of the credit card application. Economic intuition follows that lenders make money from interest, which may be the the case with people spending more on average than they can afford. Also, even one derogatory report on the credit report of the applicant can have a strong negative effect on the application, and can possibly be the reason they are rejected, unless they show other signs of stability such as owning a home.

1 Introduction

The purpose of this paper is to use various supervised and one unsupervised learning algorithms to develop a predictive model on whether the application of a credit card was accepted based on certain characteristics. Other papers in the literature have forecasted credit risk such as in Khandani et. al (2010) and used machine learning algorithms for credit card fraud detection as in Awoyemi et. al (2017) and Tuyls et. al (2002). These papers use a few algorithms mentioned in this paper such as classification trees and k-nearest neighbors but only Tuyls et.al using cutting-edge algorithms such as artificial neural networks. The choice in model in these papers is very selective while the findings over many models can provide more depth into the application process. Not many papers exist on the application process but rather on quantifying the risk of a borrower as in Kruppa et. al(2013).This paper focuses on personality and economic characteristics to determine which contribute to approval of the applicant. Based off intuition, the economic variables such as income will have a positive impact on the approval odds and the number of derogatory marks will hurt the odds.

My hypothesis is that the relationship between credit card acceptance is best predicted by using the number of derogatory marks, age, income, average monthly expenditure, and number of active cards. I would expect the number of derogatory marks to correlate negatively with the probability of being accepted because more derogatory marks can be indicative of irresponsibility with finances. The other variables could be associated with a higher chance of being accepted because higher values could indicate more awareness to the credit system, Interestingly, the Equal Credit Opportunity Act (ECOA) stops lenders from using age as a deciding factor in the credit card application process. It will be intriguing to see the size of the effect age has on the acceptance of the application.

To start on the classification model, I fit different logistic regressions to measure the log-odds ratio of the application being accepted. I also used a probit model to compare to the results of the "best" logit model. To compare models, we must compare the correct prediction accuracy rates and the reported AIC value. This is

the first step in accurately predicting whether the application was accepted and expect our machine learning methods produce more accurate results.

For shrinkage methods, Ridge and Lasso regressions will be attempted to hopefully reduce the variance of the prediction. These work by shrinking the coefficient estimates toward 0 for the Ridge, and exactly 0 for some covariates in the Lasso. The coefficients approach or are set to 0 for increasing values of the single hyperparameter in both of these models, lambda. This can aid with finding the most important covariates for predicting whether an application was accepted or not.

To begin with tree-based methods, bagging and random forests will be used. Both of these methods fit large numbers of trees on different bootstrap samples of the data. The difference between bagging and random forests is that bagging considers all of the variables at each split in the tree and random forests randomly selects from a subset of the variables. The idea with random forests is that introducing more randomness can decrease variance when using different test subsets. Bagging may produce many similar trees if some variables have strong explanatory power.

For ensemble methods, gradient boosting and eXtreme gradient boosting methods are popular. The general idea with ensemble methods is that many weak learners can produce a strong learner. These both still fit different trees but allow the model to learn from mistakes because each subsequent model is fit on the residual of the previous model. Many of these trees have a maximum depth of 1 to to reduce the complexity of each tree. XGboost includes an additional regularization term similar to Lasso and Ridge and performs faster than gradient boosting because it implements parallel processing.

To take a different route on statistical learning, an artificial neural network was created to predict the outcome on the application. This is classified as an eager learner where the model creates a classification algorithm without being provided a test subset. With this increased complexity, the model produced the highest accuracy rate. To test a more classical method, k-nearest neighbors was implemented but this lazy learner proved to work similar to the previous logistic regressions. Also, to test margin classifiers, support vector machines deemed to be very accurate and performed similar to the other tree-based methods.

The findings from these different algorithms will provide insight on the explanatory power of each variable and predict the outcome of the application. These various settings will learn which variables are the most significant and this can be used to generalize which have the most explanatory overall over different models. In the following section, I will discuss the background of the data.

