A STUDY ON LOAN DEFAULT PREDICTION

Shiladitya Bose

2022-07-17

Contents

INTRODUCTION

Granting credit to customers is the core business of a bank. In doing so, banks need to have adequate systems to decide to whom to grant credit. Credit scoring is a key risk assessment technique to analyze and quantify a potential obligor's credit risk. Essentially, credit scoring aims at quantifying the likelihood that an obligor will repay the debt. The outcome of the credit scoring exercise is a score reflecting the creditworthiness of the obligor. Throughout the past few decades banks have gathered plenty of information describing the default behavior of their customers. Examples are historical information about a customer's date of birth, gender, income, employment status, and so on. All this data has been nicely stored into huge (e.g., relational) databases or data warehouses. On top of this, banks have accumulated lots of business experience about their credit products. As an example, many credit experts do a pretty good job of discriminating between low-risk and high-risk mortgages using their business expertise only. It is now the aim of credit scoring to analyze both sources of data in more detail and come up with a statistically based decision model that allows scoring future credit applications and ultimately deciding which ones to accept and which to reject. For the historical customers, we know which ones turned out to be good payers and which ones turned out to be bad payers. This good/bad status is now the binary target variable Y, which we will relate to all information available at scoring time about our obligors. The goal of credit scoring is now to quantify this relationship as precisely as possible to assist credit decisions, monitoring, and management. Banks score borrowers at loan application, as well as at regular times during the term of a financial contract (generally loans, loan commitments, and guarantees). Once we have our credit scoring model built, we can then use it to decide whether the credit application should be accepted or rejected, or to derive the probability of a future default. To summarize, credit scoring is a key risk management tool for a bank to optimally manage, understand, and model the credit risk it is exposed to.

OBJECTIVE

1. Impute a huge no. of Missing values
2. Replace Categorical variables using WEIGHT OF EVIDENCE(WOE)
3. Checking Collinearity between Categorical variables using CRAMER's V , collinearity between ordinal categorical variables using SPEARMAN's RANK CORRELATION coefficient and collinearity between continuous variables using VIF and also using PEARSON PRODUCT-MOMENT RANK CORRELATION to make a CORRELATION PLOT for diagramatic idea about the mutual correlations among the Continuous variables.
4. Variable selection using STEPWISE selection and INFORMATION VALUE(IV)
5. Using grid search technique for Optimum thresold of Classification Probability
6. Checking Model Adequecy using different Diagonistics
7. Apply Machine Learning technique EXTREME GRADIENT BOOST(XGBOOST) Algorithm for improving the ACCURACY.
8. Determine a thresold of PROBABILITY OF DEFAULT(PD) for Defaulter prediction
9. Build a Simple Credit Scorecard for predict CREDIT SCORE of an individual.

DATASET DESCRIPTION

```
## [1] 7500   16
```

Here our dataset contains 7500 data points and 16 columns.

```
## 'data.frame':    7500 obs. of  16 variables:
##  $ Home.Ownership             : chr  "Own Home" "Own Home" "Home Mortgage" "Own Home" ...
##  $ Annual.Income              : num  482087 1025487 751412 805068 776264 ...
##  $ Years.in.current.job       : chr  NA "10+ years" "8 years" "6 years" ...
##  $ Tax.Liens                  : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ Number.of.Open.Accounts    : num  11 15 11 8 13 12 9 13 17 10 ...
##  $ Years.of.Credit.History    : num  26.3 15.3 35 22.5 13.6 14.6 20.3 12 15.7 24.6 ...
##  $ Maximum.Open.Credit         : num  685960 1181730 1182434 147400 385836 ...
##  $ Number.of.Credit.Problems   : num  1 0 0 1 1 0 0 0 1 0 ...
##  $ Months.since.last.delinquent: num  NA NA NA NA NA NA 73 18 NA 6 ...
##  $ Bankruptcies                : num  1 0 0 1 0 0 0 0 1 0 ...
##  $ Purpose                     : chr  "debt consolidation" "debt consolidation" "debt consolidation"
##  $ Term                        : chr  "Short Term" "Long Term" "Short Term" "Short Term" ...
##  $ Current.Loan.Amount         : num  99999999 264968 99999999 121396 125840 ...
##  $ Current.Credit.Balance      : num  47386 394972 308389 95855 93309 ...
##  $ Monthly.Debt                : num  7914 18373 13651 11338 7180 ...
##  $ Credit.Default              : int  0 1 0 0 0 1 0 1 0 1 ...
```

Here Our Response or Target variable is Credit Default, which contains binary response. "1" stands for Default and "0" stand for Not-Default. And rest of the 14 variables are Expanetory variables.

CHECKING MISSING VALUES AND DUPLICATE VALUES

1.CHECKING DUPLICATE VALUES

```
## [1] 0
```

Our data contains no dulpicate values.

2.CHECKING MISSING VALUES

```
##               Home.Ownership                Annual.Income
##                            0                         1557
##         Years.in.current.job                    Tax.Liens
##                          371                            0
##      Number.of.Open.Accounts      Years.of.Credit.History
##                            0                            0
##          Maximum.Open.Credit    Number.of.Credit.Problems
##                            0                            0
## Months.since.last.delinquent                 Bankruptcies
##                         4081                           14
##                      Purpose                         Term
##                            0                            0
##          Current.Loan.Amount       Current.Credit.Balance
##                            0                            0
##                 Monthly.Debt               Credit.Default
##                            0                            0
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.1 --

## v ggplot2 3.3.6     v purrr   0.3.4
## v tibble  3.1.7     v dplyr   1.0.9
## v tidyr   1.2.0     v stringr 1.4.0
## v readr   2.1.2     v forcats 0.5.1

## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()

##
## Attaching package: 'magrittr'

## The following object is masked from 'package:purrr':
##
```

```
##     set_names
```

```
## The following object is masked from 'package:tidyr':
##
##     extract
```

Hence, our dataset contains a huge number of missing values.

