December 2022

GIVE ME SOME CREDIT DATASET

MACHINE LEARNING PHASE 3

Prepared by:
Roli Arah
Natalia Vargas
Dalia Ali

TABLE OF CONTENT

1. Introduction

Debt risk is the chance of a loss due to a borrower's inability to make loan payments or fulfill contractual commitments. For phase two, we developed two models, ‘Logistic Regression’ and’ Classification Tree’, to assess the risk that a prospective borrower will fall behind on their financial obligations during the following two years. To precisely anticipate which borrower would fail and who will be a solid opportunity to invest for the bank, we have developed a more precise neural network model. To make better predictions, we used an ensemble learning approach called the ‘Stacking Model’, which process is addressed and explained later in the report.

2. Data Cleaning

2.1. Outliers

Bring about a lot of variability in our data set. So it is an important step to tactfully handle them to build a more effective model. The columns with the highest variability in our data set are Revolving Utilization Of Unsecured Lines, Debt ratio, and Monthly Income. Outliers are particularly obvious with the Revolving Utilization Of Unsecured Lines and Debt ratio which are both ratios but have maximum values of 50,708 and 329,664 respectively.

	Revolving Utilization Of Unsecured Lines	Debt Ratio	Monthly Income
MEAN	6.1	354.4	6675.1
Standard Deviation	249.755	2037.8	14384.674
Minimum Value	0	0	0
99% percentile	1.093	4979.28	25000
Maximum Value	50708	329664	3008750

Table 1: outliers in Revolving Utilization Of Unsecured Lines, Debt ratio, and Monthly Income

The table above gives us the details of these three columns including the mean value, the standard deviation, the minimum value, the 99th percentile value, and the maximum value. The 99th percentile shows where 99% of the data set in that column lies. For the Revolving Utilization Of Unsecured Lines column, 99% of the values are 1.093 and below, for the debt ratio this figure is about 5,000 and below, and for monthly income, it is 25,000 and below. On the basis of this, we capped these 3 columns at the 99th percentile figure since there is a huge disparity between this figure and the maximum value in these columns. We would therefore be taking out values that exceed these thresholds so that they reflect as missing values as they do not seem real. We then proceeded to fill in these missing values through the imputation process outlined next.

2.2. Imputation

Our model does contain missing values and merely omitting them would affect the functionality of our model. Out of 11 variables, the variables ‘Monthly Income’ and ‘Number of Dependents’ contained a total of 33,655 (29,731 rows) missing values, which represents 22% of our data set. Hence, we want to predict these missing values based on the known values in the data set. This process is referred to as ‘imputation’.

In trying to fill in missing values, we considered filling all of them with the mean of all the known entries in that column or bootstrapping by randomly sampling from the known entries to fill in missing values. The first method is the easiest but may not be as effective as bootstrapping since the same mean value is filled in for every missing value and so distorts the distribution of the variable. Bootstrapping is also relatively straightforward but its strength is that it preserves the distribution of the variable by randomly sampling known values. We would be imputing with this method.

2.3. Standardization

Standardization is a process that helps put different variables on the same scale. It ensures that no variable dominates the other and it helps for easier interpretation and better model performance. After the standardization process, the mean of the variable becomes zero and all entries become z scores which are distributed around this mean. The z scores represent the number of standard deviations above or below this mean. Given that we have a lot of numerical predictors on different scales in our data set we did consider it to be an important step. Neural networks in their processing are sensitive to the size and range of variables so this step is important to ensure no variable dominates the other. (See Appendix 1)

2.4. Model performance after cleaning

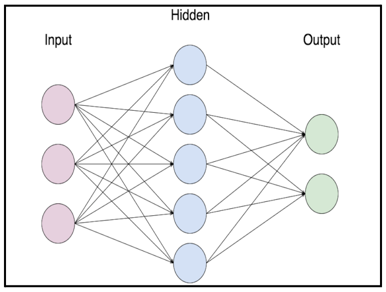
Model	**Error rate before cleaning**	**Error rate after cleaning**	
Logistic Regression	**0.0675**	**0.066**	⇩
Classification Tree	**0.066**	**0.0648**	⇩
Neural Network	**0.068**	**0.063**	⇩

Table 2: Model performance after cleaning

The table above shows the error rate of all the built models before and after the data cleaning process. The error rate which is the percentage of wrong predictions reduced in every model. This highlights the importance of data cleaning

3. Neural Networks Model

As the name implies, ***neural networks*** are constructed using brain neurons as a model. They unravel and deconstruct incredibly complicated relationships using artificial intelligence. The neural network consists of node layers, which include an input layer, one or more hidden layers, and an output layer. Each node, or artificial neuron, is connected to others and has a weight and threshold that go along with it. Any node whose output exceeds the defined threshold value is activated and provides data to the network's uppermost layer. Otherwise, no data is transmitted to the network's next tier. An example of a Neural Networks application is the Google self-driving car, which is trained to recognize a dog, a truck classically, or a vehicle. Although the neural network's structure and dynamics would be less apparent and hard to interpret, it nevertheless offers the most accurate predictions (higher accuracy rate and lower error rate).



