Ky Anh Nguyen

Springboard Data Science Boot Camp

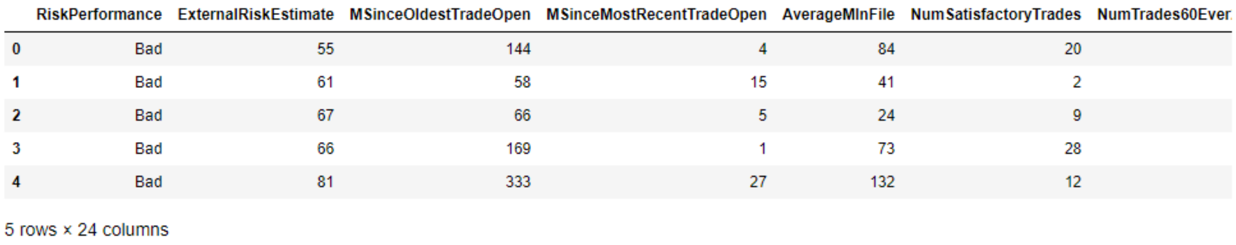
Home Equity Line of Credit

Introduction

A home equity line of credit, or HELOC, is a loan in which the lender agrees to lend a maximum amount with an agreed period (called a term), where the collateral is the borrower's equity in his/her house (akin to a second mortgage). Since a home is often a consumer's most valuable asset, many homeowners tend to only use home equity credit lines for major items such as education, home improvements, medical bills, and not to use them for day-to-day expenses. In this capstone project, I analyzed and utilized the dataset "Home Equity Line of Credit (HELOC)" which originally comes from the Explainable Machine Learning Challenge organized by the FICO company. The data consists of anonymized credit applications of HELOC credit lines (which are a type of loan) collateralized by a customer's property. The goal of this project is to build a model that can predict which clients will repay their HELOC account within 2 years, and those who will not. Furthermore, we will take a look at the most important features in the dataset and how we can utilize this information in order to steer more clients into paying their HELOC account in time.

Data Wrangling

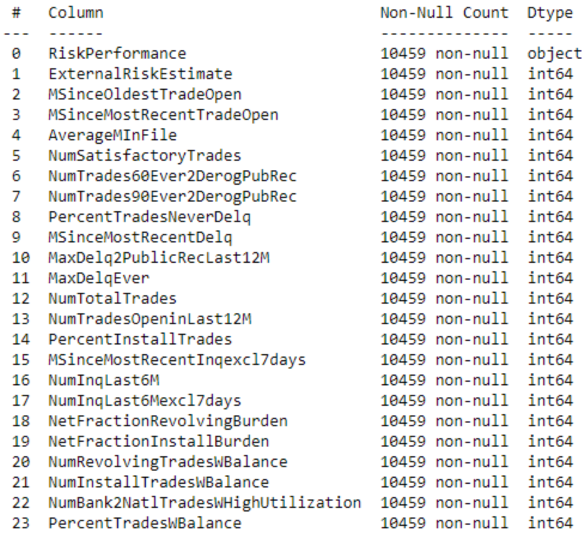
After importing the necessary packages and libraries, the first step is to always load the dataset and take a look at it by using the .head() function.



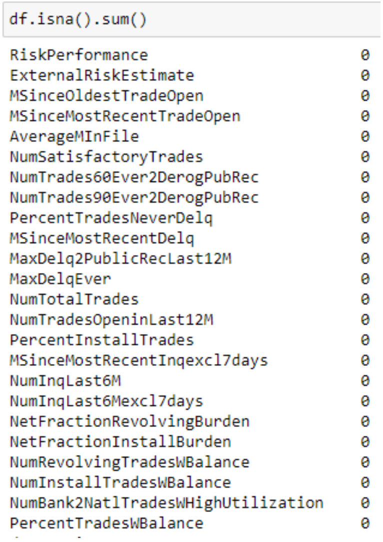
	RiskPerformance	ExternalRiskEstimate	MSinceOldestTradeOpen	MSinceMostRecentTradeOpen	AverageMInFile	NumSatisfactoryTrades	NumTrades60Ever
0	Bad	55	144	4	84	20	
1	Bad	61	58	15	41	2	
2	Bad	67	66	5	24	9	
3	Bad	66	169	1	73	28	
4	Bad	81	333	27	132	12	

5 rows × 24 columns

The dataset is relatively large, containing 10459 rows and 24 columns. Fortunately, the dataset did not contain any null values, as we can see below:



