Team member: Yuanzhuo Wang, Yi Lu, Yibo Hu, Ziwei Shi

Professor Yaron Shaposhnik

CIS 242

22 April 2019

Explanation of the Model and DSS for HELOC Evaluation

I. Abstract

Our aim is to predict the credit risk of HELOC applications to be good or bad. Firstly, after the special values cleaning, categorical variable handling and feature scaling, we used cross validation and hyper-parameter tuning to train 7 different models and compared their performances. We decided on linear support vector classifier model to make our final prediction. Then, we developed an interactive interface for our model using "ipywidgets". Finally, in order to better interpret the model result, we transformed the decision function result to a probability percentage, which is the default risk. In addition, for each input application, we listed out the 5 most significant contributing factors and their contribution in decision function result that led to its final risk default prediction result.

II. Data Insight

This credit risk performance prediction is a supervised learning classification problem. There are 23 inputs variables, and the output label is the risk performance, which is good or bad. After showing the descriptive statistics summary of the dataset, we calculated and counted the number of outliers for each attribute. There is an average of about 800 out of 10000 outliers for each attribute, which is a relatively very large number. Then, after understanding the financial meaning of each attribute, we constructed a HeatMap to visualize the correlations between any two attributes. As shown in our code, lighter the color is, closer the relationship between two attributes are.

III. Data Cleaning

In order to prepare the data for training, we needed to first clean the data.

1. Drop Empty Rows

There were 588 rows among the 10,459 rows with special value "-9" in all attributes in the dataset. For those meaningless data, we dropped them. Turn out to be, 323 of the dropped applications were bad, and 265 of them were good.

2. Count Frequency of Occurrence of Special Values

There are three special values in our dataset: -7, -8, -9. -7 represents that condition not met, which means no information of that type; -8 represents no useable/valid trades or inquiries, which means no useable information; -9 represents no bureau record or no investigation, which means when a loan applicant's credit bureau report was either not investigated or not found, all features obtained from the credit bureau report receive -9[1]. We wanted to see the number of occurrences each special value has, and in which attribute it occurs. So we constructed a table to count the frequency of occurrence, as shown in our code. This gave us insights of which attributes need to be clean for which special value.

3. Clean the Special Values in Dataset

For special value -9, there were only 10 rows left, so we dropped rows with -9. Then there were special values -7 and -8. We had two potential methods: mean and median for them, so there were four combinations: (median for -7, median for -8), (median for -7, mean for -8), (mean for -7, median for -8), (mean for -7, mean for -7). We tried all these four combinations to train the models. Finally, we used median for -7 and mean for -8 because it had the highest prediction accuracy.

4. Convert and Store All Categorical Attributes to Indicator Variables

After cleaning the special values, we converted all categorical variables to dummy variables with binary values to prepare for later computation.

IV. Scaling & Model Selection

After cleaning the data, we had five potential feature scaling methods: StandardScaler, MinMaxScaler, MaxAbsScaler, RobustScaler and Normalizer. Also, we chose 7 different potential models to fit on the dataset: Support Vector Machine (SVM), Logistic Regression (LR), K-Nearest Neighbors (KNN), Naive Bayes (NB), Decision Tree (Tree), Random Forest (RF), and Boosting.

---

[1] FICO Community. Accessed April 14, 2019. https://community.fico.com/s/explainable-machine-learning-challenge?tabset-3158a=4fbc8.

Then we chose the combination of model and scaler with the highest test score. The two highest ones after the hyper-parameter tuning were SVM with StandardScaler, and LR with MaxAbsScaler, which had Test Score of 0.7210 and 0.7226 respectively. Notice that actually, LR with MaxAbsScaler had about 0.0016 higher score than SVM with StandardScaler, but we did not choose LR since SVM was more appropriate than LR in our situation—"SVM minimizes hinge loss while LR minimizes logistic loss. Logistic loss diverges faster than hinge loss, so LR will be more sensitive to outliers"[2]. As mentioned early, the dataset had a really large number of outliers, average of 799 outliers for each attribute. Therefore, we chose SVM as our final model instead of LR.

In our final SVC model parameters, we used [0.1, 1] as the penalty, "linear" as the kernel. Moreover, the reason why we chose Test Score instead of Cross Validation (CV) Score was that we had a very large dataset. Around 10,000 rows of data gave us the ability to have an accurate Test Score after we split it into training set and test set. Thus, Test Score was more independent of this original train data and more accurate to use.

V. SVC: Decision Function

We chose SVM as the classifier of our model. Then we used training data to find the best model, and the corresponding coefficients of this model, using the methods: "final_best_model.coef()", and "final_best_model.intercept()". Notice that we used linear SVM here. So we can predict the output "y" using "final_model.decision_function", which is the same as calculating manually by the equation: $Y = a + bx_1 + cx_2 + dx_3 + \ldots$, as shown in our code. This output can indicate the risk of loans: if y is positive, it means that the risk performance is good, i.e. the risk of them not paying back is low, vice versa. Also, the more positive y is, the lower risk the applicant has of not paying back.

VI. Interactive Interface & Explanation

After our decision model was done, we designed an interactive interface to demonstrate our model by "Ipywidgets". We did not assume any technical proficiency of the users, and users

[2] Drakos, Georgios, and Georgios Drakos. "Support Vector Machine vs Logistic Regression." Towards Data Science. August 12, 2018. Accessed April 19, 2019. https://towardsdatascience.com/support-vector-machine-vs-logistic-regression-94cc2975433f.

do not need to know anything about coding. They just only need to use the slider to drag the input variables to the value they want to predict, and then our model will give the result, ranging from 0% to 100%, and detailed explanations of why they are accepted or rejected.

After finishing step V, we could already predict whether we should give him/her loan. However, the number of output "y" was not explainable. So what we did in this step was to transform the "y" from above to a well-known measurement called "default risk". Default risk is the chance that companies or individuals will be unable to make the required payments on their debt obligations[3]. Firstly, we multiplied the standardized value input and the model coefficient calculated in step V to a variable called "contribution". Secondly, we added all attributes' contributions together, and also the intercept calculated in step V to the variable "result". Thirdly, we transformed "result" to "default risk" by the sigmoid function. The range of the variable "result" is mostly (-5,5), while we wanted our risk to be a percentage ranging from 0% to 100%. So we used sigmoid function to help us achieve this aim.

In addition, as shown in our three demos, every applicant also got some different reasons why they were accepted or rejected. We accomplished this by showing the 5 most significant contributing factors that affected the prediction result, ordered by importance. Also, for each important factor, we showed its contribution in the final decision. In this way, applicants could easily understand why they were accepted or rejected.

VII. Key Lessons Learned

- Using tables to gain data insights into special values.
- Data cleaning methods of handling special values.
- Hyper-parameter tuning to find the best model by comparing test scores.
- Difference between Linear SVC and LR—sensitivity to outliers.
- Interactive interface by "Ipywidgets".
- Explanations of our result by giving users a probability of Default Risk and 5 most significant contributing factors based on different user inputs.

---

[3] Kagan, Julia. "Default Risk." Investopedia. April 17, 2019. Accessed April 19, 2019. https://www.investopedia.com/terms/d/defaultrisk.asp.