So, now we impute missing values using different techniques.

MISSING VALUE IMPUTATION

Now we check the distributions of missing values according to the Credit Default and Not Default of Missing explanatory variables.

At first, we will see it for Annual Income. In general, the intution may suggest that the Missing rows in Annual Income implies the customers are not an earning person. So, impute these missing rows with 0 is an idea. But, before doing it, we must see the distributions of missing values according to the Credit Default and Not Default.

```
## Warning: package 'sqldf' was built under R version 4.2.1
```

```
## Loading required package: gsubfn
```

```
## Warning: package 'gsubfn' was built under R version 4.2.1
```

```
## Loading required package: proto
```

```
## Warning: package 'proto' was built under R version 4.2.1
```

```
## Loading required package: RSQLite
```

```
## Warning: package 'RSQLite' was built under R version 4.2.1
```

```
## Credit.Default
##    0    1
## 1028  529
```

Here, ratio of defaulters and non-defaulters is 1:2. So, Impute the missing values using 0 is not a good idea here. So, we use here MissForest algorithm for missing value imputation.

MISSFOREST is another machine learning-based data imputation algorithm that operates on the Random Forest algorithm. Stekhoven and Buhlmann, creators of the algorithm, conducted a study in 2011 in which imputation methods were compared on datasets with randomly introduced missing values.

First, the missing values are filled in using median/mode imputation. Then, we mark the missing values as 'Predict' and the others as training rows, which are fed into a Random Forest model trained to predict, in this case, ANNUAL INCOME based on CREDIT DEFAULT. The generated prediction for that row is then filled in to produce a transformed dataset.

This process of looping through missing data points repeats several times, each iteration improving on better and better data. It's like standing on a pile of rocks while continually adding more to raise yourself: the model uses its current position to elevate itself further.

The model may decide in the following iterations to adjust predictions or to keep them the same.

Iterations continue until some stopping criteria is met or after a certain number of iterations has elapsed. As a general rule, datasets become well imputed after four to five iterations, but it depends on the size and amount of missing data.

```
## Warning: package 'missForest' was built under R version 4.2.1
```

Then, we will see it for Bankruptcies. In general, here also the intution may suggest that the Missing rows in Bankruptcies implies the customers are not an earning person. So, bankruptcies is not an issue for him/her. So, impute these missing rows with 0 is an idea. But, before doing it, we must see the distributions of missing values according to the Credit Default and Not Default.

```
## Credit.Default
##  0  1
## 10  4
```

Here, also the ratio of defaulters and non-defaulters is nearly 1:2. So, Impute the missing values using 0 is not a good idea here. So, we use here Apriori algorithm for missing value imputation.

We will now use Association Rule Mining (ARM) method to impute the missing values of the categorical variables.

```
## Loading required package: Matrix
```

```
##
## Attaching package: 'Matrix'
```

```
## The following objects are masked from 'package:tidyr':
##
##     expand, pack, unpack
```

```
##
## Attaching package: 'arules'
```

```
## The following object is masked from 'package:dplyr':
##
##     recode
```

```
## The following objects are masked from 'package:base':
##
##     abbreviate, write
```

```
##
##    0    1    2    3    4
## 6660  786   31    7    2
```

```
##      Home.Ownership Bankruptcies             Purpose       Term
## 101        Own Home         <NA> educational expenses Short Term
## 257            Rent         <NA>   debt consolidation Short Term
## 258   Home Mortgage         <NA>   debt consolidation Short Term
## 899            Rent         <NA>   debt consolidation Short Term
## 1405           Rent         <NA>                other Short Term
## 3064           Rent         <NA>                other Short Term
## 3253           Rent         <NA>        business loan Short Term
## 3352  Home Mortgage         <NA>   debt consolidation Short Term
## 3402           Rent         <NA>                other Short Term
## 3497           Rent         <NA> educational expenses Short Term
## 4335           Rent         <NA>   debt consolidation Short Term
## 5567       Own Home         <NA>                other Short Term
## 7185           Rent         <NA>   debt consolidation Short Term
## 7380       Own Home         <NA>       small business Short Term
##      Current.Loan.Amount Credit.Default
## 101             99999999              A
## 257             99999999              A
## 258               447480              A
## 899               456808              B
## 1405               11242             A
## 3064               44814             B
## 3253              156970             A
## 3352              528968             A
## 3402            99999999             A
## 3497              210166             A
## 4335              167882             A
## 5567               92620             A
## 7185               46706             B
## 7380               71170             B
```

```
##     lhs                             rhs                  support confidence  coverage      lift coun
## [1] {Bankruptcies=A,
##      Term=Short Term}            => {Credit.Default=A} 0.5018702  0.7672044 0.6541544 1.0681220  3757
## [2] {Term=Short Term}            => {Credit.Default=A} 0.5675928  0.7666907 0.7403153 1.0674069  4249
## [3] {Purpose=debt consolidation} => {Credit.Default=A} 0.5725354  0.7217918 0.7932140 1.0048975  4286
## [4] {Bankruptcies=A,
##      Purpose=debt consolidation} => {Credit.Default=A} 0.5080150  0.7212213 0.7043815 1.0041032  3803
## [5] {}                           => {Credit.Default=A} 0.7182741  0.7182741 1.0000000 1.0000000  5377
## [6] {Bankruptcies=A}             => {Credit.Default=A} 0.6387924  0.7180180 0.8896607 0.9996435  4782
```

```
## 
##    0    1    2    3    4 
## 6674  786   31    7    2
```

So, Imputation is good enough.