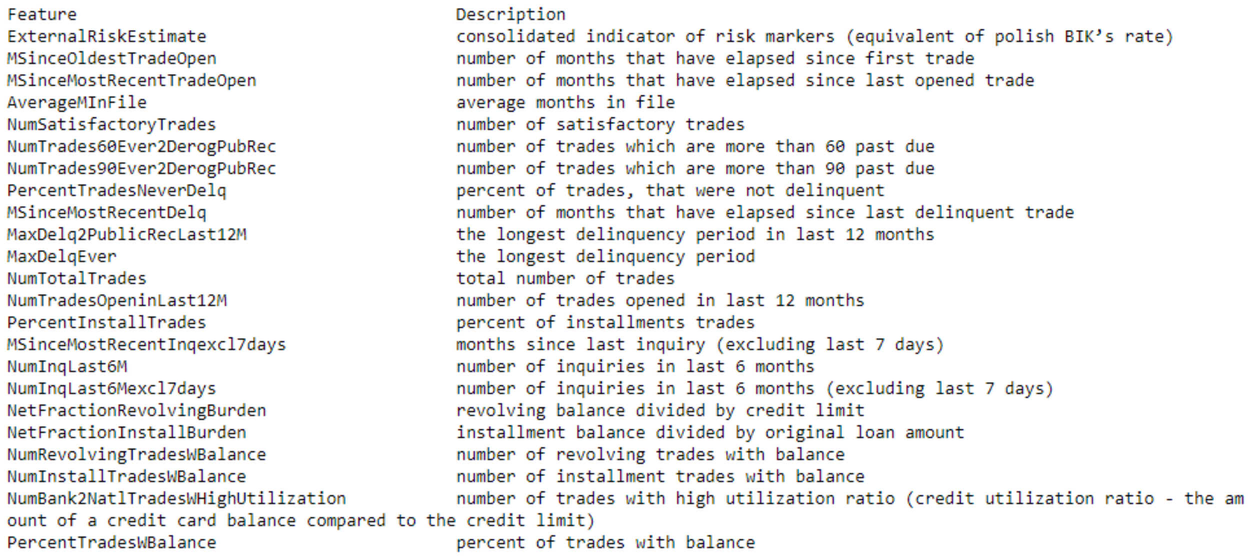
```
 #   Column                              Non-Null Count  Dtype
---  ------                              --------------  -----
 0   RiskPerformance                     10459 non-null  object
 1   ExternalRiskEstimate                10459 non-null  int64
 2   MSinceOldestTradeOpen               10459 non-null  int64
 3   MSinceMostRecentTradeOpen           10459 non-null  int64
 4   AverageMInFile                      10459 non-null  int64
 5   NumSatisfactoryTrades               10459 non-null  int64
 6   NumTrades60Ever2DerogPubRec         10459 non-null  int64
 7   NumTrades90Ever2DerogPubRec         10459 non-null  int64
 8   PercentTradesNeverDelq              10459 non-null  int64
 9   MSinceMostRecentDelq                10459 non-null  int64
 10  MaxDelq2PublicRecLast12M            10459 non-null  int64
 11  MaxDelqEver                         10459 non-null  int64
 12  NumTotalTrades                      10459 non-null  int64
 13  NumTradesOpeninLast12M              10459 non-null  int64
 14  PercentInstallTrades                10459 non-null  int64
 15  MSinceMostRecentInqexcl7days        10459 non-null  int64
 16  NumInqLast6M                        10459 non-null  int64
 17  NumInqLast6Mexcl7days               10459 non-null  int64
 18  NetFractionRevolvingBurden          10459 non-null  int64
 19  NetFractionInstallBurden            10459 non-null  int64
 20  NumRevolvingTradesWBalance          10459 non-null  int64
 21  NumInstallTradesWBalance            10459 non-null  int64
 22  NumBank2NatlTradesWHighUtilization  10459 non-null  int64
 23  PercentTradesWBalance               10459 non-null  int64
```



```
df.isna().sum()
```

```
RiskPerformance                       0
ExternalRiskEstimate                  0
MSinceOldestTradeOpen                 0
MSinceMostRecentTradeOpen             0
AverageMInFile                        0
NumSatisfactoryTrades                 0
NumTrades60Ever2DerogPubRec           0
NumTrades90Ever2DerogPubRec           0
PercentTradesNeverDelq                0
MSinceMostRecentDelq                  0
MaxDelq2PublicRecLast12M              0
MaxDelqEver                           0
NumTotalTrades                        0
NumTradesOpeninLast12M                0
PercentInstallTrades                  0
MSinceMostRecentInqexcl7days          0
NumInqLast6M                          0
NumInqLast6Mexcl7days                 0
NetFractionRevolvingBurden            0
NetFractionInstallBurden              0
NumRevolvingTradesWBalance            0
NumInstallTradesWBalance              0
NumBank2NatlTradesWHighUtilization    0
PercentTradesWBalance                 0
```

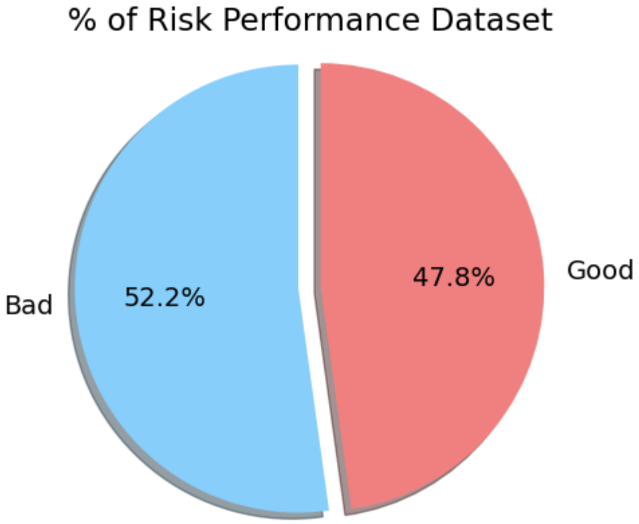
Whenever we have access to the descriptions of each column, it is useful to then make a table with the column names and their corresponding descriptions. The “RiskPerformance” column classifies clients as either “Good” or “Bad.” Clients that repaid their HELOC account within two years were classified as good and those that did not were classified as bad. The descriptions for the rest of the columns can be seen in the table below:



```
Feature                                     Description
ExternalRiskEstimate                        consolidated indicator of risk markers (equivalent of polish BIK’s rate)
MSinceOldestTradeOpen                       number of months that have elapsed since first trade
MSinceMostRecentTradeOpen                   number of months that have elapsed since last opened trade
AverageMInFile                              average months in file
NumSatisfactoryTrades                       number of satisfactory trades
NumTrades60Ever2DerogPubRec                 number of trades which are more than 60 past due
NumTrades90Ever2DerogPubRec                 number of trades which are more than 90 past due
PercentTradesNeverDelq                      percent of trades, that were not delinquent
MSinceMostRecentDelq                        number of months that have elapsed since last delinquent trade
MaxDelq2PublicRecLast12M                    the longest delinquency period in last 12 months
MaxDelqEver                                 the longest delinquency period
NumTotalTrades                              total number of trades
NumTradesOpeninLast12M                      number of trades opened in last 12 months
PercentInstallTrades                        percent of installments trades
MSinceMostRecentInqexcl7days                months since last inquiry (excluding last 7 days)
NumInqLast6M                                number of inquiries in last 6 months
NumInqLast6Mexcl7days                       number of inquiries in last 6 months (excluding last 7 days)
NetFractionRevolvingBurden                  revolving balance divided by credit limit
NetFractionInstallBurden                    installment balance divided by original loan amount
NumRevolvingTradesWBalance                  number of revolving trades with balance
NumInstallTradesWBalance                    number of installment trades with balance
NumBank2NatlTradesWHighUtilization          number of trades with high utilization ratio (credit utilization ratio - the am
ount of a credit card balance compared to the credit limit)
PercentTradesWBalance                       percent of trades with balance
```

