Loan Application Prediction Model

University of Texas at Dallas
BUAN 6356 - Group 1
Om Sharan, Shresth Abrol, Hyung “Jay” Kim, Trevor Baum

Executive Summary

This report will detail the findings of four MS Business Analytics candidates at The University of Texas at Dallas. Four predictive methods are used (Decision Trees & Random Forest, Logit, Neural Network and various combinations of the three) to create several business intelligence models that predict whether a loan applicant will be approved by the bank or not based on attributes compiled in the loan application. The combination of these attributes are the basis of the business intelligence models. We are able to find which attributes are most important to the approval of a business model which lead to key managerial implications and translate to real world business benefit. Our goal was to create a model that would be useful in the automation of the loan application decision process and create value for loan issuing banks and the managers that work at these banks.

Project Motivation and Background

Our group wanted to find a project that we would be able to leverage on a resume towards an internship or job in banking or a related field like corporate finance. This project will allow us to have experience in building models that are related to banking and finance. In our search for a dataset that would fit our desires, we came across a hackathon competition to create the best model for predicting whether the loan applicant's application would be approved:

"Dream Housing Finance company deals in all home loans. They have a presence across all urban, semi urban and rural areas. Customer first apply for a home loan after that company validates the customer eligibility for loan.

Company wants to automate the loan eligibility process (real time) based on customer detail provided while filling online application form. These details are Gender, Marital Status, Education, Number of Dependents, Income, Loan Amount, Credit History and others. To automate this process, they have given a problem to identify the customers segments, those are eligible for loan amount so that they can specifically target these customers. Here they have provided a partial data set."

Before this project, some of our group had experience in finance but for the most part, we were very unfamiliar with the loan process but wanted to learn. This project has allowed us to expand on our skills learned in the classroom of data visualization, manipulation, transformation, imputation and modeling using a real world scenario of loan applications.

Data Description

In our data, we have inputs on hundreds of loan applications with 12 attributes ranging from familial data points like number of dependents to data on what type of property the loan is for; like semiurban, rural and urban. We also have attributes about the loan applicant such as their education level, number of dependents, if they are self-employed, their income, and their co-applicant's income. Lastly, we also have the loan application's attributes such as the loan amount request, the loan terms and whether or not the loan was approved.

Data Cleaning

In the dataset provided, we had 12 variables and 614 observation. Of the 12 variables, 2 were ordered categorical variables, 6 were binary variables, 1 was unordered categorical variables and 3 were numeric variables. Of the 614 observations, 85 observations had null values and 49 observations were blank values.

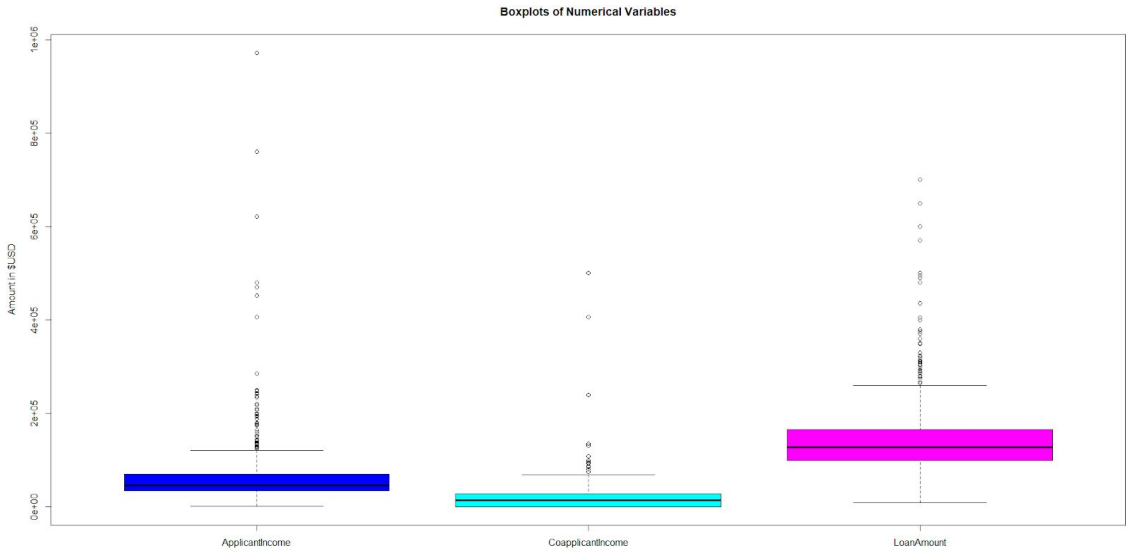
First, we set all the blank values as null values. Then, set all the null values of the two ordered categorical variables (LoanAmount and Loan_Amount_Term) as its median because all of its values were whole numbers.