In neural nets, a hidden layer sits between the algorithm's input and output, where the function gives the weights of the inputs and sends them into the activation function as the outcome. In this model, the default activation function used is the ReLU activation function which is widely used and is the default choice as it yields better results and is easy to compute (Appendix 3). Regarding the ReLU function[1], If

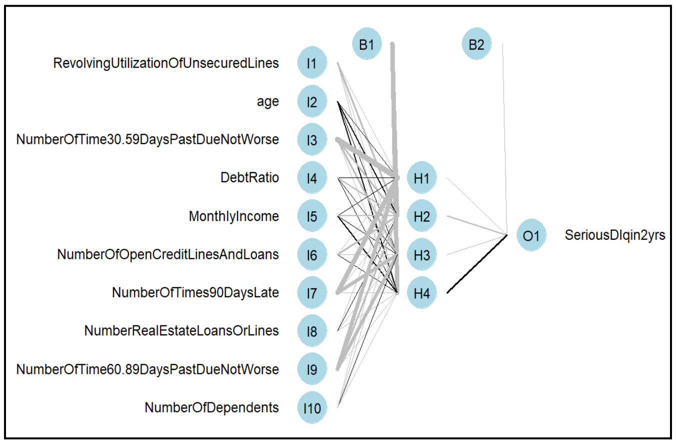
[1] the ReLU function formula f(z)= max(0,z)

the input is positive, the activation function will output the input directly; if it is negative, it will output zero.

3.1.Model

The goal is to predict if the potential borrower will experience financial delinquency in the next 2 years. The data is split into a training set and a test set. While the test set evaluates the model's effectiveness, the training set is utilized to determine the relationship between the dependent and independent variables. 60% of a dataset is used as the training set. Using random sampling, the data are divided into training and test sets. Using the sample () function in R, we sample randomly. To preserve consistency and provide the same random sample each time, we utilized set.seed(1234).

Our data was fitted with a neural network with four 4 nodes. For the analysis, we used a *neural net* library. Then, we visualized the neural net with values for each variable as shown in figure 3. The gray lines represent negative weights, whereas the black lines represent positive weights. The weight's relative magnitude to the other weights determines the line's thickness. The dependent variable is displayed in the far right layer (named *"SeriousDlqin2yrs"*), while each one of the ten random variables is displayed throughout the first layer (named I1 through I10). The size parameter in the *net* function was used to specify the hidden layer, which is identified as H1 through H4. Similar to the slope terms in linear regression, bias layers B1 and B2 add constantly to the nodes.


B1
B2
RevolvingUtilizationOfUnsecuredLines
I1
age
I2
NumberOfTime30.59DaysPastDueNotWorse
I3
DebtRatio
I4
MonthlyIncome
I5
NumberOfOpenCreditLinesAndLoans
I6
NumberOfTimes90DaysLate
I7
NumberRealEstateLoansOrLines
I8
NumberOfTime60.89DaysPastDueNotWorse
I9
NumberOfDependents
I10
H1
H2
H3
H4
O1
SeriousDlqin2yrs

3.2. Performance Measures

As we mentioned in Phases 1 and 2, There is a significant disparity between the count of defaulters and non-defaulters across the entire dataset. For example, 7% of customers default on their obligations, compared to 93% who are non-defaulters. Thus, wrongly assuming that the person will not face financial obligation (false negative - we predicted non-defaulter but observed defaulter) is higher than the opposite. Hence, focusing on the customers we wrongly predicted as non-defaulters will reduce the risk the bank might face; and our current model is missing 3370 defaulters while detecting 597 defaulters. To accurately measure the model's performance we created a model confusion matrix and we are using four measures: accuracy, precision, sensitivity, and specificity.