2 Data and Summary Statistics

The data was retrieved from the "Applied Econometrics with R" package in R authored by Christian Kleiber and Achim Zeileis. The data is from a cross-section of 1300 applicants for a type of credit card and the dependent variable would be whether or not the person was accepted for the credit card. Since our dependent variable is a categorical variable, this model would be concerned with classification rather than regression. The independent variables in this dataset include the number of derogatory marks on a persons' credit report, the age of the applicant in years, yearly income measured in USD 10,000, the ratio of monthly credit card expenditure to yearly income, the average monthly credit card expenditure, whether or not the applicant is self-employed, the number of dependents under the applicant, the number of months the applicant has lived at their current address, the number of major credit cards the applicant held, and the number of active credit accounts. After experimentation with the data, the ratio of monthly credit card expenditure to yearly income was removed from the models due to near-perfect multicollinearity. Optimistically, a set of these variables will have strong explanatory power in predicting whether the application for the credit card was accepted. The selected model can be helpful in understanding which factors are associated with a financially responsible borrower so lenders can minimize their risk when accepting someone for a credit card.

To interpret the summary statistics, the median age was ~31 years old and the median income was ~$27,000. There is also strong variation in whether the applicant owns their house and more people are not self employed. In terms of derogatory reports and months at the current address for the applicant, most applicants had 0 and ~3 years based on the medians respectively. The other variables corresponding to major credit cards

held and number of active credit cards, most applicants held 1 major card and had 5 cards based on the median values. It is also important to note the presence of outliers in most of our variables which is skewing the mean towards unrepresentative values. In the following section, I will discuss the models used based on objective functions and accuracy rates.

3 Models

3.1 Logistic Regression

Logistic regressions are a type of classifier that models the probability that the dependent variable belongs to a class, in this case whether the applicant was accepted or not. This can be used for multiple class levels rather than a binary classification in this case. The log-odds ranges from (0,1). The output is the log-odds of the situation belonging to $Y = 1$ and is

The objective function is:

$$p(Y = 1|X) = p(X) = \frac{e^{\beta_0+\beta_1 x}}{1 + e^{\beta_0+\beta_1 x}}$$

To compute the log odds we can take the log of this function and get the following:

$$log(\frac{p(X)}{1 - p(X)}) = \beta_0 + \beta_1 x$$

3.1.1 Results

```
mean(glm.pred == CreditCard$card)
```

```
## [1] 0.8597422
```

Using the logistic regression, I obtained an ~86% accuracy rate. The variables were highly significant (1% level) were the number of derogatory reports, income, number of dependents, and the number of active cards. Interestingly, whether the applicant owned their home,whether they were self employed and the number of major cards held were also significant. Age and the number of months at the current address were both insignificant.

To discuss the magnitudes of the negative effects on the approval process, derogatory marks hurts the log-odds ratio of being approved the most, with being self-employed and the number of dependents following, respectively. The economic background behind the negative coefficient on self-employment is that may be a more "risky" borrower. There may be more variation in income for a self-employed individual depending on their occupation, compared to a person working at an established company. The number of dependents also correlates negatively with the log-odds ratio of being accepted because more dependents can mean a larger financial burden which in turn leads to more debt. It is expensive to raise children and the borrower may divert their financial focus towards the dependents rather than the credit card bill. In terms of positive factors on application approval, income, the number of active credit cards, and whether the person owned their home all indicate stability. More credit cards means that the applicant may need another credit line and that they can handle one credit line properly. Owning a home can be also be seen as financially stable or as an asset that the lender could remortgage in the case of needing to make a payment.

3.2 Ridge Regression

To begin with shrinkage methods, the Ridge regression shrinks coefficients towards 0 to reduce variance. This is done through including an additional shrinkage penalty alongside the residual sum of squares, which is eventually minimized. This shrinkage parameter is controlled by the hyperparameter λ, with higher values shrinking the estimates further toward zero but not exactly zero.