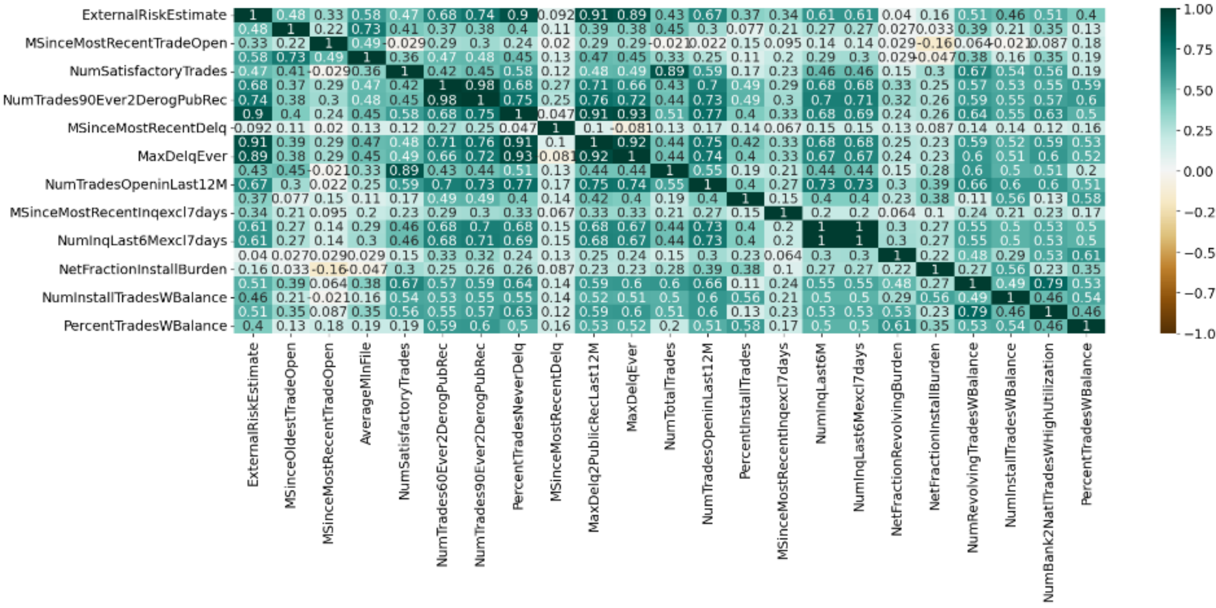
Exploratory Data Analysis

The first piece of information we should examine is what proportion of clients were classified as either good or bad. Since there are only two values in this column, a pie chart will suffice in presenting these proportions together:

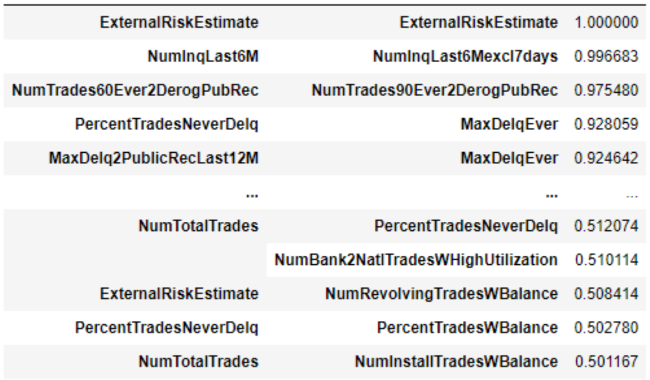


We can see that there is a slight majority of bad clients to good clients. This can also be verified in the fact that there are 5459 bad clients and 5000 good clients. Since “RiskPerformance” is our target variable and that the number of good clients and number of bad clients are almost equal, we can say that this is a balanced dataset.

In this dataset, we have a good quantity of features to work with (24 of them in fact). If we take a look at the descriptions of some of the features we can see that they are closely related to each other. To get a better examination of these relationships, let's take a look at a heat map of the correlation between the features (excluding risk performance) and organize these correlations from largest to smallest.

