CIS 432 Final Project: ML Project

By Team 26., Ruxin Cheng, Jichao Gui, Ching-An Chung, Zhen Yang, Weiming Zhu

Introduction

For this project, we tend to use several models to find the best application for our traditional procedures of apply financial data to predict the best prediction for the financial credit default risk. To achieve this goal, we analyzed our financial data in these models: Linear Regression / Linear SVM/ SVM (Polynomial) / SVM (Radial basis)/ Linear Discriminant Analysis/KNN/ Random Forest/Logistic Regression. After we analyzed all 8 models, we made a decision with one model (Random Forest) which produced one best result out of eight models.

Data Cleaning

There are 24 features in this dataset. Risk Performance is the variable that we are trying to predict, which has only two values: 'good' or 'bad'.

We use 0 to represent bad, and 1 to represent good.

Among the dataset, -7,-8,-9 represent no usable/valid values. To clean these data, we first took out rows that all of their values are -9. Then we change -7, -8, -9 to na.

To replace these NA values, we check standard deviations of each column. For columns with large standard deviation, we replace NA values with median. For columns with small standard deviation, we replace NA value with mean. The logic behind this is that large standard deviation columns can be strongly affected by extreme value, so median can be more representative.

Features that we replace with median:

1. MSinceMostRecentDelq,
2. ExternalRiskEstimate,
3. MSinceOldestTradeOpen,
4. AverageMInFile,
5. MaxDelq2PublicRecLast12M,
6. MaxDelqEver,
7. MaxDelqEver,
8. NumTradesOpeninLast12M,
9. NumInqLast6M,
10. NumInqLast6Mexcl7days,
11. NetFractionRevolvingBurden,

12. NetFractionInstallBurden,
13. NumRevolvingTradesWBalance,
14. NumInstallTradesWBalance,
15. PercentTradesWBalance,

Other columns we replace NA value by means.

Finally, we split the data to training dataset(two-thirds) and test dataset(one-third), and we also scaled all SVM and KNN models.

Model Selection

After data has been cleaned, we begin to construct models, which includes:

Linear Regression / Linear SVM/ SVM (Polynomial) / SVM (Radial basis)/ Linear Discriminant Analysis/KNN/Logistic Regression

We use Mean Square Error to measure the performance of each model, and we cross validate each model(set cv = 5).

The results are listed below:

Linear Regression: 0.4284784252549584
Linear Regression(drop variable with low correlation): 0.43374040890532095 (We try to drop variables with low correlation in Linear Regression, but the performance seems worse.)
Linear SVM : 0.4338906281789418
SVM (Polynomial): 0.5298561964206132
SVM (Radial basis): 0.5147015500374474
Linear Discriminant Analysis: 0.5227348171332183
KNN: 0.612097326641144
Logistic Regression: 0.5236607046432165

Model Explanation

After running 8 kinds of models(Linear Regression / Linear SVM/ SVM (Polynomial) / SVM (Radial basis)/ Linear Discriminant Analysis/KNN/Random Forest/Logistic Regression)), we select the model with the smallest root mean squared error or highest accuracy, which is the random forest model (Accuracy: 0.71). We will discuss the mean squared error, accuracy, precision, and recall for each model.

Best Model – Random Forest (Result & Interpretation)

To find more accurate results, we implement hyper-parameter tuning, using grid search and cross validation to find the best estimator.

```
param_grid = [{'n_estimators':[3,10,30],'max_features':[2,4,6,8]},
              {'bootstrap':[False],'n_estimators':[3,10],'max_features':[2,3,4]}]
```

```
RandomForestRegressor(bootstrap=True, criterion='mse', max_depth=None,
                      max_features=2, max_leaf_nodes=None,
                      min_impurity_decrease=0.0, min_impurity_split=None,
                      min_samples_leaf=1, min_samples_split=2,
                      min_weight_fraction_leaf=0.0, n_estimators=30,
                      n_jobs=None, oob_score=False, random_state=None,
                      verbose=0, warm_start=False)
```

Then, we present the confusion matrix, the classification report, the root mean squared error, accuracy rate, and feature importance.