The objective function is:

$$\min_{\beta} \sum_{i=1}^{n} (y_i - \beta_0 - \sum_{j=1}^{p} \beta_j x_{i,j})^2 + \lambda \sum_{j=1}^{p} \beta_j^2$$

To test the model accurately, I split the data into a training and validation set, so the model is tested on data it has never seen before. The data is also now standardized because the regularization parameter is sensitive to the size of the coefficients.

3.2.1 Results

The hyperparameter, λ, is tuned using k-fold cross validation to minimize the binomial deviance, and selected within 1 standard error of the minimum lambda.

```
mean(ridge_pred == y_test)
```

```
## [1] 0.8956743
```

```
table(ridge_pred,y_test)
```

```
##           y_test
## ridge_pred  no yes
##        no   53   6
##        yes  35 299
```

This model increases the accuracy rate by ~3%. To interpret the size of the coefficients, the model is fit on the entire dataset. The results are similar where derogatory reports hurt the chances of being accepted the most but the magnitudes of the coefficient for major cards held increased signficantly. This may be due to the applicant having a higher credit score after owning previous cards, or it can show experience with the credit system. Still, owning a home seems to be the most positive factor towards being accepted for the card. Yearly income, average monthly expenditure, and number of active credit cards are similar in magnitude but these factors are not as strong as owning a home because these values vary more between individuals more than the categorical variable.

3.3 Lasso

Lasso is another shrinkage method similar to Ridge but it can set some coefficients to exactly zero, so this may help with variable selection as well. The objective function is still similar to Ridge with a tweak on the shrinkage penalty which allows the constraint region to be a square (ℓ^1 norm) to set coefficients to 0, rather than a circle with Ridge (ℓ^2 norm).

The objective function is:

$$\min_{\beta} \sum_{i=1}^{n} (y_i - \beta_0 - \sum_{j=1}^{p} \beta_j x_{i,j})^2 + \lambda \sum_{j=1}^{p} |\beta_j|$$

```
mean(lasso_pred == y_test)
```

```
## [1] 0.9618321
```

```
table(lasso_pred,y_test)
```

```
##           y_test
## lasso_pred  no yes
##        no   88  15
##        yes   0 290
```

3.2.1 Results

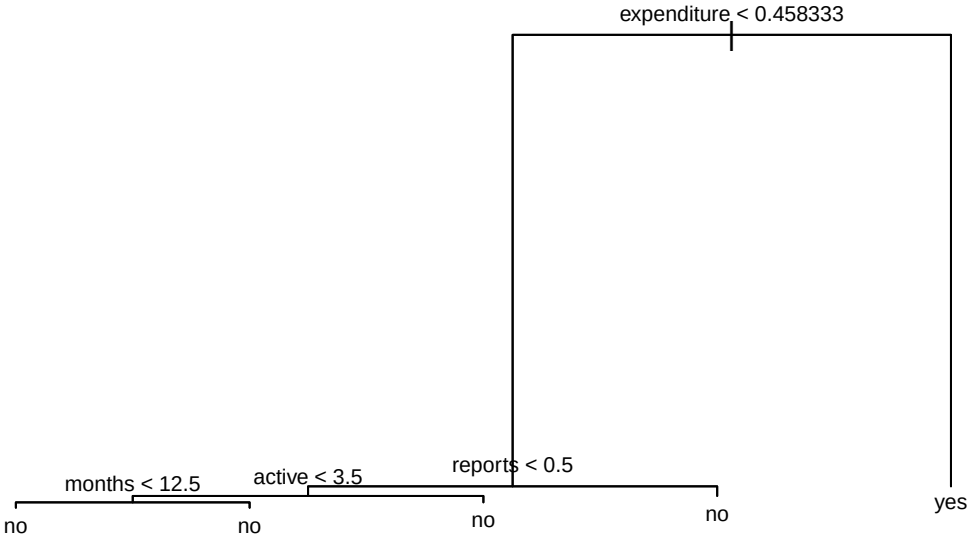
The Lasso offers a steep improvement over the logistic and Ridge regression with a 96% accuracy. Interpreting the coefficients, age and the number of months at the current address are very close to zero meaning they were not contributing enough to the variation in application approval. Now, average monthly expenditure and major cards held have similar coefficients to owning a home. I predicted average monthly expenditure to be a relevant predictor because lenders would want applicants that spend more, and when they cannot pay the full balance, the lender will make money off interest and this proved to be true in this model. Interestingly, the coefficient for self-employment has changed to positive. A possible reason is that people that are self-employed have complete control over the financial aspects of their occupation, meaning they can possibly pocket more money than they would by being an employee.