For binary variables, we used weighted average to fill the null values. For the Gender variable, we had a Female to Male ratio of 4:1. With 13 null values, we set 10 of them as Females and the remaining 3 as Males. For the Married variable, we had a Not_Married to Married ratio of 2:3. With 3 null values, we set 2 of them as Not_Married and 1 of them as Married. For the Self_Employed variable, we had a self employed to not self employed ratio of 9:1. With 32 null values, we set 30 as self employed and 2 as not self employed. For the Credit_History variable, we had an approved to not approved ratio of 21:4. With 50 null values, we set 42 of them as approved and the remaining 8 as not approved. For the Dependents variable, we had 4 levels: 0, 1, 2 and 3+ with a ratio of 7:2:2:1, respectively. With 15 null values, we set 10 of them as 0 dependent, 2 as 1 dependent, 2 as 2 dependent and 1 as 3+ dependent.

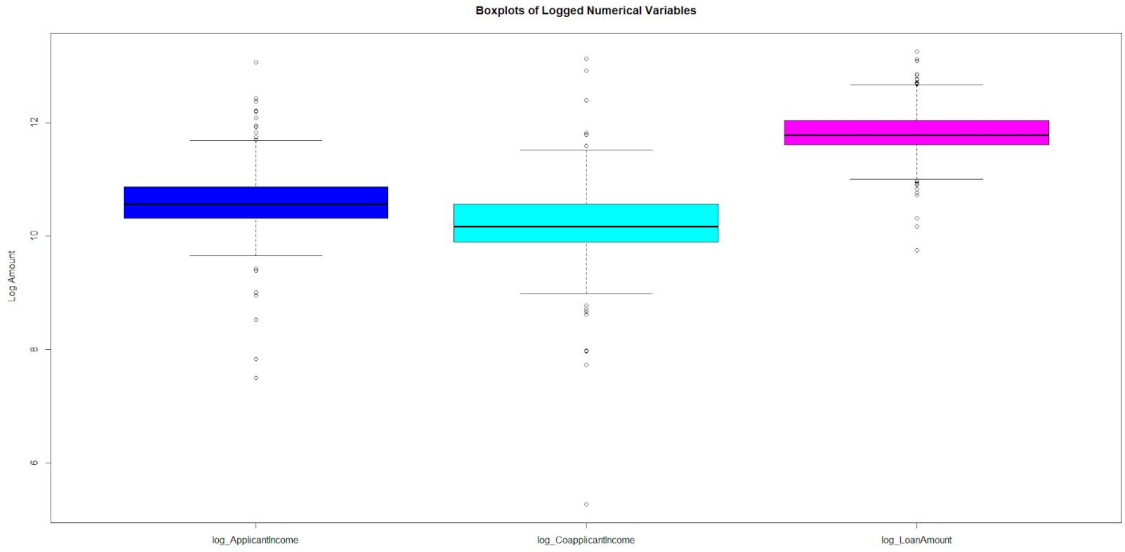
After filling the null values, we converted Gender, Married, Dependents and Self_Employed as a binary variable.

Property_Area variable had 3 unordered categorical factors: Rural, Semiurban and Urban. Hence, we decided to use dummy variables and omit the Urban column so it does not create redundancy.