- Root Mean Squared Error: 0.439398147326179
  Accuracy: 0.7117863720073665

The random forest with best estimator has 71% accuracy rate and 43% mean squared error.

We can consider the accuracy rate, since the data set is not very unbalanced, with 47% Good Risk Performance and 53% Bad Risk Performance.

- Confusion Matrix & Classification Report

```
              precision    recall  f1-score   support

           0       0.71      0.76      0.73      1694
           1       0.72      0.65      0.69      1564

    accuracy                           0.71      3258
   macro avg       0.71      0.71      0.71      3258
weighted avg       0.71      0.71      0.71      3258

[[1295  399]
 [ 540 1024]]
```

The recall rate for our model is 65%, indicating that we successfully predicted 65% of the true positive, and 76% of the true negative.

From our model when a person is classified as 1(Good Risk Performance), 72% are true positive, which means 72% of the positive predictions are correctly classified.
Compare with prevalence, when we classified randomly, the positive rate is only 47% = (4735 / (4735+5136)), indicating that after our analytics, our accuracy improved 53%.

Feature Importance

	Imporatance
ExternalRiskEstimate	0.112755
NetFractionRevolvingBurden	0.075978
AverageMInFile	0.068570
MSinceOldestTradeOpen	0.068552
PercentTradesWBalance	0.060203
NumSatisfactoryTrades	0.055841
NumTotalTrades	0.050649
PercentInstallTrades	0.049474
PercentTradesNeverDelq	0.046529
NetFractionInstallBurden	0.044785
MSinceMostRecentTradeOpen	0.042940
NumRevolvingTradesWBalance	0.038512
MSinceMostRecentDelq	0.036841
NumBank2NatlTradesWHighUtilization	0.035212
MSinceMostRecentInqexcl7days	0.034130
NumTradesOpeninLast12M	0.028264
NumInqLast6M	0.026673
NumInstallTradesWBalance	0.026461
NumInqLast6Mexcl7days	0.023931
MaxDelq2PublicRecLast12M	0.023166
MaxDelqEver	0.022388
NumTrades60Ever2DerogPubRec	0.016121
NumTrades90Ever2DerogPubRec	0.012026

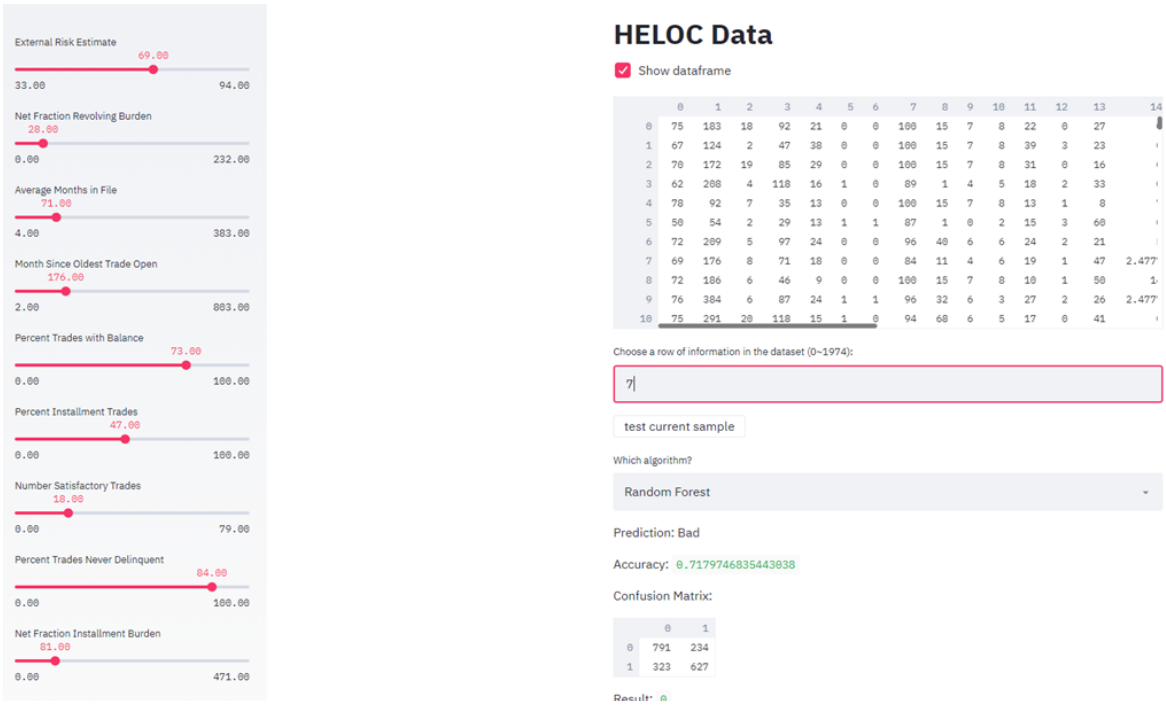
From the feature importance data frame, we can find out that the most important features are External Risk Estimate, Net Fraction Revolving Burden and Average Min File. In the interface part, we will discuss how those features affect the result.