Next, we will see it for Years in current job. In general, here also the intution may suggest that the Missing rows in Years in current job implies the customers are not an earning person. So, Years in current job is not an issue for him/her. So, impute these missing rows with <1 year is an idea. But, before doing it, we must see the distributions of missing values according to the Credit Default and Not Default.

```
## Credit.Default
##   0   1 
## 234 137
```

Here, also the ratio of defaulters and non-defaulters is nearly 2:3. So, Impute the missing values using <1 year is not a good idea here. So, we use here knn algorithm which use Gower Distance for missing value imputation.

GOWER DISTANCE

One of the most important task while clustering the data is to decide what metric to be used for calculating distance between each data point. In various real-life fields where cluster analysis is commonly used, such as biology, social sciences, or marketing surveys, datasets with both quantitative and categorical variables are often applied. This type of data is referred as mixed data. Many distance metrics exist, and one of them is, the Gower distance (1971) which is used when the data is of Mixed data.

What is Gower's Distance?

Gower's Distance can be used to measure how different two records are. The records may contain combination of logical, categorical, numerical or text data. The distance is always a number between 0 (identical) and 1 (maximally dissimilar). The metrics used for each data type are described below:

1) for Quantitative variable : range-normalized MANHATTAN distance
2) for Ordinal Categorical variable is first ranked, then MANHATTAN distance is used with a special adjustment for ties.

3) for Nominal variables of k categories are first converted into k binary columns and then the DICE COEFFICIENT is used.

```
## Loading required package: colorspace

## Loading required package: grid

## VIM is ready to use.

## Suggestions and bug-reports can be submitted at: https://github.com/statistikat/VIM/issues

##
## Attaching package: 'VIM'

## The following object is masked from 'package:missForest':
##
##     nrmse

## The following object is masked from 'package:datasets':
##
##     sleep

##
##  < 1 year    1 year 10+ years   2 years   3 years   4 years   5 years   6 years
##       563       504      2332       705       620       469       516       426
##   7 years   8 years   9 years
##       396       339       259

##
##  < 1 year    1 year 10+ years   2 years   3 years   4 years   5 years   6 years
##       563       504      2469       705       620       469       516       426
##   7 years   8 years   9 years
##       396       573       259
```

So, Imputation is good enough.

Then, we will see it for Months Since Last Delinquent. In general, here also the intution may suggest that the Missing rows in Months Since Last Delinquent implies the customers are not an earning person. So, Months Since Last Delinquent is not an issue for him/her. So, impute these missing rows with a very high value, say, 130 is an idea. But, before doing it, we must see the distributions of missing values according to the Credit Default and Not Default.

```
## Credit.Default
##    0    1
## 2951 1130
```

Here, also the ratio of defaulters and non-defaulters is nearly 2:5. So, Impute the missing values using 130 is not a good idea here. So, also we use here knn algorithm which use Gower Distance for missing value imputation.

```
##
##   0   1   2   3   4   5   6   7   8   9  10  11  12  13  14  15  16  17  18  19
##  18  26  25  30  31  51  64  64  68  61  63  51  65  65  76  48  61  58  58  65
##  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39
##  54  47  52  42  59  54  56  46  45  71  53  51  51  68  55  59  46  51  63  49
##  40  41  42  43  44  45  46  47  48  49  50  51  52  53  54  55  56  57  58  59
##  48  50  43  45  36  50  46  37  44  25  39  19  26  34  36  36  23  29  24  32
##  60  61  62  63  64  65  66  67  68  69  70  71  72  73  74  75  76  77  78  79
##  32  36  23  33  26  28  17  22  36  26  22  30  24  21  25  24  23  21  29  20
##  80  81  82  83  84  86  91  92 118
##  28  19   4   3   1   1   1   1   1
```

After Imputation

```
## 
##   0   1   2   3   4   5   6   7   8   9  10  11  12  13  14  15  16  17  18  19 
##  18  26  25  30  32  51  66  80  75  77  82  74  96 134 131  91 134 128 123 155 
##  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39 
## 141  96 132 119 148 164 159 122 121 225 673 152 125 274 150 227 153 161 185 144 
##  40  41  42  43  44  45  46  47  48  49  50  51  52  53  54  55  56  57  58  59 
## 132 124  95 102  98 134  88  75 109  39  65  35  50  62  65  52  32  55  42  47 
##  60  61  62  63  64  65  66  67  68  69  70  71  72  73  74  75  76  77  78  79 
##  50  44  32  46  37  37  33  32  48  26  22  37  24  30  25  24  23  21  30  20 
##  80  81  82  83  84  86  91  92 118 
##  28  19   4   3   1   1   1   1   1
```

```
##              Home.Ownership                   Tax.Liens 
##                           0                           0 
##     Number.of.Open.Accounts      Years.of.Credit.History 
##                           0                           0 
##         Maximum.Open.Credit   Number.of.Credit.Problems 
##                           0                           0 
##                Bankruptcies                     Purpose 
##                           0                           0 
##                        Term         Current.Loan.Amount 
##                           0                           0 
##      Current.Credit.Balance                Monthly.Debt 
##                           0                           0 
##              Credit.Default               Annual_Income 
##                           0                           0 
##        Years.in.current.job Months.since.last.delinquent 
##                           0                           0
```

So, we complete our data missing values imputation.