LoanAmount variable was initially in 1000s, so we decided to multiply the entire column by 1000 to get an exact loan amount. ApplicantIncome and CoapplicantIncome were in amounts per month, hence we multiplied both the columns by 12 to also get an exact applicant and co-applicant income. For these three numeric variables, we plotted a box-plot to find outliers and the skewness. Below is the given box-plot:



Here, we can see that all three of the variables are extremely right skewed and have a lot of outliers. To solve this issue, we took the log of these numeric variables. Below is the box-plot of ApplicantIncome, CoapplicantIncome and LoanAmount after we took the log:



After the log, we can see the data gets less skewed and becomes more normally distributed.

Variable Creation

We created two variables Monthly_est (monthly estimates: which is the estimated payments per month) and Monthly_ratio (monthly ratio: is an indicator of the ability to pay off the payments).

Monthly_est = LoanAmount * (0.05*(1+0.05) ^ Loan_Amount_Term)

Monthly_ratio = (Monthly_est + 1081 * Dependents_new) / (ApplicantIncome + CoapplicantIncome)

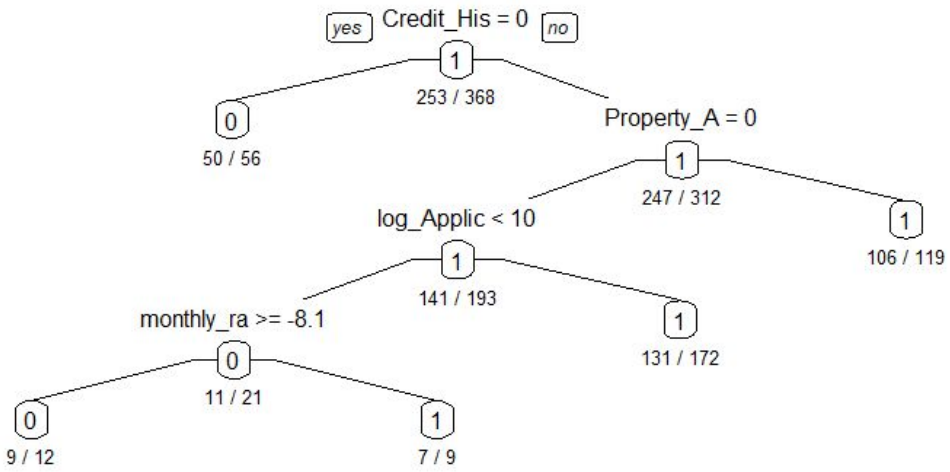
Creating Training & Validation Datasets

The dataset was split into a training and validation dataset which allowed for the various business intelligence models to be configured and tested using the data. The subsets were created by randomly choosing rows from the dataset. 60% of the dataset was assigned to the training dataset and 40% to the validation dataset.

Decision Tree & Random Forest

One of the business intelligence techniques used were decision trees as they are excellent for supervised learning and classification. As loan applications need to be classified as being approved or not based on various categorical variables, a decision tree would be perfect to use for prediction. Furthermore, with their ease of interpretation, they are useful in providing managerial implications.

The decision tree classifies if the loan should be approved or not. As determined computationally, the best decision tree has 5 leaves with the credit history being the most important when run on the training dataset. The credit history is most important because it is the first decision node.