Summary

After our analysis, we believe we got result of 71% of accuracy rate from random forest model. With Random forest model, we got 72% true positive which could interpret as 72% of prediction are corrected classified. By using Random forest model, we improved our accuracy by 53%.

For interface section, we interpreted our results into a straightforward platform which could provide the platform user with more easy understanding of the interface. Based on our analysis, we believe there are several features are really important when it comes to predict the risk of default. We ranked them and assigned a weight based on their importance.

Interface Result



The graph is the screenshot of our interface.

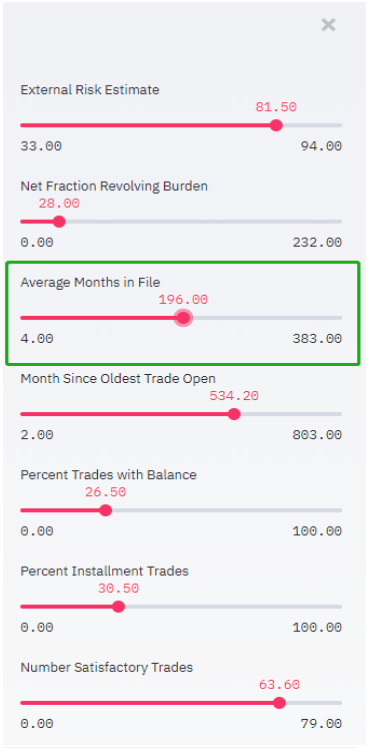
On the left-hand side, it is the side bar which contains all the features in the dataset. More features can be displayed by scrolling down the side bar. We choose to do this instead of input value box because it is clearer for user to see the minimum and maximum value, user just need to drag the dots to the number they need to see the prediction result. Also, the color and the dots are better demonstrations than pure numbers to show the chosen value. Clicking “Show dataframe” will show the test data set below. We can obtain a row of information in the dataset by inputting the number of rows we need. The value of different features for the row will be displayed in the side bar. “Test current sample” button is used to indicate the values of different features. By choosing the algorithm which contains Random Forest and Logistic Regression, the interface will then show the prediction. The accuracy and the confusion matrix is the predicted result for the test data. The interface is not limited to the dataset. Users can change the input value for features and obtain the predicted result by choosing different algorithms.