3.3 Classification Tree

To take a different approach on predicting the outcome of the credit card application, tree-based methods are very popular. They outline decisions the program takes to classify decisions, typically done through entropy. An example using this data is shown. The criteria for making a split is through the classification error rate, outlined as:

$$E = 1 - max_k(\hat{p}_{mk})(1 - \hat{p}_{mk})$$

where $\hat{p}_{mk}$ is the fraction of training observations in the mth region that belong to the kth class.



3.3.1 Results

```
mean(tree1_pred == test$card)
```

```
## [1] 0.9796438
```

```
table(tree1_pred,test$card)
```

```
##
## tree1_pred  no yes
##        no   88   8
##        yes   0 297
```

While the results may seem promising, the tree is very sensitive to different samples of the dataset. It is better to use averages over many trees and by using different samples of the dataset which will be done in bagging and random forests. To interpret the tree, if expenditure is below a certain threshold, we move to the left side of the tree, and then check the number of derogatory reports against a threshold and so on,

before predicting the applicant was declined. If the condition is satisfied, the tree predicts the outcome on the right branch, depending on what depth of the tree the decision is made in. If expenditure is over a certain threshold, the predicted outcome is the applicant was accepted for the card.

3.4 Bagging

Bagging or bootstap-aggregation involves making many trees over different bootstrap samples of our dataset. Bootstrapping is used to sample the data many times with replacement and with classification trees, the outcome is obtained over many trees. The number of variables tried at each split is equal to the number of variables in the dataset.

3.4.1 Results

```
mean(bagging_pred == test$card)
```

```
## [1] 0.9720102
```

```
table(bagging_pred,test$card)
```

```
##
## bagging_pred  no yes
##          no   85   8
##          yes   3 297
```

This model results with a slightly higher error rate after fitting 400 different trees on 400 different bootstrap samples of our dataset. The single decision tree is non-robust and varies with each different sample of our dataset, and may not have the level of prediction accuracy as bagging or random forests. Despite having virtually the same error rate for the single tree, I would conclude that the bagging model is more robust and provides better prediction accuracy since it is the average of 400 trees over different samples of the dataset.

The interpretation is very similar to the original classification tree, where expenditure is the most important variable to split the tree on initially. The other variables important in this model were the number of derogatory reports, active credit cards, and whether the person owns their home, for reasons mentioned previously.

3.5 Random Forest

Bagging is simply a case of random forest. Random forests are almost identical to bagging except the number of variables chosen at each a set is a random subset of variables. The number of variables to randomly select at each split is another hyperparameter that needs to be tuned in this model. If certain variables such as expenditure have strong explanatory power, the trees produced will be very similar to each other. The idea behind randomly choosing among variables at each split is that adding more randomness to the model can produce different situations and the average outcome over many trees may be more powerful. The optimal number of variables in this case was chosen to be two.