	Observations	
Neural Net Predictions	0 (non-defaulter)	1(defaulter)
0 (non-defaulter)	55618	***3370***
1 (defaulter)	*415*	597

Table 3: Neural Net model confusion matrix

Measure	Description	result
accuracy	Percentage of correct predictions for the test data. In simple words, the number of correct predictions from the total number of predictions.	**~93%**
precision	Ratio of true positives to the total predicted positives. For instance, how many of the individuals we predicted as defaulters are truly defaulters?	**~59%**
Specificity	Ratio of true negatives to total negatives in the data. For instance, How many people who are non-defaulters did we accurately predict?	**~99.1%**
Sensitivity	Ratio of true positives to total (actual) positives in the data. For instance, Of all the customers who are defaulters, how many of them did we properly predict?	**~15%**

Table 4: Neural Net model performance measurement

By using Neural Net to predict if the borrower will face financial obligations during the following two years, our accuracy percentage for predicting is ~93.3% which means that our model will fail to predict only ~6.4% which is a lower error rate compared to logistic regression and classification tree models used in phase 2.

4. Stacking

4.1. Construction

A 'stacked' model combines individual models to overcome their weaknesses, with the objective of improving and/or stabilizing predictions for obtaining a better model performance. In order to obtain better predictions and results for our previously presented *neural nets* model, we decided to create a 'stacked' model that uses a *tree classification* model and a *logistic regression* model as 'helper models', with a *neural nets model* as the 'manager model'. This means that the stacked model uses the manager model to take the best of every helper model by using their strengths and inputs, with the purpose of making prediction more efficient.

After making predictions for each helper model, we built the *neural nets* manager model, which was allowed to visualize and take both original inputs and predictions of the helper models as additional inputs. Then, the manager model was able to choose which model to trust. By doing this, new informative columns were created and added to the dataset that helped predict the target, allowing us to get new and more precise predictions. It is important to mention that for the stacked neural nets model, the dataset went through a standardization process, as explained previously for the model of the neural net, in order to keep accuracy along the results.

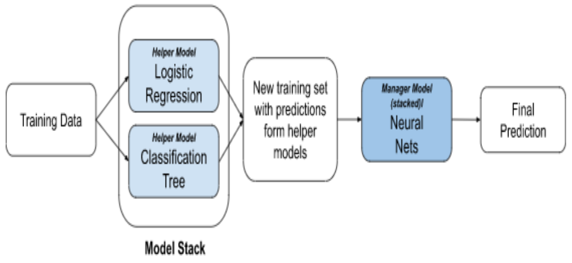


Figure 4: Stacking process[2]

4.2.Interpretation of results

After the neural nets stacked model was built and predictions were obtained, we decided to obtain specificity and sensitivity in order to test the performance of the new stacked model and compare it with the previously made neural nets model. Also, we obtained the error rates for each model. The purpose of this was to see if the stacked model was able to perform better predictions than the neural net model.

Error Rates		
Benchmark	Neural Nets	Stacked Neural Nets
0.067	0.064	0.063

Table 5: Error Rates

As shown in the table, the error rates of the neural net's initial model and for the stacked neural nets model seem relatively similar and low; however, it is always important to have a point of comparison to make proper conclusions about the model performance. In this case, between both models, the stacked performed a lower error rate by 0.001. The stacked model and the neural net models perform slightly better than the benchmark.

Sensitivity		Specificity	
Neural Nets	Stacked Neural Nets	Neural Nets	Stacked Neural Nets
0.150	0.153	0.991	0.997

Table 6: Sensitivity and Specificity

Although both stacked and unstacked neural nets models performed really low sensitivity indicators, the stacked model showed to perform better; as consequence, we can say that around 15% of the time the stacked model predicts correctly when someone will experience financial distress in the next two years, which is around 2% more than the non-stacked model. For the percentage of times that the models correctly predict when a client will not experience in the next two years, both specificity indicators performed really well but on this occasion, the stacked model performed better by just around 1%. The improved neural net model correctly predicts when someone will not face a financial crisis 99.7% of the time. The number of false negatives that both models predict is concerning because it means that the main purpose of predicting if a client will face financial distress within the next two years is not being addressed properly and around 85% of those clients that are going to face the crisis are taken into consideration as if they were not.