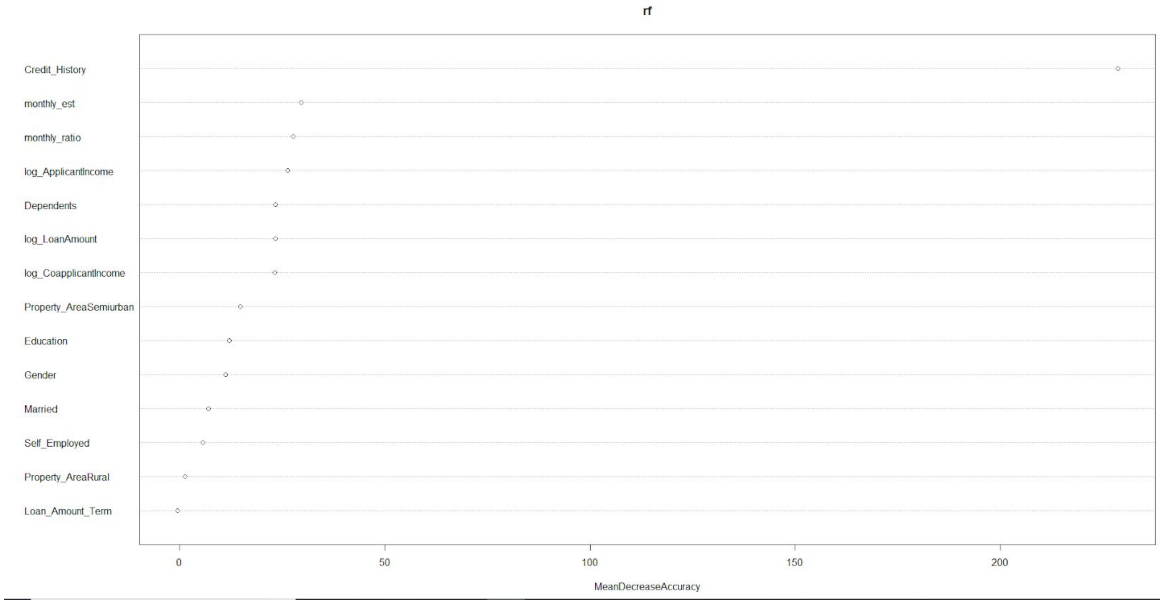


When predicting the training data, the decision tree was able to predict the loan status with 83.7% accuracy. The 95% confidence interval was 79.5% to 87.3%. It was equally accurate at predicting approved and declined applications as it had a positive prediction accuracy of 83.1% and negative prediction accuracy of 83.9%. It had a No Information Rate of 68.75% indicating that the 83.7% accuracy was significantly higher and provides value.

When using this tree on the validation dataset, the accuracy dropped to 78.9%. The No Information Rate was 68.4% indicating that this decision tree model did provide value again. This time the negative prediction accuracy dropped slightly to 78.5% while the positive prediction accuracy increased to 86.5%.

To improve the prediction accuracy, a Random Forest was created with 10,000 trees. The node size used was 7 because this was half the total number of variables available.

Based on this Random Forest, the Credit History was by far the most important variable as it had a Mean Decrease Accuracy greater than 200. The monthly payment estimate, monthly ratio and log Applicant income were also important variables.



When using this random forest on the validation dataset, the random forest was accurate at predicting loan approvals with 81.7% accuracy. This is an increase of 2.8% from the accuracy of the single decision tree. The random forest did incredibly well at predicting when loans should be declined as it accurately predicted 33/34 (97.1%) of these applications correctly. It predicted approved loans with an accuracy of 79.2% accuracy which is 0.7% better than the single decision tree.

Logit Model

A logit model was also used to try and classify loan applications as approved or declined. Initially all variables were included in the logit model. In this model, credit history, dependents, property_areaSemi-urban, logApplicantIncome, log CoapplicantIncome and the monthly ratio were statistically significant.

As there were variables that were not statistically significant, a variable selection process was performed. Using backward selection, the model with the lowest AIC value was chosen.

The best performing model included only the credit history, property area semi-urban and log co-applicant income. Credit history and Property Area Semi-urban were statistically significant at the 95% confidence level while the log(Co-applicant Income) was significant at the 90% confidence level.

Loan_Status ~ 3.56*Credit_History + 1.06*Property_AreaSemiUrban + 0.03*log(Co-applicant Income) -2.74

This model indicates that if an applicant meets the credit history requirements, they have a 73% chance of getting approved for their loan even if they don't have a co-applicant or a semi-urban property. An applicant that meets the credit history requirements and also has semi-urban property has an 88.7% chance of being approved even if they don't have a co-applicant.

Again, the credit history proves to be the most significant variable as it has the highest coefficient. If an applicant did not meet the credit history requirement, an application with semi-urban property and $100,000 in co-applicant income would still only give the application a 20.2% chance of being approved.