Now our data is free from Missing values and go for further analysis.

```
## 
##    0    1 
## 5387 2113
```

So, I deal with a Imbalance data.

Here, Accuracy is not a good metric to use when you have class imbalance.

This may be good enough for a well-balanced class but not ideal for the imbalanced class problem. The other metrics such as precision is the measure of how accurate the classifier's prediction of a specific class and recall is the measure of the classifier's ability to identify a class.

For an imbalanced class dataset F1 score is a more appropriate metric. It is the harmonic mean of precision and recall and the expression is

So, if the classifier predicts the minority class but the prediction is erroneous and false-positive increases, the precision metric will be low and so as F1 score. Also, if the classifier identifies the minority class poorly, i.e. more of this class wrongfully predicted as the majority class then false negatives will increase, so recall and F1 score will low. F1 score only increases if both the number and quality of prediction improves.

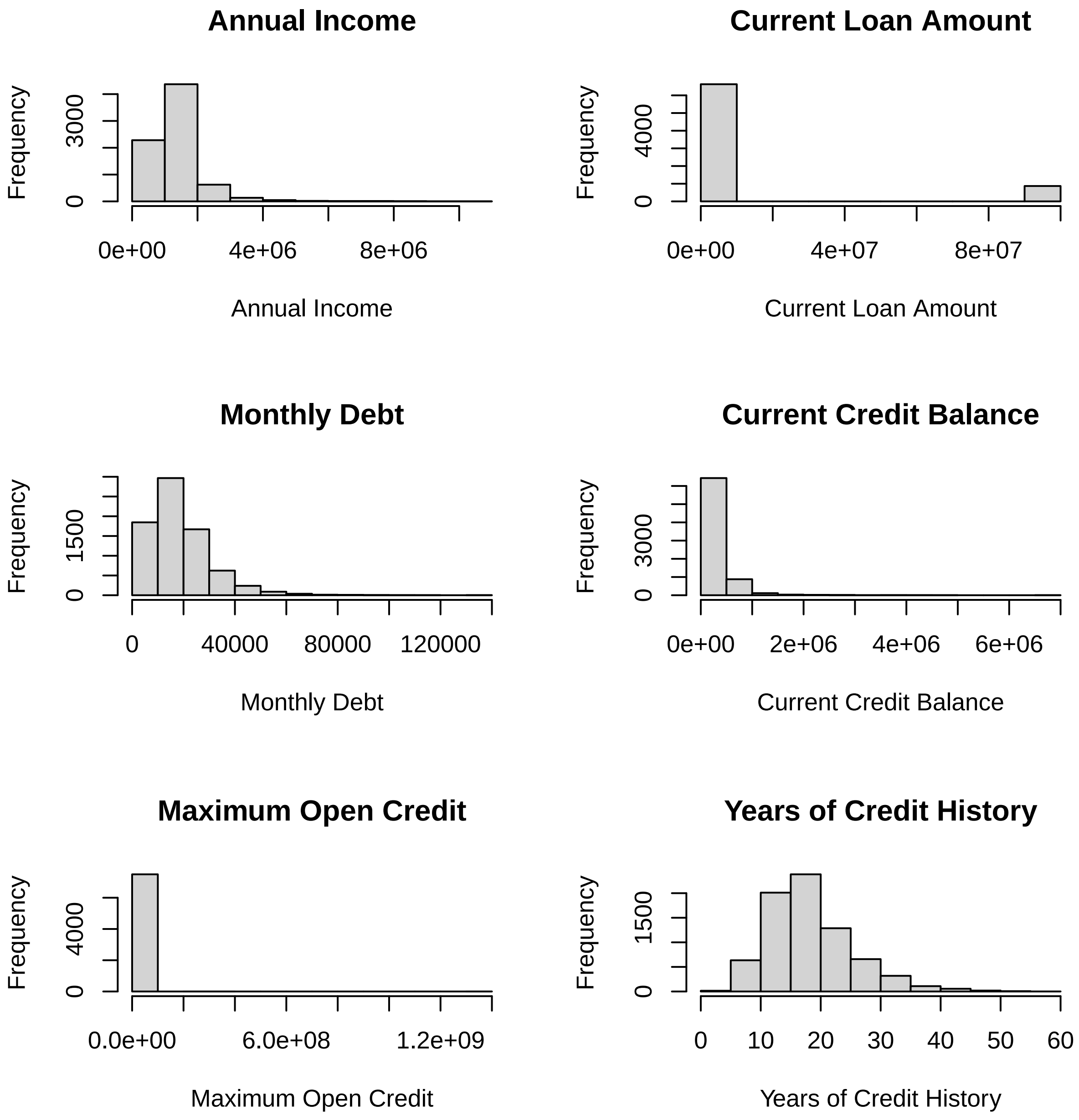
F1 score keeps the balance between precision and recall and improves the score only if the classifier identifies more of a certain class correctly.