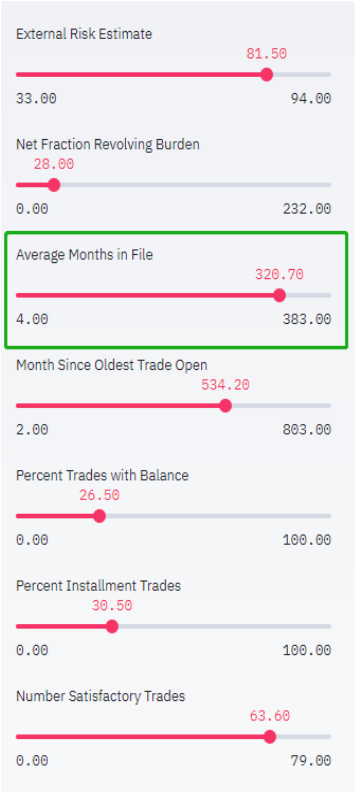
For the technical proficiency of the user, the requirement of user technical proficiency is low because users only need to manually adjust the values of different features they need, then the model will indicate the result of the risk performance.

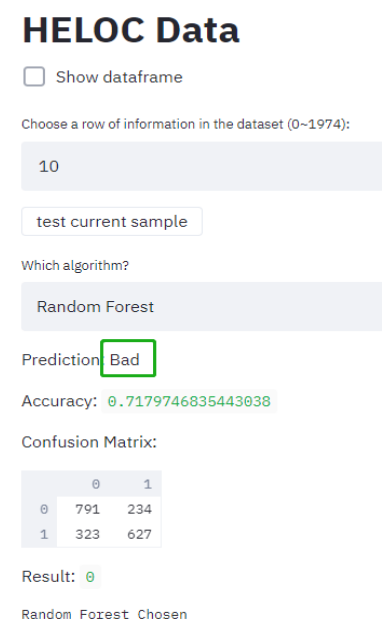
	Imporatance
ExternalRiskEstimate	0.112755
NetFractionRevolvingBurden	0.075978
AverageMInFile	0.068570
MSinceOldestTradeOpen	0.068552
PercentTradesWBalance	0.060203
NumSatisfactoryTrades	0.055841
NumTotalTrades	0.050649
PercentInstallTrades	0.049474
PercentTradesNeverDelq	0.046529
NetFractionInstallBurden	0.044785
MSinceMostRecentTradeOpen	0.042940
NumRevolvingTradesWBalance	0.038512
MSinceMostRecentDelq	0.036841
NumBank2NatlTradesWHighUtilization	0.035212
MSinceMostRecentInqexcl7days	0.034130
NumTradesOpeninLast12M	0.028264
NumInqLast6M	0.026673
NumInstallTradesWBalance	0.026461
NumInqLast6Mexcl7days	0.023931
MaxDelq2PublicRecLast12M	0.023166
MaxDelqEver	0.022388
NumTrades60Ever2DerogPubRec	0.016121
NumTrades90Ever2DerogPubRec	0.012026

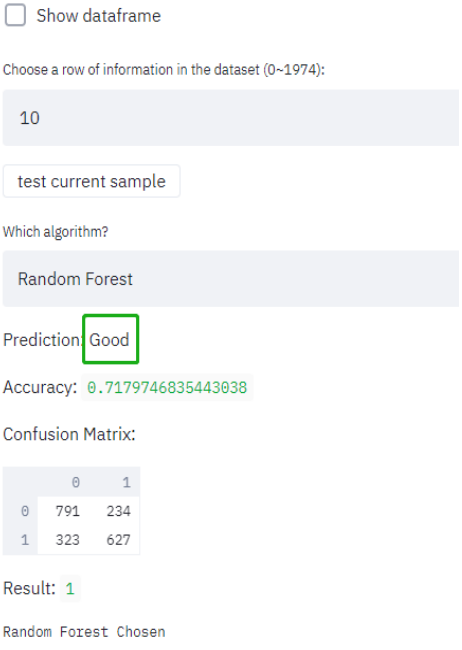
Here is the importance ranking of all the feature, we select the top seven features to demonstrate the model by using Random Forest.

At the first picture, our prediction is bad. Holding other features constant, if we increase the average months in file, our model will have prediction good. It means, the model can help user to know the particular applicant may qualify for a line of credit if they are in file for longer time.









Another example is the number of satisfactory trades. If other features keep their values and we only change the value of number of satisfactory trades, higher number of satisfactory will have prediction result is good, lower number of satisfactory trades will result in prediction result is bad.