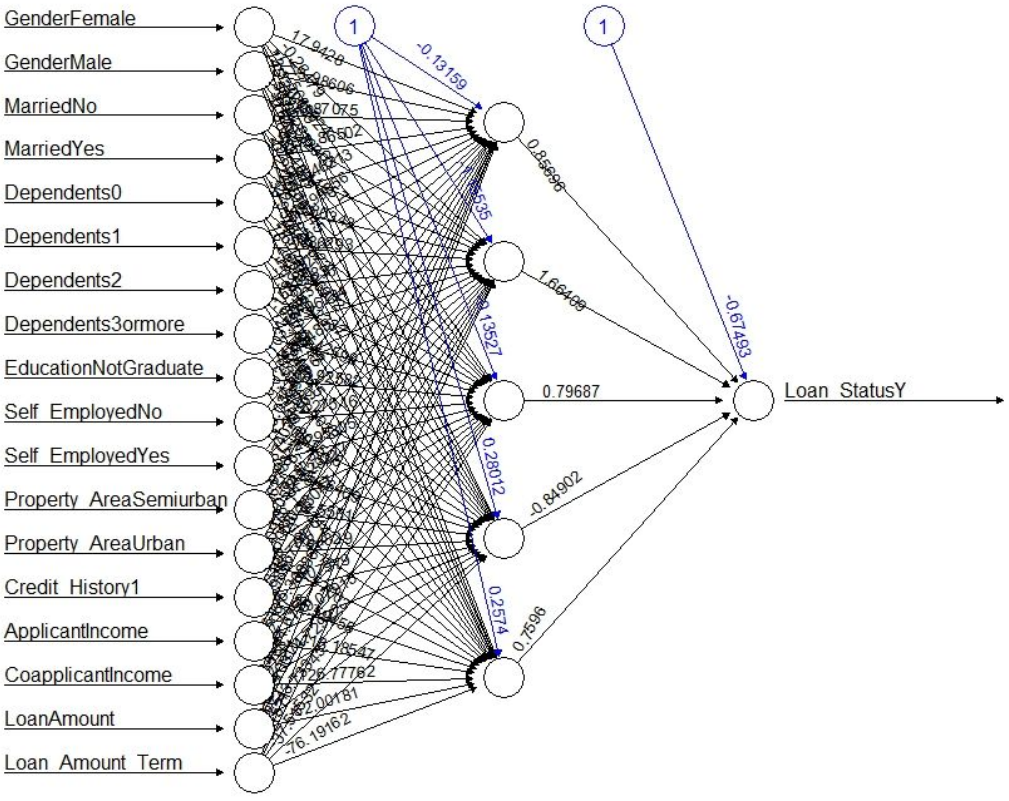
When using the logit model on the validation dataset, the model had an overall accuracy of 80.0%. It was 97% accurate at predicting when loans should be declined and 78.8% accurate at predicting when loans should be approved.

Neural Network

Neural Networks, also known as Artificial Neural Networks (ANN), are a form of multiprocessor computer system with simple processing elements, which transform inputs into outputs. Neural networks are a great supervised learning tool for computing fast arithmetic and functioning accurately as what the programmer programs it to do. A neural network is often called a black box in the sense that analyzing its structure will not provide you any insights while it can approximate any functions. However, considering the purpose of the project, creating a model that predicts whether or not an applicant will be approved for the loan, neural networks can grant a powerful classification tool for improving the loan application process, which is the most important for building the automated system. Also, a neural network is a great fit for our dataset since the volume of the provided data is relatively small, which means the process does not require long training time.

To create a neural network in R, we used the neuralnet package by Marvin N. Weight. This package requires an all numeric input data frame/matrix, so we transformed all the factor variables: Gender, Married, Dependents, Education, Self-Employed, Property_Area, Credit_History and Loan_Status. To make these variables favorable, by using model.matrix() function, we created a design matrix that allows expanding factors to a set of dummy variables.