3.5.1 Results

```
mean(pred_rf_tuned == test$card)
```

```
## [1] 0.9720812
```

```
table(pred_rf_tuned,test$card)
```

```
##
## pred_rf_tuned  no yes
##           no   88  11
##           yes   0 295
```

In terms of accuracy rate, this model provides a very slight increase in accuracy from the bagging and a large differences between the initial logistic regression and Ridge regression. The model also slightly outperforms the Lasso regression.

The results of this model are similar to the bagging model but the variables such as major credit cards held and whether the person is self employed hold little importance in this model. This may be due to the overpowering effect of the average monthly expenditure where that explains most of the variation in the application process.

3.6 Gradient Boosting

A problem with bagging and random forests are that the trees do not seem learn from errors (misclassifications). Boosting is a popular ensemble method which aims to fit subsequent models on the errors of the previous models. The objective is to combine many weak learners into a strong learner, and the rate of this learning is λ. The hyperparameters of this model are the number of trees B, the shrinkage parameter/learning rate λ, and the number of splits in each tree d. With many trees, boosting models can overfit the data in the train set and will not perform well in the validation set. Concerning split, trees of depth one (one split) tend to work well because many of these simple trees can combine into a strong model. This model is tuned using grid-search and cross-validation for the hyperparemeters to avoid overfitting.

First, initialize $\hat{f(x)} = 0$ and $r_i = y_i$. The model works by updating $\hat{f(x)}$ with a shrunken tree $\lambda\hat{f^b}$ of d splits and then having the residual (error) updated for the number of trees B.

The final boosted model is:

$$\hat{f(x)} = \sum_{b=1}^{B} \lambda \hat{f^b(x)}$$

3.6.1 Results

```
mean(boost_pred1 == test$card)
```

```
## [1] 0.9796438
```

```
table(boost_pred1,test$card)
```

```
##
## boost_pred1  no yes
##         no   88   8
##         yes   0 297
```

The optimal tuning parameters are obtained after performing k-fold cross validation and using ROC as the accuracy metric.

This model increases by 1% to roughly ~98%, the learning from these trees proved to help predicting the outcome. Still, the results are very similar and the interpretations of the variables are virtually the same. Average monthly expenditure has the most explanatory power with the number of derogatory reports helping classify the applicants that were declined.

So far, the models have been increasing in accuracy rate with the gradient boosting algorithm outperforming all previous models. In terms of variable importance, average monthly expenditure seems to be the most important variable comparing across different models.

3.7 eXtreme Gradient Boosting

XGboost (eXtreme Gradient Boosting) is a further development on gradient boosting. It includes more hyperparameters such as γ which is the number of leaves (terminal nodes) and a regularization parameter λ on the vector of weights ω. The regularization parameter weights ω defined the minimum sum of weights needed in a child before further splitting.

The objective function is:

$$\sum_{i=1}^{n} l(y_i, \hat{y_i}^{t-1} + f_t(x_t)) + \Omega(f_t) = \sum_{i=1}^{n} l(y_i, \hat{y_i}^{t-1} + f_t(x_t)) + \gamma T + \frac{1}{2}\lambda \sum_{j=1}^{T} \omega_j^2$$

3.7.1 Results

```
mean(xgb_pred == y_test)
```

```
## [1] 0.9796438
```

```
table(xgb_pred,y_test)
```

```
##         y_test
## xgb_pred  no yes
##      no   88   8
##      yes   0 297
```

I received the same accuracy rate using XGboost, this may be due tuning the hyperparameters using "random" search as opposed to grid search to save computing time. Both of these models implement gradient boosting and that could be a reason they are learning from the data with the same approach.

The coefficients most important in this model are the average monthly expenditure, the number of derogatory reports, income, the number of active credit cards, and whether the applicant is self-employed. This model determined that months at the current address, whether the person owned their home, number of dependents, age, and major credit cards held were not important in predicting the outcome of the application. These results stem from the boosting algorithm using the variables that provided the largest reduction in loss, and ultimately the most valuable splits according to the regularization parameter λ on the sum of the instance weights.