F1 score ranges from 0 to 1, where 1 is a perfect score indicating that the model predicts each observation correctly. A good f1 score is of course dependent on the data you are working with and the use case, for example a model predicting cancer would have a very different expectation than an abandoned cart model. However, there is a general rule of thumb when it comes to f1 scores, which is as follows:

1) if F1 Score $>$ 0.9, model is Very good
2) if F1 Score is b/w 0.8 - 0.9, model is Good
3) if F1 Score is b/w 0.5 - 0.8, model is OK
4) if F1 Score is$<$ 0.5, model is Not good

So, I focus more on F1 Score of the model

NOW WE CHECK GRAPHICAL PREVIEW OF CONTINUOUS COLUMN & CHECK THEIR CHARACTERISTICS



All of these continuous columns are right skewed except the YEARS OF CREDIT HISTORY.

SPLITTING THE INTO TRAIN AND TEST

We split the dataset into (4:1) ratio for train and test.

```
## [1] 5625   16
```

```
## [1] 1875   16
```

VARIABLE SELECTIONS

An Important Technical Aspect of Developing Logistic Regression: Variable Selection

Variable selection aims at reducing the number of variables in a model. It will make the model more concise and faster to evaluate. Logistic regression has a built-in procedure to perform variable selection. It is based on a statistical hypothesis test to verify whether the coefficient of a variable included in the model is significantly different from zero.

In credit scoring, it is very important to be aware that statistical significance is only one evaluation criterion to consider in doing variable selection. As mentioned before, interpretability is also an important criterion (Martens et al. 2007). In logistic regression, this can be easily evaluated by inspecting the sign of the regression coefficient. It is highly preferable that a coefficient has the same sign as anticipated by the credit expert; otherwise he or she will be reluctant to use the model. Coefficients can have unexpected signs due to multicollinearity issues, noise, or small sample effects. Sign restrictions can be easily enforced in a forward regression setup by preventing variables with the wrong sign from entering the model.

Legal issues also need to be properly taken into account. For example, in the United States, there is the Equal Credit Opportunity Act, which states that no one is allowed to dis- criminate based on gender, age, ethnic origin, nationality, beliefs, and so on. These variables must not be included in a credit scorecard. Other countries have other regulations, and it is important to be aware of this.

NOW CHECKING CORRELATION AMONG NOMINAL CATEGORICAL VARIABLES

Cramer's V is used to examine the association between two categorical variables when there is more than a 2 X 2 contingency (e.g., 2 X 3). In these more complicated designs, phi is not appropriate, but Cramer's statistic is. Cramer's V represents the association or correlation between two variables.

```
## Cramer V
##  0.09922
```

```
## Cramer V
##   0.3557
```

```
## Cramer V
##   0.1316
```

```
## Cramer V
##  0.06958
```

```
## Cramer V
##  0.09414
```

```
## Cramer V
##  0.05555
```

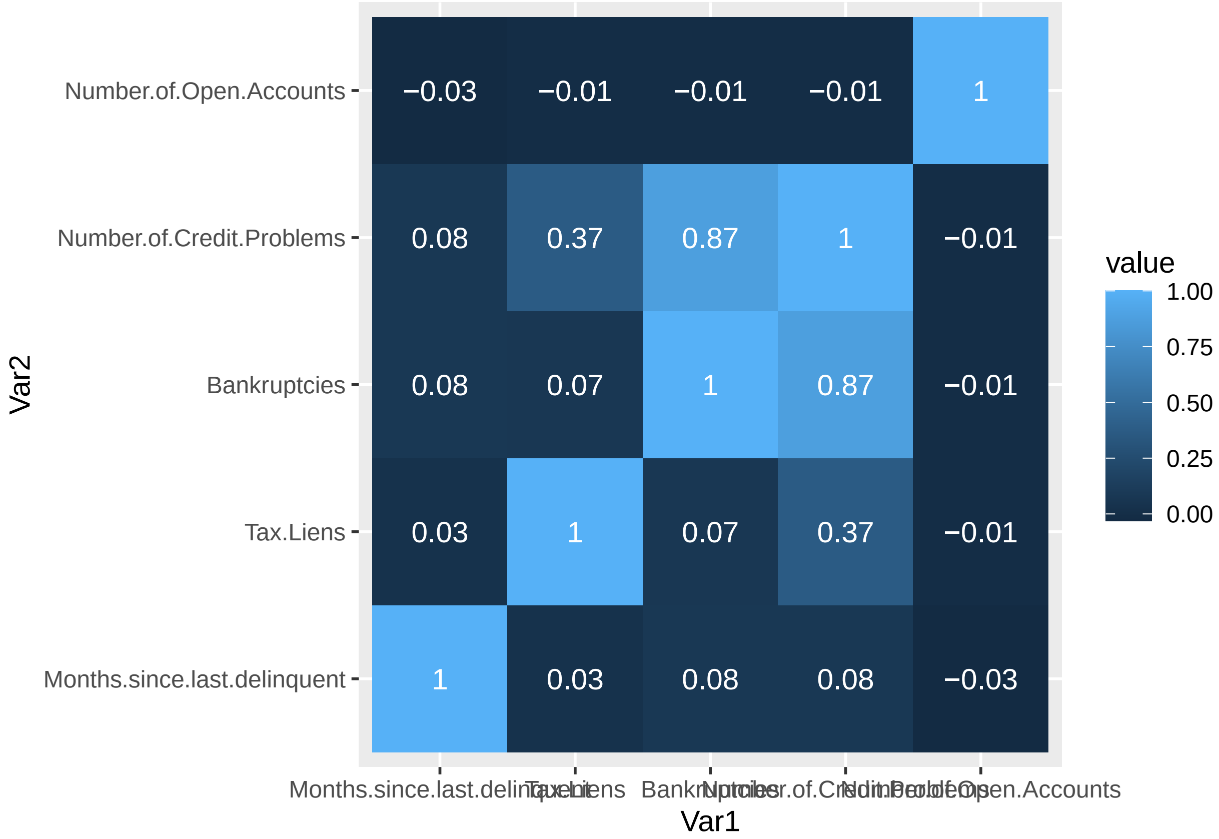
The correlation are not too high. So, we retain all Nominal Categorical columns.

