Final Project

Miguel Novo Villar, Dikshant Tiwari, Maheen Ansari, Lisa Pink & Thomas Durkin
Simon School of Business, University of Rochester

CIS432: Predictive Business Analytics with Python
Dr. Shaposhnik
April 28th, 2022

1. Introduction

The ever-increasing availability of data and computing power has changed the way in which businesses operate. Using machine learning techniques to perform predictive analytics can improve the operational efficiency and profitability of a business. This report will provide an in-depth discussion of how predictive modeling was applied to the Home Equity Line of Credit (HELOC) dataset to obtain a prototype for a decision support system. Using "streamlit" as a front-end interface we will emulate a support dashboard utilized by a third party to make business decisions based on credit risk. The algorithm selected for this project is logistic regression as it combines a stable performance across testing, training and validation sets and also uses computational resources in an efficient manner.

2. Dataset

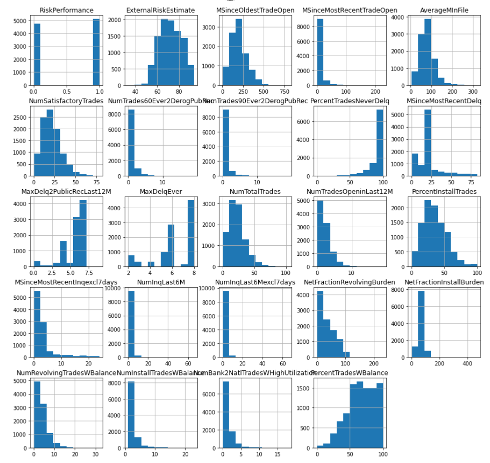
The HELOC dataset contains real credit applications that were made by homeowners who requested a line of credit ranging from $5,000 to $150,000. The credit report of each homeowner is analyzed to predict whether or not the HELOC account will be successfully repaid within two years. Aiming to classify the risk performance of a homeowner as "Good" or "Bad", we rely on predictor variables from the credit report. The classification of "Bad" risk performance entails a customer being at least 90 days past due one or more times over the two-year period from when the customer opened the credit account. In opposition, a customer falls under the "Good" classification of risk performance if there were no payments more than 90 days overdue.

3. Experiment

a) Data Preprocessing & Exploratory Data Analysis

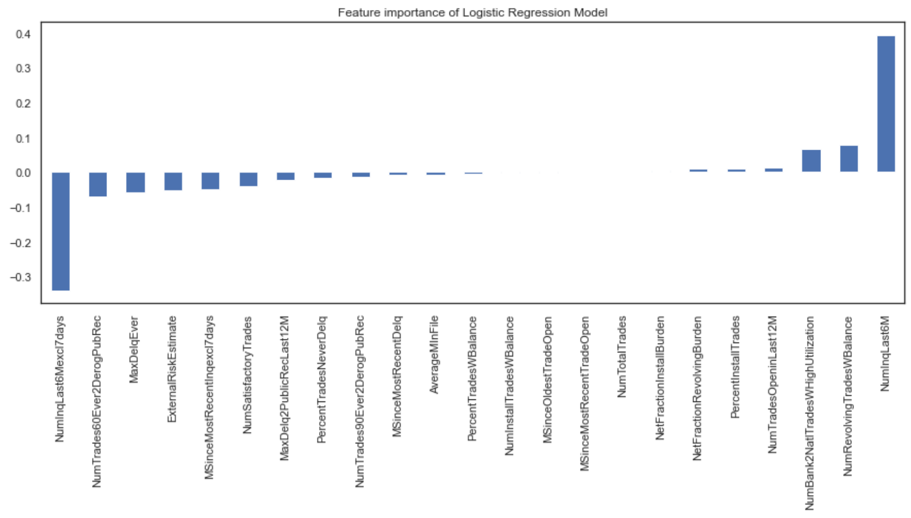
The first step in the process was to create an overview of the dataset and gain a general understanding of the distribution of the variables. The multi histogram function from python demonstrated that there were a large number of negative variables that would need to be engineered in order to implement our predictive algorithms and present the result to a hypothetical stakeholder. Based on the nature of the negative variables we decided to eliminate the columns that contained a "-9" and impute the mean or median of the variables for "-8" and "-7" based on the previously observed distributions of the attributes - for non-skewed variables the mean was imputed, whereas for skewed variables the median was imputed. Figure 1 displays the distributions of the independent variables after the previously stated feature engineering.