Once all of the variables were determined, we ran the neuralnet() function to create a neural network model for the loan prediction. After training the model, we predicted the probability score using the compute() function and used the ifelse() function to convert probabilities into binary classes to indicate the result of predictions. In other words, if one instance has a higher probability than 0.5, the model will predict that the applicant will be approved for the loan. Using resilient backpropagation algorithms recommended by the package and including 5 hidden nodes in the model, the model performed an accuracy of 90.6%. Building a neural network was a success in showing both the advantages and disadvantages of it; the prediction accuracy is comparable with other statistical, but it only provides a non-intuitive model which makes challenging to interpret the results of it.




```
> summary(loan.df$Result)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
 0.0000  1.0000  1.0000  0.9062  1.0000  1.0000
```

Combined Model

Finally, a combined model was considered which includes the random forest, logit model, and neural networks. This would provide information on how often each model's predictions matched with others as well as generate insights on how all three models can be used together to improve approval accuracy.

On the validation dataset, all three models had matching predictions on 201/241 (83.4%). When all three did not match: the logit and random forest models matched 37 times, the logit and neural network matched twice, and the random forest and neural network matched once.

	matching	nn_pred	n	accuracy_logit	accuracy_rf	accuracy_nn
	<chr>	<dbl>	<int>	<dbl>	<dbl>	<dbl>
1	all	0	32	1	1	1
2	all	1	169	0.929	0.929	0.929
3	logit&nn	1	2	1	0	1
4	logit&rf	0	36	0.139	0.139	0.861
5	logit&rf	1	1	0	0	1
6	rf&nn	0	1	0	1	1

The table below breaks down the accuracy by the model match and predicted outcome. All three models predicted a declined application 32 times and was 100% accurate. All three models predicted an approved application 169 times and was 92.9% accurate.

The two instances when only the logit and neural network models matched, both times the matching models predicted an approved application and were correct both times. The one instance when only the random forest and neural network models matched, the matching models predicted a declined application and were correct in their prediction.

There were 37 instances when the logit and random forest agreed in their prediction, but the neural network did not. When the neural network predicted a declined application and contradicted the other two models, it was correct 31/36 (86.1%) of the time. The one time the neural network predicted an approved application while the two believed it would be rejected, the neural network again was right.

If the rule was to take the prediction of the from the highest number of matching models, this model would have an accuracy of 198/241 (82.1%). However, if the rule is

just to take prediction from the neural network, the accuracy is 224/241 (92.9%). Therefore, this combined model does not provide any lift over the neural network model.

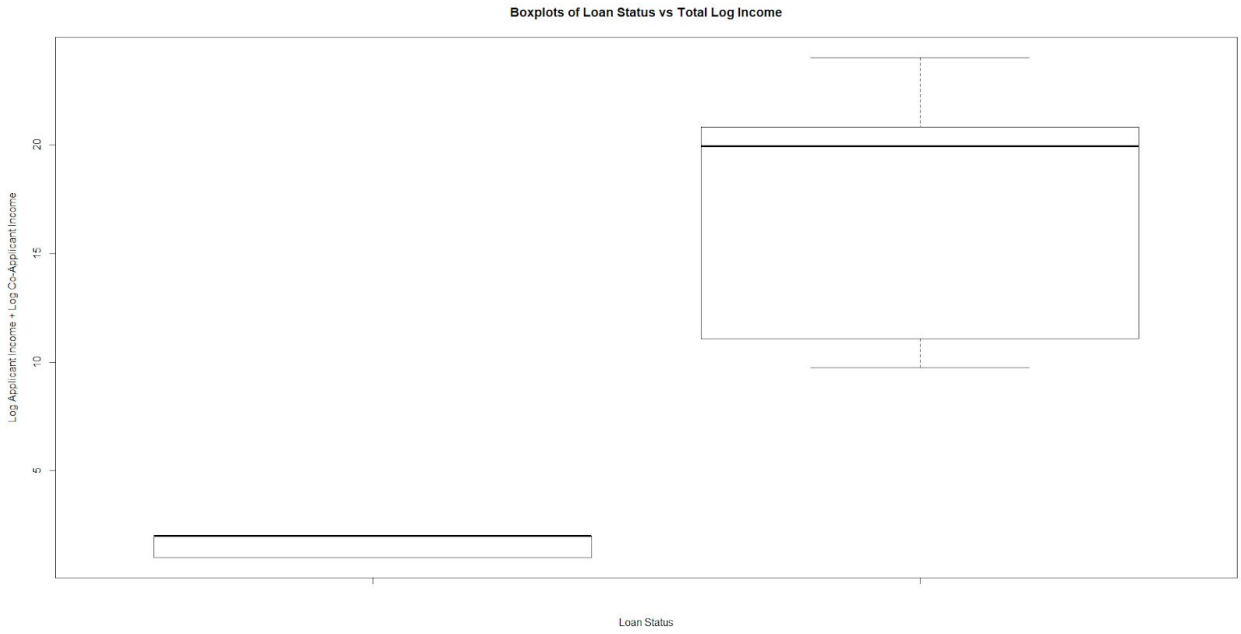
The combined model can still be useful though from a managerial standpoint. The neural network correctly predicted declined applications with 64/69 (92.8%) of the time. The five occurrences when the neural network inaccurately predicted an approved loan as declined occurred when the logit and random forest opposed the neural network. When the neural network matched with at least one other model at predicting a declined application, they were 100% accurate. This can help the automation process as it would automatically decline an applicant if the neural network predicted a decline along with another model. Then the system can flag applications where the neural network predicting a declined application does not match with the other two for review. This reduces the number of applications that need to be reviewed.