NOW CHECKING CORRELATION AMONG ORNINAL CATEGORICAL VARIABLES

Here we use Spearman Rank Correlation coefficient.

```
##
## Attaching package: 'reshape2'
```

```
## The following object is masked from 'package:tidyr':
##
##     smiths
```



Here, We can see the correlation between Number of Credit Problem and Bankruptcies is significantly high. So, We drop one column among them. We retain Bankruptcies but drop Number of Credit Problems, as we think Bankruptcies is much significant in Credit default problem than Number of Credit Problems.

REPLACE CATEGORICAL VARIABLES BY WOE

WEIGHT OF EVIDENCE (WOE) AND INFORMATION VALUE(IV)

Prior to building a binary classification model (e.g., Logistic Regression, etc.), a common step is to perform variable screening and exploratory data analysis. This is the step where we get to know the data and weed out variables that are either ill-conditioned or simply contain no information that will help us predict the action of interest. Note that the purpose of this step should not to be confused with that of multiple-variable selection techniques, such as stepwise regression, where the variables that go into the final model are selected. Rather, this is a precursory step designed to ensure that the approaches deployed during the final modeling phases are set up for success.

The weight of evidence (WOE) and information value (IV) provide a great framework for for exploratory analysis and variable screening for binary classifiers. WOE and IV have been used extensively in the credit risk world for several decades, and the underlying theory dates back to the 1950s.

WOE and IV are simple, yet powerful techniques to perform variable transformation and selection. These concepts have huge connection with the logistic regression modeling technique. It is widely used in credit scoring to measure the separation of good vs bad customers. In addition, the advantages of WOE transformation are:

1)Handles missing values.

2)Handles outliers.

The transformation is based on logarithmic value of distributions. This is aligned with the logistic regression output function No need for dummy variables.

By using proper binning technique, it can establish monotonic relationship (either increase or decrease) between the independent and dependent variable

According to Baesens et al. (2016) and Siddiqi (2012), WOE and IV analysis enable one to:

1)Consider each variable's independent contribution to the outcome.

2)Detect linear and non-linear relationships.

3)Rank variables in terms of "univariate" predictive strength.

4)Visualize the correlations between the predictive variables and the binary outcome.

5)Seamlessly compare the strength of continuous and categorical variables without creating dummy variables.

6)Seamlessly handle missing values without imputation.

7)Assess the predictive power of missing values.

By convention the values of the IV statistic for variable selec tion can be used as follows:

1)Less than 0.02: the predictor is not useful for modeling (separating the Goods from the Bads).

2)From 0.02 to 0.1: the predictor has only a weak relationship to the Goods/Bads odds ratio.

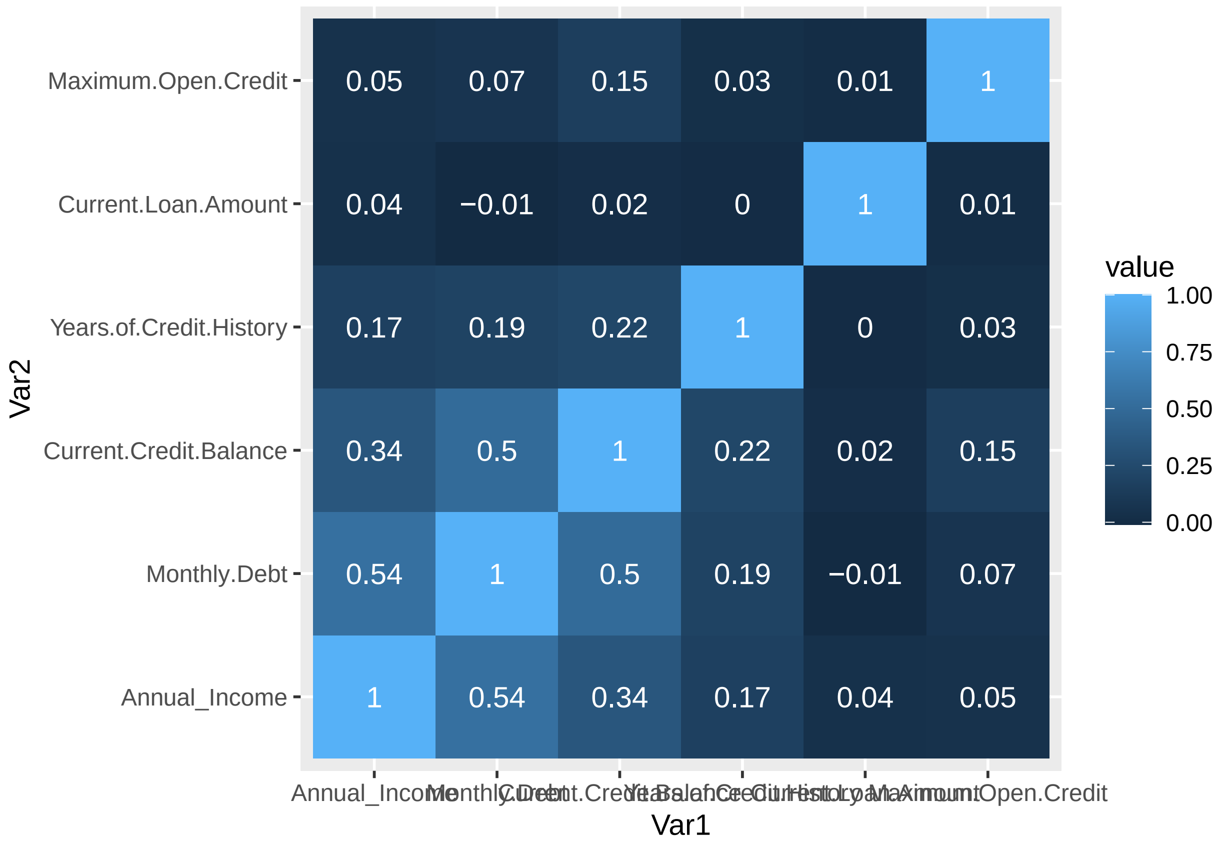
3)From 0.1 to 0.3: the predictor has a medium strength relationship to the Goods/Bads odds ratio.