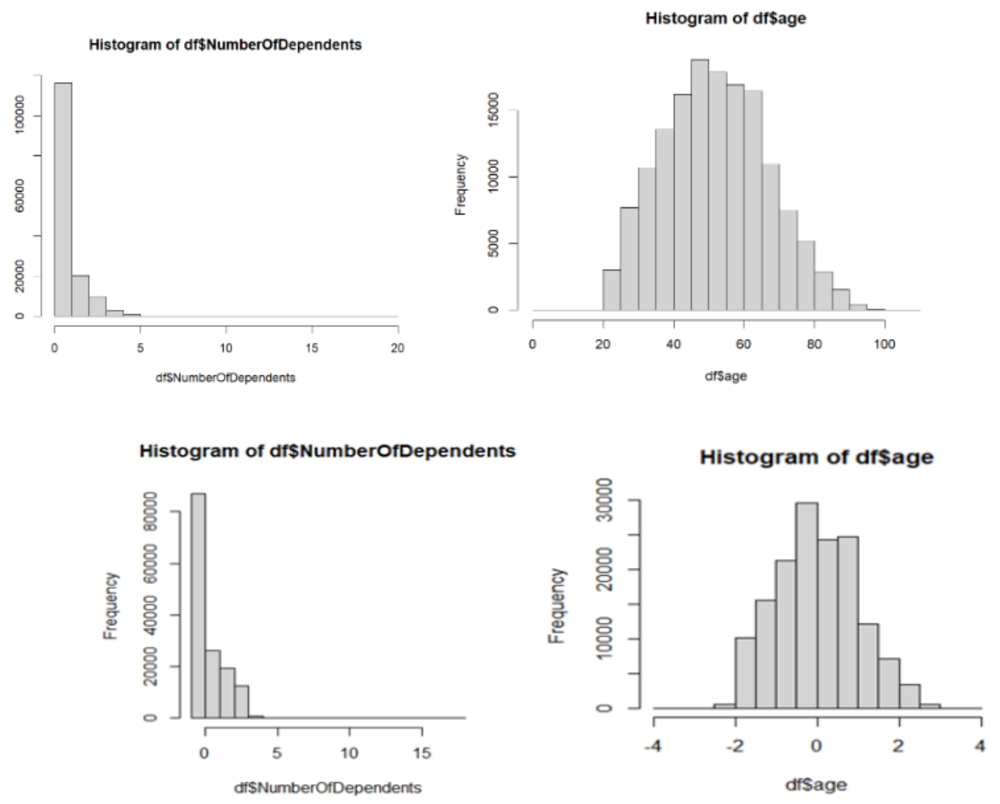
5. Conclusion and recommendations

Stacking using the neural network as our manager model helped optimize the blend of all our earlier models and performed better in terms of error rate in predicting those likely to default in two years. Based on this report it is evident that the Classification tree model performed better than our Logistic regression model; however, the Neural network and stacking models performed the best. The benefits of the LR model and the classification tree models are ease of interpretability both models agreed on the variables of 'NumberOfTimes90DaysLate', "NumberOfTime30-59DaysPastDueNotWorse' and 'NumberRealEstateLoansOrLines' being significant in default prediction. It is important for the financial institution to consider these factors in guiding their decision-making with regard to issuing credit.

In general terms, the stacked model performed better for prediction-making than the neural nets, classification tree, and logistic regression. All of the model performance indicators show that the stacked model performs better and commits fewer mistakes; however, it's not efficient enough in terms of predicting those clients who will face a financial crisis.

Despite the stacking neural nets model performing better than every other, it is important to remember the sensitivity levels of its performance. Even though its predictions are more precise, the ones for customers that will default in two years need to have more success, as a higher sensitivity rate would be beneficial to the bank's operations

6. Appendix



Appendix 1: before and after standardization

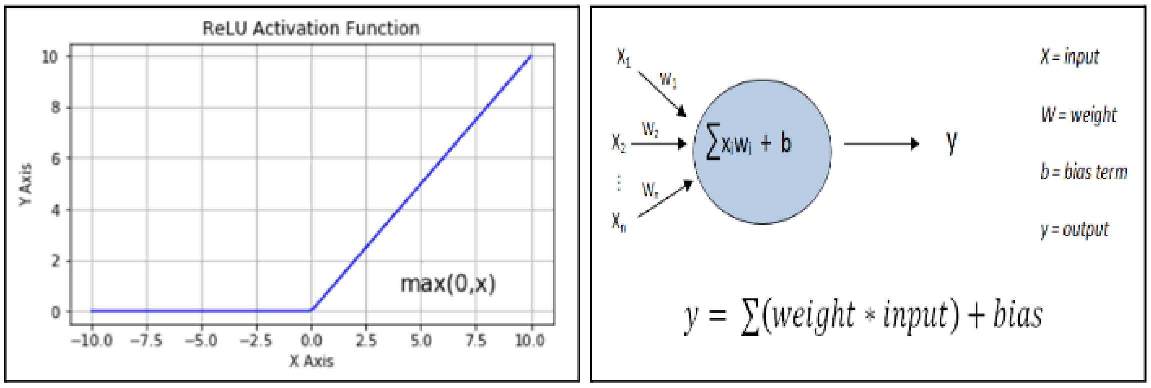


Figure 2: Neural Nets Activation function. (A) represents the inputs and weights for the node and how the activation function generates the output. (B) Line Plot of Rectified Linear Activation for Negative and Positive Inputs