3.8 Artificial Neural Network

Neural networks are the cutting-edge machine learning method used on this dataset and possibly the most interesting. The data goes through a network through different layers which contain neurons. The goal of this is to find the correct weights to connect neurons to give more importance to certain variables. This process is done through back-propagation where the weights are repeatedly adjusted through gradient descent, until the error rate cannot be further improved. Gradient descent is popular in machine learning where the the algorithm moves toward steep directions in order to find the local or global minimum of a function. This framework guiding the weights the network sets, can lead to powerful prediction power.

Since this is a classification problem, I used softmax as the activation function and cross-entropy(deviance) as a measure of fit. The choice behind softmax and not sigmoid is that I wanted the probabilities rather than just 0 or 1 as in sigmoid. In this case, there is no difference as it is binary classification and softmax is a special case of sigmoid for multi-class level classification.

Softmax objective function:

$$g_k(T) = \frac{e^{z^{(i)}}}{\sum_{j=0}^{k} e^{z_k^{(i)}}}$$

where

$$z = \sum_{i=0}^{m} w_i x_i$$

which is the weight vector and vector of the datapoints.

```
table(test_class,test_labels[,2])
```

```
##
## test_class   0   1
##          0  83   4
##          1   5 301
```

3.8.1 Results

The model is created with two hidden layers with 64 neurons each and using "softmax" as the activation function. First, the model is fit with a large number of epochs to visualize the loss and accuracy. The model will stop after 20 epochs once the validation loss increases. The accuracy obtained on the test set is 97.20% which is slightly lower than the boosting and XGboost model. This loss in accuracy may be justified by the tree-based ensemble methods simply learning more from the data than the complex artificial neural network. If there was more data, we might see that the ANN will outperform the gradient boosting methods but for this amount of data the differences between XGboost and neural network may not be sufficient. Still, the differences are minimal and the ANN outpeforms the other shrinkage methods and random forest/bagging models.

3.9 KNN

K-nearest neighbors is a traditional machine learning method for classification and regression. The only parameter is k, the number of neighbors to consider while assigning the data point to the most common class. This model was included just to test the performance of a non-parametric method in comparison to the more complex methods. The model will find the nearest K neighbors and count the neighbors in each group before assigning a class.

The objective function is:

$$Pr(Y = j|X = x_0) = \frac{1}{K} \sum_{x \in N_0} I(y_i = j)$$

3.9.1 Results

```
mean(knn_pred == knn_test$card)
```

```
## [1] 0.8625954
```

KNN performs more similar to the logistic regression and Ridge regression, compared all previous methods. This would be classified as a lazy learner where the model doesn't learn from the data but tries to simply classify observations based on distance.

3.10 Support Vector Machines (SVM)

Support vector machines are models that separate data using a linear or non-linear hyperplane, while maximizing the margin between the data points and the hyperplane. The support vectors in these models are the data points that are very close to the hyperplane that can shift the direction or position of the hyperplane. There is a cost function to be tuned which is associated for observations on the wrong side of the hyperplane.

The objective function is:

$$\max_{\beta_0, \beta_1, ..., \beta_p, M} M$$

subject to

$$\sum_{j=1}^{p} \beta_j^2 = 1$$

$$y_i(\beta_0 + \beta_1 x_{i1} + ... + \beta_p x_{ip}) > M(1 - \epsilon_i)$$

$$\epsilon_i > 0$$

$$\sum_{i=1}^{n} \epsilon_i < C$$

where M is the margin, C is the cost and ϵ_i is the slack variable which relaxes the constraint to allow observations on the wrong side of the margin.

```
mean(svm_pred == test$card)
```

```
## [1] 0.9695431
```

```
table(svm_pred,test$card)
```

```
##
## svm_pred  no yes
##      no   88  12
##      yes   0 294
```

Support vector machines perform slightly worse than the bagging/random forest, boosting/XGboost, and ANN but perform better than the Lasso regression and other models.