Figure 1



The next figure displays the attributes with a stronger weight on the outcome of the predictions. Figure 2 shows the logistic regression coefficients plotted and ranked in terms of weight in the final prediction. It is important to highlight that the coefficients from "NumInqLast6Mexcl7days" and "NumInqLast6M" yielded the highest absolute coefficients in the implemented logistic regression model.

Figure 2



b) Models implemented and tested

Table 1 displays the 4 different models implemented for this project. The prediction uses a binary classifier to determine whether a certain customer has good or bad credit. Several approaches were implemented and the best result in terms of reliability, computational burden, potential scalability and explainability is LogisticRegression.

Table 1

Model	Train Accuracy	Validation Accuracy	CV Accuracy
DecisionTreeC	0.623	0.615	0.623
KNeighborsC	0.668	0.672	0.668
GradientBoostingC	0.712	0.718	0.725
LogisticRegressionC	0.725	0.732	0.725

c) Best model selection

As stated in the introduction and in section “b”, the desired algorithm to implement in the front-end solution is Logistic Regression as it yields a consistent accuracy of over 72% in all the different tests and is easy to explain to stakeholders - it makes a binary prediction based on a probability over or under 50%. While DecisionTreeC and KNeighborsC were automatically discarded due to the lower scoring across all trials, the reason why GradientBoosting has not been selected is due to the bigger computational burden with 1000 estimators (2.74 s ± 153 ms per loop) and the more complex explainability to a potential stakeholder that wants to understand the model that is running in the background as it combines outputs and builds a tree structure.

4. Front-end solution

After making final decisions regarding which model to use, our team developed an interactive interface that could be used by sales representatives to automate the decision of whether to accept or reject an application. The prototype was created using the Streamlit platform, and was designed with the intention of being used by a bank representative who is assumed to not have any knowledge of the field of machine learning and predictive analytics. With this in mind, the interface was kept simple on the front end. The interface consists of the home page, application page, and file upload page.

The home page welcomes the user to the application with a title and brief description of the goal of the dashboard. On the far left, a drop-down bar allows the user to navigate from the home page to the other pages. The application form page holds the main functionality of the application. Here, the user enters a value for each variable based on the customer's credit report information. Upon pressing the submit button, the model runs in Python using the values for each variable and outputs the recommendation of the model for the application to be accepted or rejected. Below the suggestion is a table that is also generated to help provide some justification for the decision. The table contains insight and a percentile for the top 9 most influential variables. We decided to only display the top 9 because many of the features were of very low importance to the model and displaying them was making the output appear too cluttered. The top 9 can be found in Figure 2. The percentile represents which percentile the entered value falls for each category, compared to past customers (the dataset). An insight is then provided for further interpretability. The classification is based on the percentile, with the top 20% being classified as "Very good", the bottom 20% being "Very poor", and three more classifications of "Good", "Fair", and "Poor" filling in the middle 60%. For some variables, a higher value is more desirable and for other variables a lower value is better - the Python script took this into consideration when building the table and classified it accordingly.

The final page provides a file upload option, which allows the user to upload a CSV file with multiple applications. This could improve efficiency by allowing the user to obtain the suggested application decision for multiple users simultaneously.

5. Conclusion

In conclusion, we were able to build a user-friendly support system for banks that determines whether a person's HELOC application should be rejected or accepted. We determined that the best model to use was logistic regression because it is easily understood and its accuracy was on par or above all the other models tested. Unfortunately, using GridSearch to tune the hyperparameters yielded no further improvements to our model. A model with around 73% accuracy is still reliable so an improvement in performance would only marginally boost the accuracy of the support system's predictions. In the future, we would like to potentially improve performance by switching our model to a neural network as they are able to identify hidden patterns between the dependent and independent variables.