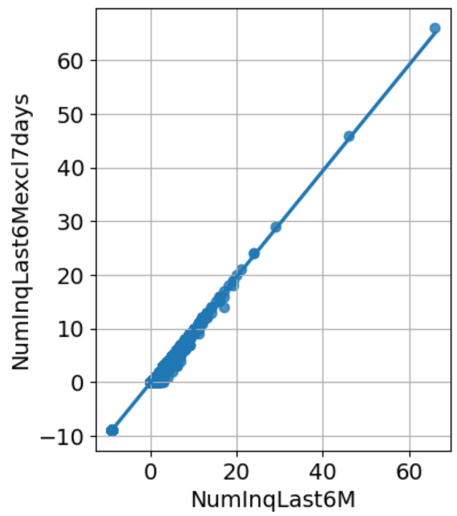


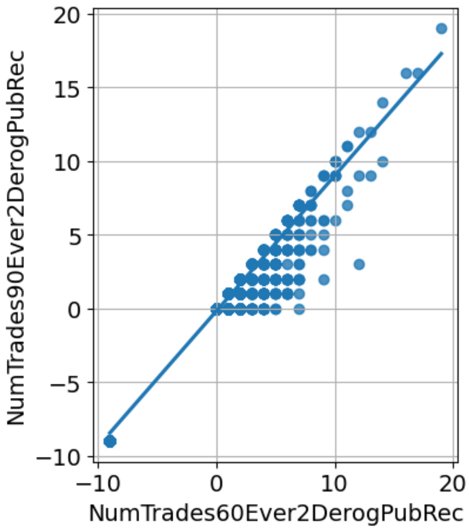
From the heat map above, we see that the vast majority of correlations between features are positive. There are only 6 pairs of features that negatively correlate with each other. If the coefficient value of the correlation lies between ± 0.5 and ± 1, then it is said to be a strong correlation. Given that, let's take a look at a few pairs of features whose correlation is between ± 0.5 and ± 1.



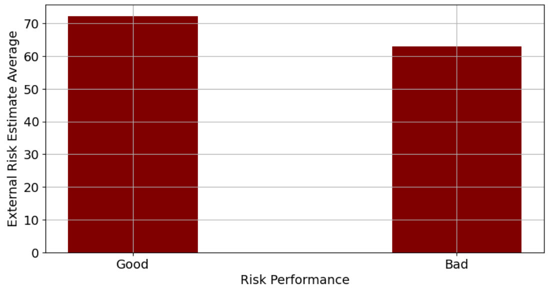
ExternalRiskEstimate	ExternalRiskEstimate	1.000000
NumInqLast6M	NumInqLast6Mexcl7days	0.996683
NumTrades60Ever2DerogPubRec	NumTrades90Ever2DerogPubRec	0.975480
PercentTradesNeverDelq	MaxDelqEver	0.928059
MaxDelq2PublicRecLast12M	MaxDelqEver	0.924642
...	...	...
NumTotalTrades	PercentTradesNeverDelq	0.512074
	NumBank2NatlTradesWHighUtilization	0.510114
ExternalRiskEstimate	NumRevolvingTradesWBalance	0.508414
PercentTradesNeverDelq	PercentTradesWBalance	0.502780
NumTotalTrades	NumInstallTradesWBalance	0.501167

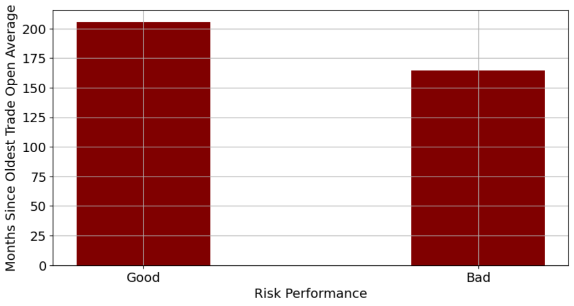
Disregarding the first pair (as they are the same feature), there are several pairs of features that have a very strong correlation. However, by taking a look at the descriptions of some of these features, we can see why they are so highly correlated. For example, “NuminqLast6M” and “NuminqLast6Mexcl7days” have a correlation of 0.997. That is because they have the descriptions "number of inquiries in last 6 months" and "number of inquiries in last 6 months (excluding last 7 days)" respectively. Since they have almost the exact same data, this means that their correlation will be almost 1. Other pairs of features with strong correlations have similar descriptions and hence similar data. Here are some regression plots of some of these relationships:



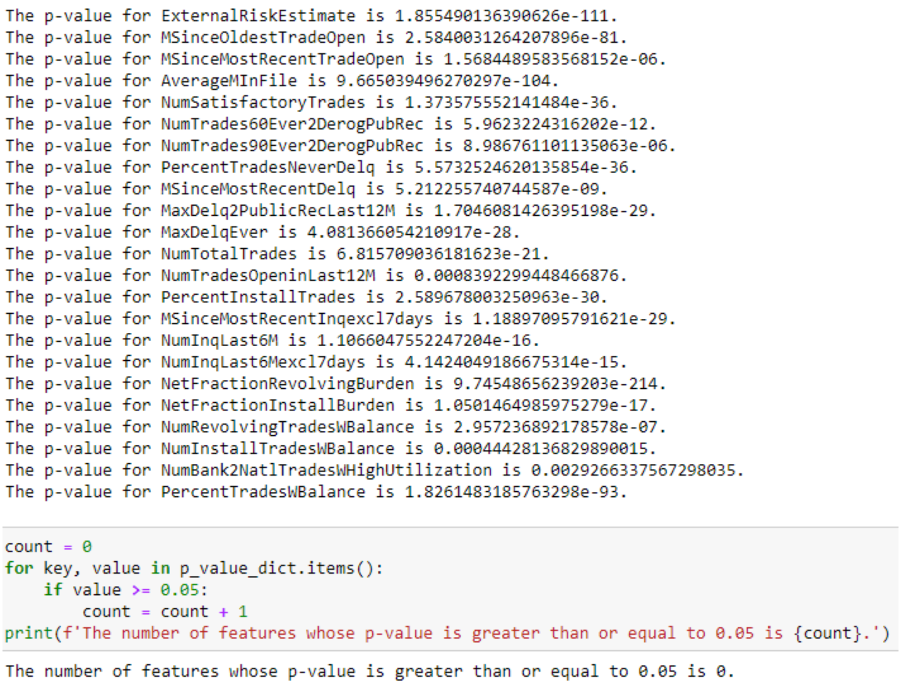


Another aspect worth exploring are the averages of the features grouped by their risk performance. For the following bar plots, we take a look at the average “External Risk Estimate” and “Months Since Oldest Trade Open” between good and bad customers:





It appears as though the averages between good and bad clients for these features are significantly different (significant in the sense that this difference is perhaps not due to chance). To see whether there is a significant difference in the averages of features between good and bad clients, we conduct a t-test for the difference in means between good and bad risk performance. P-values are collected for each feature and examined to see if they are less than 0.05. If it is, we conclude that the difference in mean is significant; otherwise it is not significant. Here are the results of that t-test:



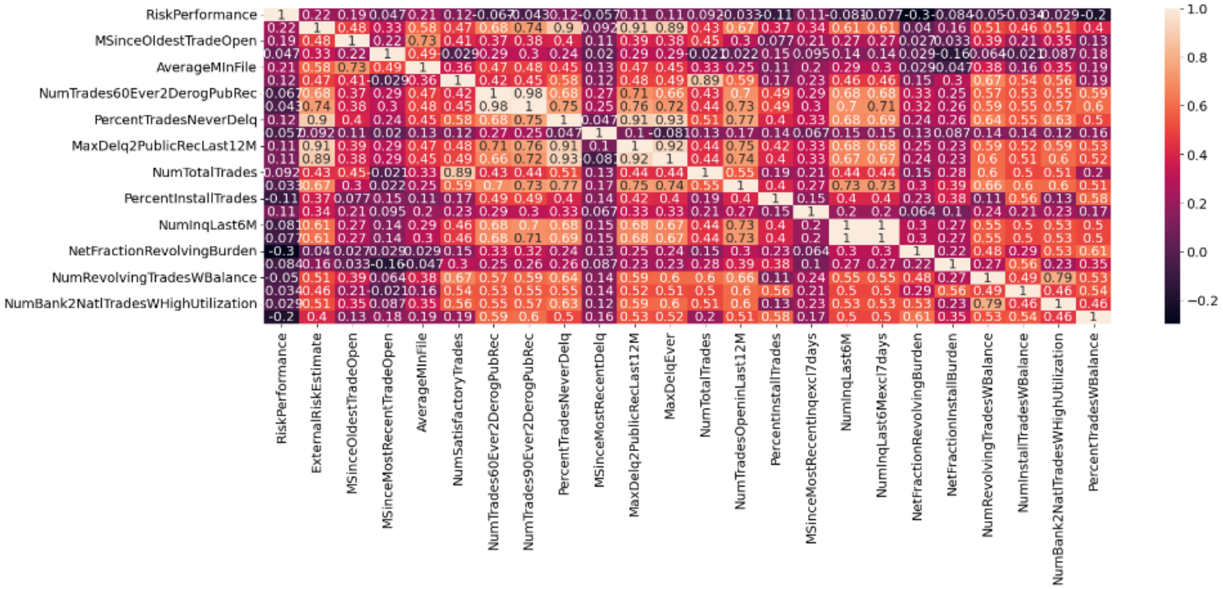
```
The p-value for ExternalRiskEstimate is 1.855490136390626e-111.
The p-value for MSinceOldestTradeOpen is 2.5840031264207896e-81.
The p-value for MSinceMostRecentTradeOpen is 1.5684489583568152e-06.
The p-value for AverageMInFile is 9.665039496270297e-104.
The p-value for NumSatisfactoryTrades is 1.373575552141484e-36.
The p-value for NumTrades60Ever2DerogPubRec is 5.9623224316202e-12.
The p-value for NumTrades90Ever2DerogPubRec is 8.986761101135063e-06.
The p-value for PercentTradesNeverDelq is 5.5732524620135854e-36.
The p-value for MSinceMostRecentDelq is 5.212255740744587e-09.
The p-value for MaxDelq2PublicRecLast12M is 1.7046081426395198e-29.
The p-value for MaxDelqEver is 4.081366054210917e-28.
The p-value for NumTotalTrades is 6.815709036181623e-21.
The p-value for NumTradesOpeninLast12M is 0.0008392299448466876.
The p-value for PercentInstallTrades is 2.589678003250963e-30.
The p-value for MSinceMostRecentInqexcl7days is 1.18897095791621e-29.
The p-value for NumInqLast6M is 1.1066047552247204e-16.
The p-value for NumInqLast6Mexcl7days is 4.1424049186675314e-15.
The p-value for NetFractionRevolvingBurden is 9.74548656239203e-214.
The p-value for NetFractionInstallBurden is 1.0501464985975279e-17.
The p-value for NumRevolvingTradesWBalance is 2.957236892178578e-07.
The p-value for NumInstallTradesWBalance is 0.00044428136829890015.
The p-value for NumBank2NatlTradesWHighUtilization is 0.0029266337567298035.
The p-value for PercentTradesWBalance is 1.826148318576329 8e-93.
```