4 Conclusion

From the exploratory analysis on whether credit card applications were accepted, I predicted that the number of derogatory marks, age, income, average monthly expenditure, and number of active cards would have the most explanatory power. Interpreting the results across different models, initially, the findings from the logistic regression indicate that the number of derogatory reports, income, number of dependents, and the number of active cards were highly significant on the log-odds ratio of being accepted. Also, whether the applicant owned their home,whether they were self employed and the number of major cards held were also significant at the 5% level. Economically speaking, this makes sense for the number of derogatory reports to have the strongest negative effect because a lender would not want to give a credit card to a person with numerous missed payments. The negative relationship between number of dependents and being accepted is interesting because families with more children may bear more expenses and therefore may default. On the contrary, the number of major credit cards held seemed to have a strong positive effect on the log-odds of the application being accepted. This also makes sense because it indicates that that you have been accepted through the application process by major credit cards in the past which suggests you are a responsible borrower. Yearly income had the second strongest effect on the log-odds of being accepted and this follows intuition where the more money you make, the less likely you will default, assuming you have responsible spending practices. For the number of active credit cards held, if the applicant is looking to get a second or third credit card, they may be trying to spread out charges and trying to improve their credit score. In the lenders' perspective, having multiple cards is proof that you are a responsible borrower so you may be more likely to get accepted.

The shrinkage methods (Ridge/Lasso) also indicated that derogatory reports hurt the chances of an applicant being accepted. On the contrary, this model displayed that owning a home can strongly increase the probability of being accepted. This can be exemplified by the house being seen as "collateral" in the case of a default, despite whether or not this was mentioned when the applicant was accepted. Still, owning a home can be seen as an asset of wealth and shows financial stability. The Lasso model also found that average monthly expenditure and self-employment are both factors that increase the probability of being accepted, while also being more accurate than the Ridge model. Throughout the other tree-based models and ensemble methods, almost all of them agreed that average monthly expenditure is the most important factor, with active credit cards and income also having relative importance. In terms of accuracy rate, the gradient boosting model performed the best through cross-validation, even in comparison to the artificial neural network. This may be due to the nature of the data and even with >1000 observations, this may not be enough data for the neural network to outperform the boosting models. The neural network performed similar to the support vector machine and the random forest/bagging models.

This paper contributed to the literature analyzing credit card applications through various economic factors that most people will identify with. Based on the results of the models with the highest prediction accuracy, the higher the average monthly expenditure, the more likely the applicant is to be accepted, unless the applicant has even one derogatory report on their credit report. Other factors such as owning a home and number of active cards indicate economic stability. Other papers conduct risk analysis based off purely financial metrics and forecasting delinquent payments which is also useful for analyzing the stability of the borrower. This research can be further improved by more data on other factors to further generalize this data to the population and to demonstrate the power of cutting-edge machine learning algorithms such as artificial neural networks.

4 References

Awoyemi, J. O., Adetunmbi, A. O., & Oluwadare, S. A. (2017). Credit card fraud detection using machine learning techniques: A comparative analysis. *2017 International Conference on Computing Networking and Informatics (ICCNI)*, 1–9. https://doi.org/10.1109/iccni.2017.8123782

E. Aleskerov, B. Freisleben and B. Rao, "CARDWATCH: a neural network based database mining system for credit card fraud detection," *Proceedings of the IEEE/IAFE 1997 Computational Intelligence for Financial Engineering (CIFEr)*, New York City, NY, USA, 1997, pp. 220-226, doi: 10.1109/CIFER.1997.618940.