4)From 0.3 to 0.5: the predictor has a strong relationship to the Goods/Bads odds ratio.

5)Higher than 0.5: we should check carefully when selecting variables have IV higher than 0.5 because of suspicious relationship.

```
##
## Attaching package: 'scorecard'

## The following object is masked from 'package:tidyr':
##
##     replace_na

## [INFO] creating woe binning ...

## [INFO] converting into woe values ...

## [INFO] converting into woe values ...
```

NOW CHECKING CORRELATION AMONG CONTINUOUS VARIABLES



From the correlation matrix, the correlations are not much high to infer that any 2 column have correlation among them. For better understand about the correlation, we use VARIANCE INFLATION FACTOR.

VARIANCE INFLATION FACTOR(VIF)

Now we will investigate whether the continuous regressors in our dataset are involved in multicollinearity or not. In a regression problem with multiple regressors , multicollinearity refers to a near-linear relationship among the regressors. Multicollinearity may happen due to overspecification of model, bad data collection or sampling techniques, inclusion of too many higher order terms in a polynomial regression model etc. Multicollinearity has some serious consequences eg. exceptionally high value of parameter estimates, large variances of some parameter estimators. Several multicollinearity diagnostic measures are available. Here we have used “Variance Inflation Factor” to detect multicollinearity among the continuous variables of our dataset. The variance inflation factor for the jth explanatory variable (when all the regressors are scaled to unit norm) is defined as:

$$VIF_j = \frac{1}{1 - R_j^2}$$

where,

R_j^2 denotes the coefficient of determination obtained when X_j is regressed on the remaining regressor variables.

In practice, usually a VIF > 5 indicates that the corresponding explanatory variable is involved in multicollinearity. Here we will use an iterative algorithm that drops variable with highest VIF and then checks V IF again and then drop until VIF of all variables is less than 5.

```
## Loading required package: carData

##
## Attaching package: 'car'

## The following object is masked from 'package:scorecard':
##
##     vif

## The following object is masked from 'package:arules':
##
##     recode

## The following object is masked from 'package:dplyr':
##
##     recode

## The following object is masked from 'package:purrr':
##
##     some

##           Annual_Income            Monthly.Debt  Current.Credit.Balance
##                1.542739                1.817614                3.460311
## Years.of.Credit.History     Current.Loan.Amount     Maximum.Open.Credit
##                1.080528                1.002212                3.106613
```

So, VIFs are less than 5. So, we infer that the continuous columns are not mutually correlated.

Logistic Regression for Developing a PD Model

Logistic regression is a very popular credit scoring classification technique due to its simplicity and good performance. Just as with linear regression, once the parameters have been estimated, the regression can be evaluated in a straightforward way, contributing to its operational efficiency. From an interpretability viewpoint, it can be easily transformed into an interpretable, user-friendly, points-based credit scorecard.