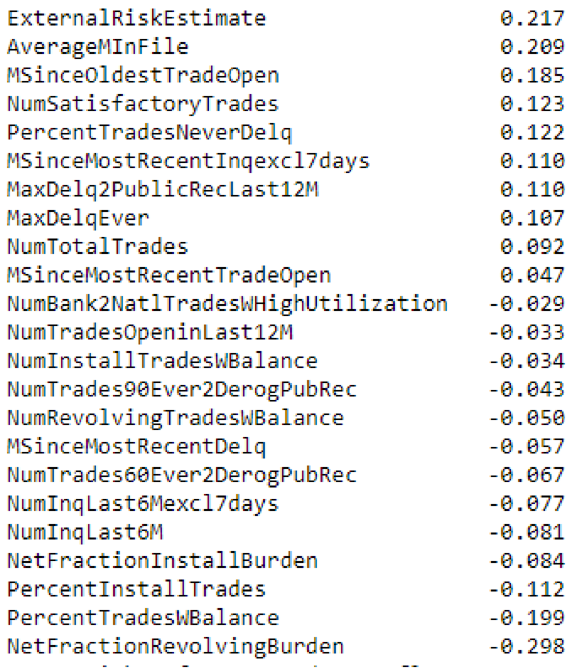
```python
count = 0
for key, value in p_value_dict.items():
    if value >= 0.05:
        count = count + 1
print(f'The number of features whose p-value is greater than or equal to 0.05 is {count}.')
```

```
The number of features whose p-value is greater than or equal to 0.05 is 0.
```

Since there are no features whose p-value is greater than or equal to 0.05, we can conclude that there is a significant difference in the means for every feature partitioned by whether the customer has a good or bad risk performance.

Not only is it important to examine the correlation between features, but also to examine the correlation between the features with the target variable. To do this, we create a heatmap of all of the correlations and examine which features had the highest correlation with risk performance.



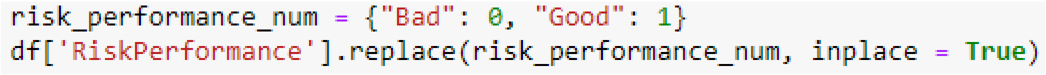


```
ExternalRiskEstimate                   0.217
AverageMInFile                         0.209
MSinceOldestTradeOpen                  0.185
NumSatisfactoryTrades                  0.123
PercentTradesNeverDelq                 0.122
MSinceMostRecentInqexcl7days           0.110
MaxDelq2PublicRecLast12M               0.110
MaxDelqEver                            0.107
NumTotalTrades                         0.092
MSinceMostRecentTradeOpen              0.047
NumBank2NatlTradesWHighUtilization    -0.029
NumTradesOpeninLast12M                -0.033
NumInstallTradesWBalance              -0.034
NumTrades90Ever2DerogPubRec           -0.043
NumRevolvingTradesWBalance            -0.050
MSinceMostRecentDelq                  -0.057
NumTrades60Ever2DerogPubRec           -0.067
NumInqLast6Mexcl7days                 -0.077
NumInqLast6M                          -0.081
NetFractionInstallBurden              -0.084
PercentInstallTrades                  -0.112
PercentTradesWBalance                 -0.199
NetFractionRevolvingBurden            -0.298
```

With the correlations of the features with our target value contained between -0.298 and 0.217, we can say that none of the features are strongly correlated with the risk performance of the clients.

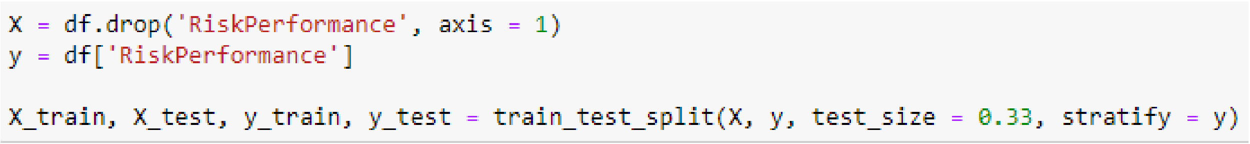
Modeling

By examining the dataset's information, we saw that not only did the dataset have no null values, but also that all but one of the features are numeric values. Our target variable "RiskPerformance" is a categorical variable. Since models cannot be fitted on categorical data, we must encode “RiskPerformance” into numerical values.



```
risk_performance_num = {"Bad": 0, "Good": 1}
df['RiskPerformance'].replace(risk_performance_num, inplace = True)
```

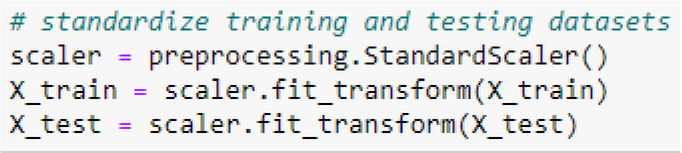
We then set our predictor variables to be all features except for “RiskPerformance,” our target variable to be “RiskPerformance,” and split the data into training and test sets.



```
X = df.drop('RiskPerformance', axis = 1)
y = df['RiskPerformance']

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.33, stratify = y)
```

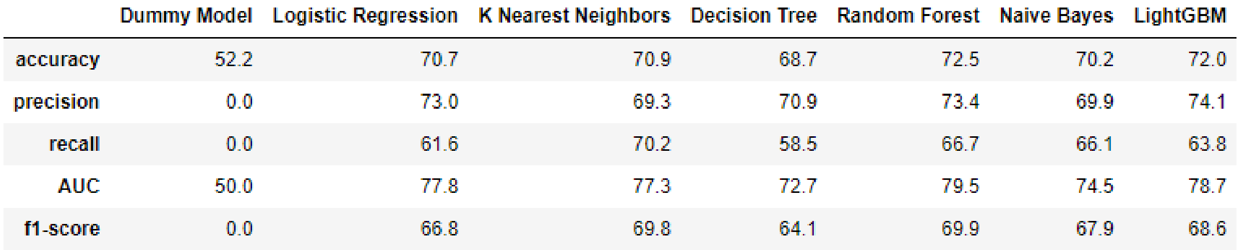
Additionally, we standardize the training and test sets using StandardScaler() so that the data representation is more suitable for these algorithms.



```
# standardize training and testing datasets
scaler = preprocessing.StandardScaler()
X_train = scaler.fit_transform(X_train)
X_test = scaler.fit_transform(X_test)
```