Khandani, A. E., Kim, A. J., & Lo, A. W. (2010). Consumer credit-risk models via machine-learning algorithms. *Journal of Banking & Finance*, 34(11), 2767–2787. https://doi.org/10.1016/j.jbankfin.2010.06.001

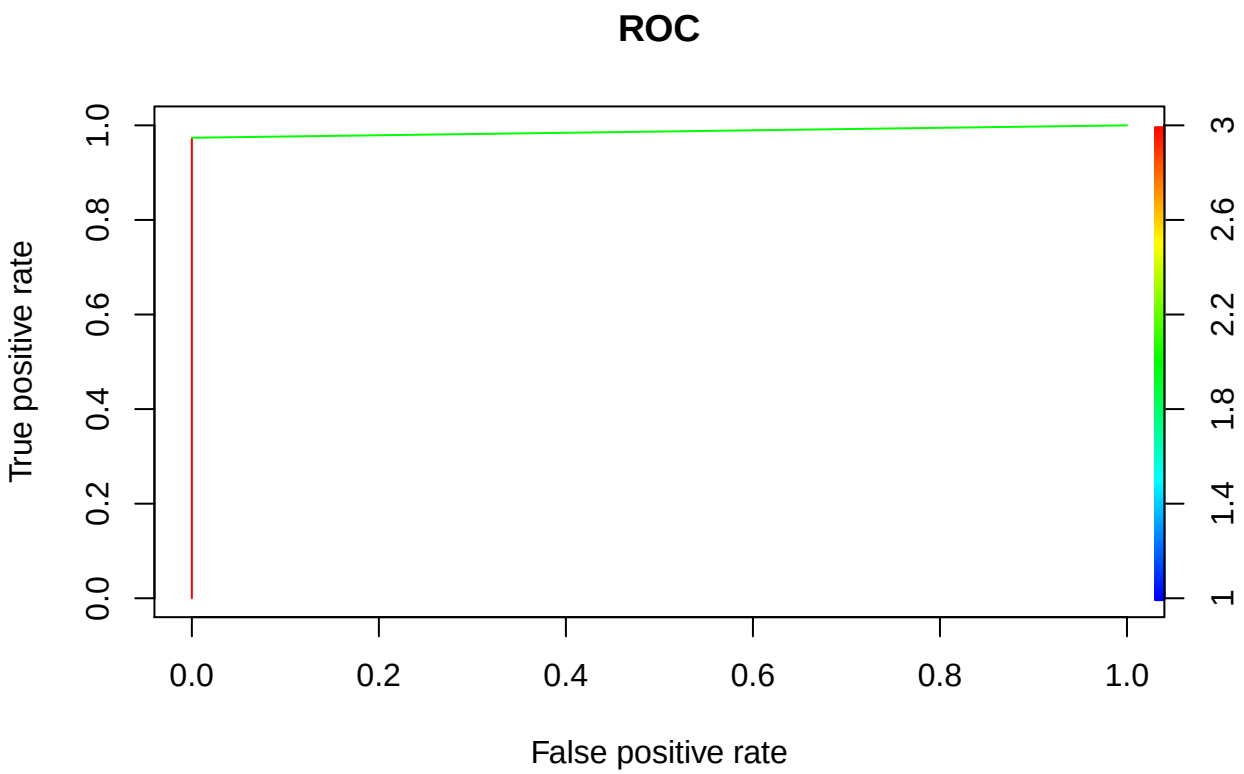
Kruppa, J., Schwarz, A., Arminger, G., & Ziegler, A. (2013). Consumer credit risk: Individual probability estimates using machine learning. *Expert Systems with Applications*, 40(13), 5125–5131. https://doi.org/10.1016/j.eswa.2013.03.019

Tuyls, Karl & Maes, Sam & Vanschoenwinkel, B.. (2015). Machine Learning Techniques for Fraud Detection. https://www.researchgate.net/publication/254198382_Machine_Learning_Techniques_for_Fraud_Detection

5 Appendix

5.1 ROC/AUC from XGBoost

Graphic representation of the best model can be displayed using a ROC curve.



The area under the curve (AUC) is as follows:

```
## [1] 0.9868852
```