Findings

In our initial findings concluded that for this data set, the neural network model was the single best model from an accuracy perspective with an accuracy rate of just over 90% but this model is confusing in that it is a black box and it is not very user friendly or useful for real world application other than accepting inputs and providing outputs.

From a perspective that values actionable insights for management, the combined (RF, Logit, Neural Network) model is the most useful and effective model. When all three models agree on the loan decision, there is a 6.8% chance of a type 2 error and a 0% chance of a Type 1 error. This can be used to automate the loan decisioning process and deny unqualified applicants quickly.

Unrelated to the models, we also found that there was a unique relationship between the log of the total income of the applicants and whether they were approved for the loan. If the log of the total income was above 10, the loan status decision was yes. If the log of the total applicant income was below 2.5 then the loan status decision was sure to be no. These findings are shown in the box plot below:



Conclusions & Managerial Implications

All semester, it has been stressed in class that it is not enough to only understand the processes and meanings behind the predictive models. We, as business analysts, must also be able to apply real and actionable insights to the models and their results. These models can provide a great deal of value for managers of banks because they are able to reduce loan officer staff and save on payroll expense, improve the bank's "money creation" efficiency, and increase customer satisfaction rates.

Banks are able to saves costs on staffing of loan officers. With much of the process becoming automated, not as many loan officers will be needed and payroll expense will decrease. This also reduces the risk that banks take on because it reduces some of the human element that is involved in the process and makes decisions based on historical data.

The banks "money creation" ability will increase with the automation of the the loan decisioning cycle because the process will become faster and the velocity of money will increase. With these models, a bank will be able to turn its customers savings accounts into loans which are dollars that are circulating in the economy.

Customers will have increased satisfaction due to the reduction of the loan decisioning process. The customers will not have to wait as long to receive a decision

on their loan application and they will be more satisfied with the timeliness of a decision because of the automation.

Overall, the models provide lift and value for managers in a way that they are now able to save money, increase efficiency in their bank and improve their overall customer satisfaction.

References

Neural Networks in R Tutorial. Retrieved from http://www.learnbymarketing.com/tutorials/neural-networks-in-r-tutorial
Neural Network Models in R. Retrieved from https://www.datacamp.com/community/tutorials/neural-network-models-r
Comp.ai.neural-nets FAQ. Retrieved from http://www.faqs.org/faqs/ai-faq/neural-nets
Package ‘neuralnet’. Retrieved from https://cran.r-project.org/web/packages/neuralnet/neuralnet.pdf
Mortgages Calculator. Retrieved from https://www.nytimes.com/real-estate/mortgage-calculator