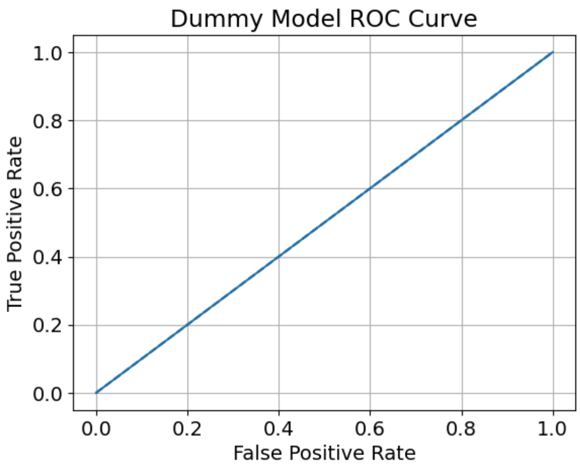
There are several models that we develop and compare with one another: Logistic Regression, K Nearest Neighbors, Decision Tree, Random Forest, Naive Bayes, and LightGBM. We also fit the data on a Dummy Classifier with the purpose of comparing the performance of the other models with the accuracy of the Dummy Model, which in this case predicts that every sample belongs to the majority class. For every model, we either conduct a grid search or random search to find the optimal parameters and fit the models accordingly. Furthermore, if it improved the accuracy of the model, we raised the probability threshold for that model. We raise the probability threshold above 50%, as limiting false positives (that is, limiting the number of times the model classified a client who did not repay their HELOC account as having done so) is more costly than avoiding false negatives. Raising the probability threshold optimizes precision, which in turn limits the number of false positives.

A table with the performance metrics for all of the developed models are shown below:

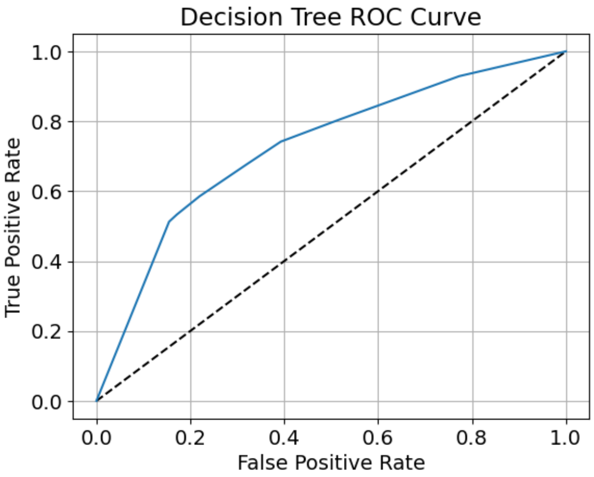


	Dummy Model	Logistic Regression	K Nearest Neighbors	Decision Tree	Random Forest	Naive Bayes	LightGBM
accuracy	52.2	70.7	70.9	68.7	72.5	70.2	72.0
precision	0.0	73.0	69.3	70.9	73.4	69.9	74.1
recall	0.0	61.6	70.2	58.5	66.7	66.1	63.8
AUC	50.0	77.8	77.3	72.7	79.5	74.5	78.7
f1-score	0.0	66.8	69.8	64.1	69.9	67.9	68.6

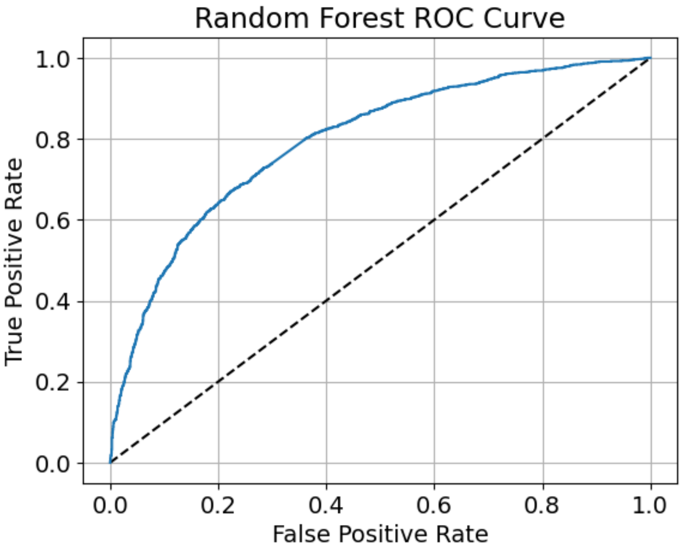
By examining the above table, we see that the most accurate model is Random Forest, with an accuracy of 72.5%. However, we see that there are other models with comparable accuracy scores, namely LightGBM and Logistic Regression. How should we then decide which model to use in order to extract insight from the features? To answer this question, we take a look at the ROC curve for some of the models in the table above.



Dummy Model ROC Curve



Decision Tree ROC Curve



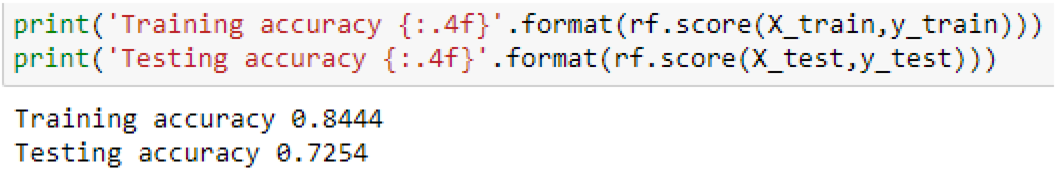
Random Forest ROC Curve

The ROC curve shows the *false positive rate* (FPR) against the *true positive rate* (TPR). The true positive rate is just another name for recall, while the false positive rate is the fraction of false positives out of all negative samples:

$$FPR = \frac{FP}{FP + TN}$$

For the ROC curve, the ideal curve is close to the top left: we want a classifier that produces a *high recall* while keeping a *low false positive rate*. We can summarize the ROC curve using a single number, which is the area under the curve (AUC). The AUC is equivalent to the probability that a randomly picked point of the positive class will have a higher score according to the classifier than a randomly picked point from the negative class. AUC always returns a value between 0 (worst) and 1 (best). Essentially, the higher the AUC, the better the model is at distinguishing between the two classes. Predicting the points randomly always produces an AUC of 0.5, no matter how imbalanced the classes in the dataset are. We can see that with the AUC score for the Dummy Model in the table and how the area under the line in its ROC curve is exactly 0.5. For the decision tree model, its AUC score is higher, as the entire curve is above the dotted line. The model with the best AUC score is the Random Forest model. Since it has the best accuracy and AUC score, we will be using the Random Forest model to make our predictions.

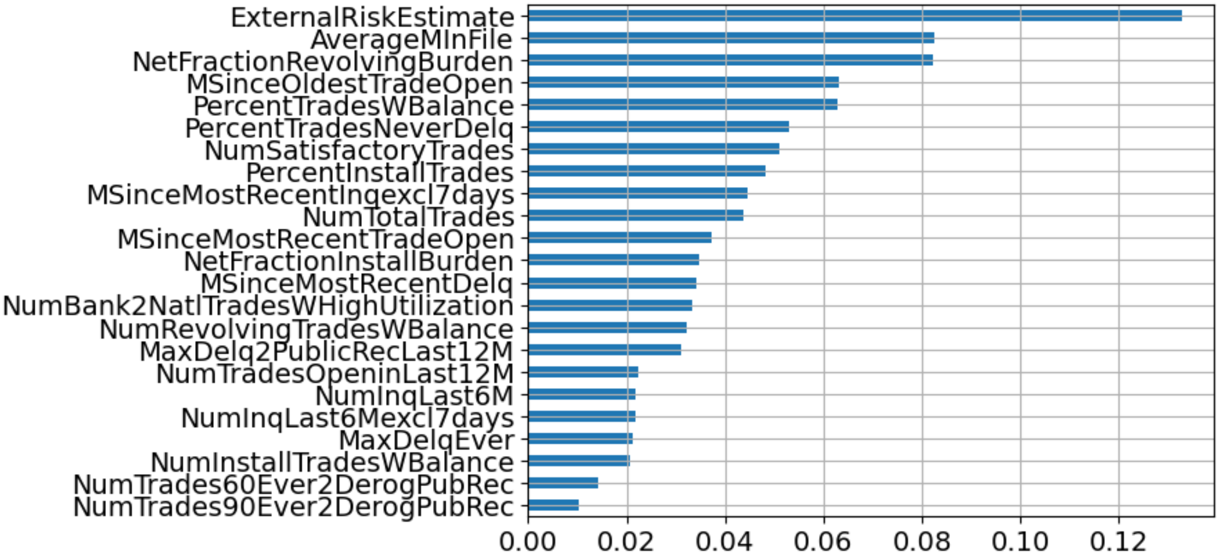
Having a model with an accuracy of 72.5% with an AUC score of 79.5 is an adequate model. Moreover, we can be sure that the model is not overfitting by comparing its training and testing accuracy.



```
print('Training accuracy {:.4f}'.format(rf.score(X_train,y_train)))
print('Testing accuracy {:.4f}'.format(rf.score(X_test,y_test)))
```

```
Training accuracy 0.8444
Testing accuracy 0.7254
```

The testing accuracy isn't significantly less than the training accuracy, which signifies that the model fits the data well. We can therefore feel confident in the validity of our model. However, we do not only desire a model that can predict our target variable, but also to extract information regarding the features used to influence our predicting model. To do that, we take a look at the feature importances of the Random Forest model and order them from greatest to least.



The three most important features are external risk estimate, net fraction involving burden, and average months in file. External risk estimate is by far the most important feature. This makes sense as the higher the risk estimate for a given client, the more likely that they will not pay back their loan in time.

The second most important feature is net fraction involving burden, which is computed by the revolving balance divided by credit limit. To get a better idea of what exactly this feature means, let's take a look at a definition of revolving balance. Revolving credit, such as a credit card, allows a consumer to make purchases up to a certain spending limit and pay down the debt each month. As long as the spending cap has not been reached, the consumer can make purchases using the line of credit. The consumer does not have to pay off the total amount borrowed every month, but any balance that carries over month to month is the revolving balance. Therefore, the net fraction revolving feature is precisely what the revolving balance is divided by the credit limit. In essence, the higher the value is in this feature, the more likely it is that the client will not pay back their loan in time. This is because high revolving balances may indicate that a borrower is relying too much on credit.

The third most important feature is the average number of months in file. In essence, the higher the average number of months the client is on file, the more likely they will not pay back their loan within two years, as those clients who do pay off their loans on time tend to do so earlier rather than later.