Next we use STEPWISE SELECTION.

```
## Start:  AIC=5887.09
## Y ~ Tax.Liens + Number.of.Open.Accounts + Years.of.Credit.History +
##     Maximum.Open.Credit + Bankruptcies + Current.Loan.Amount +
##     Current.Credit.Balance + Monthly.Debt + Annual_Income + Months.since.last.delinquent +
##     Home.Ownership_woe + Purpose_woe + Term_woe + Years.in.current.job_woe
##
##                                Df Deviance    AIC
## - Years.of.Credit.History       1   5857.1 5885.1
## - Bankruptcies                  1   5857.2 5885.2
## - Months.since.last.delinquent  1   5857.7 5885.7
## <none>                              5857.1 5887.1
## - Current.Credit.Balance        1   5864.0 5892.0
## - Purpose_woe                   1   5866.4 5894.4
## - Number.of.Open.Accounts       1   5867.8 5895.8
## - Tax.Liens                     1   5869.3 5897.3
## - Monthly.Debt                  1   5882.9 5910.9
## - Maximum.Open.Credit           1   5883.2 5911.2
## - Home.Ownership_woe            1   5885.0 5913.0
## - Years.in.current.job_woe      1   5894.4 5922.4
## - Annual_Income                 1   5965.4 5993.4
## - Term_woe                      1   6016.0 6044.0
## - Current.Loan.Amount           1   6271.5 6299.5
##
## Step:  AIC=5885.1
## Y ~ Tax.Liens + Number.of.Open.Accounts + Maximum.Open.Credit +
##     Bankruptcies + Current.Loan.Amount + Current.Credit.Balance +
##     Monthly.Debt + Annual_Income + Months.since.last.delinquent +
##     Home.Ownership_woe + Purpose_woe + Term_woe + Years.in.current.job_woe
##
##                                Df Deviance    AIC
## - Bankruptcies                  1   5857.2 5883.2
## - Months.since.last.delinquent  1   5857.7 5883.7
## <none>                              5857.1 5885.1
## + Years.of.Credit.History       1   5857.1 5887.1
## - Current.Credit.Balance        1   5864.0 5890.0
## - Purpose_woe                   1   5866.4 5892.4
## - Number.of.Open.Accounts       1   5867.9 5893.9
## - Tax.Liens                     1   5869.3 5895.3
## - Monthly.Debt                  1   5883.0 5909.0
## - Maximum.Open.Credit           1   5883.3 5909.3
## - Home.Ownership_woe            1   5885.5 5911.5
## - Years.in.current.job_woe      1   5894.4 5920.4
## - Annual_Income                 1   5965.7 5991.7
## - Term_woe                      1   6016.0 6042.0
## - Current.Loan.Amount           1   6271.5 6297.5
##
## Step:  AIC=5883.2
## Y ~ Tax.Liens + Number.of.Open.Accounts + Maximum.Open.Credit +
##     Current.Loan.Amount + Current.Credit.Balance + Monthly.Debt +
##     Annual_Income + Months.since.last.delinquent + Home.Ownership_woe +
##     Purpose_woe + Term_woe + Years.in.current.job_woe
##
##                                Df Deviance    AIC
## - Months.since.last.delinquent  1   5857.9 5881.9
## <none>                              5857.2 5883.2
## + Bankruptcies                  1   5857.1 5885.1
## + Years.of.Credit.History       1   5857.2 5885.2
## - Current.Credit.Balance        1   5864.3 5888.3
## - Purpose_woe                   1   5866.7 5890.7
## - Number.of.Open.Accounts       1   5867.9 5891.9
## - Tax.Liens                     1   5869.3 5893.3
## - Monthly.Debt                  1   5883.1 5907.1
## - Maximum.Open.Credit           1   5883.3 5907.3
## - Home.Ownership_woe            1   5885.7 5909.7
## - Years.in.current.job_woe      1   5894.6 5918.6
## - Annual_Income                 1   5965.8 5989.8
## - Term_woe                      1   6016.4 6040.4
## - Current.Loan.Amount           1   6271.6 6295.6
##
## Step:  AIC=5881.89
## Y ~ Tax.Liens + Number.of.Open.Accounts + Maximum.Open.Credit +
##     Current.Loan.Amount + Current.Credit.Balance + Monthly.Debt +
##     Annual_Income + Home.Ownership_woe + Purpose_woe + Term_woe +
##     Years.in.current.job_woe
##
```

```
##                                Df Deviance    AIC
## <none>                              5857.9 5881.9
## + Months.since.last.delinquent  1   5857.2 5883.2
## + Bankruptcies                  1   5857.7 5883.7
## + Years.of.Credit.History       1   5857.9 5883.9
## - Current.Credit.Balance        1   5864.9 5886.9
## - Purpose_woe                   1   5867.4 5889.4
## - Number.of.Open.Accounts       1   5868.6 5890.6
## - Tax.Liens                     1   5869.8 5891.8
## - Monthly.Debt                  1   5883.7 5905.7
## - Maximum.Open.Credit           1   5883.9 5905.9
## - Home.Ownership_woe            1   5886.3 5908.3
## - Years.in.current.job_woe      1   5895.2 5917.2
## - Annual_Income                 1   5966.0 5988.0
## - Term_woe                      1   6017.4 6039.4
## - Current.Loan.Amount           1   6271.7 6293.7
```

```
##
## Call:  glm(formula = Y ~ Tax.Liens + Number.of.Open.Accounts + Maximum.Open.Credit +
##     Current.Loan.Amount + Current.Credit.Balance + Monthly.Debt +
##     Annual_Income + Home.Ownership_woe + Purpose_woe + Term_woe +
##     Years.in.current.job_woe, family = binomial, data = N)
##
## Coefficients:
##             (Intercept)                Tax.Liens   Number.of.Open.Accounts
##              -4.024e-01                4.164e-01                 2.450e-02
##     Maximum.Open.Credit      Current.Loan.Amount    Current.Credit.Balance
##              -4.513e-07               -7.104e-08                 5.753e-07
##            Monthly.Debt            Annual_Income        Home.Ownership_woe
##               2.043e-05               -6.644e-07                 9.723e-01
##             Purpose_woe                 Term_woe  Years.in.current.job_woe
##               9.296e-01                1.083e+00                 9.896e-01
##
## Degrees of Freedom: 5624 Total (i.e. Null);  5613 Residual
## Null Deviance:       6723
## Residual Deviance: 5858  AIC: 5882
```

Now, we check the the selected model is nearly the saturated model or not, using deviance statistics.

```
## [1] TRUE
```

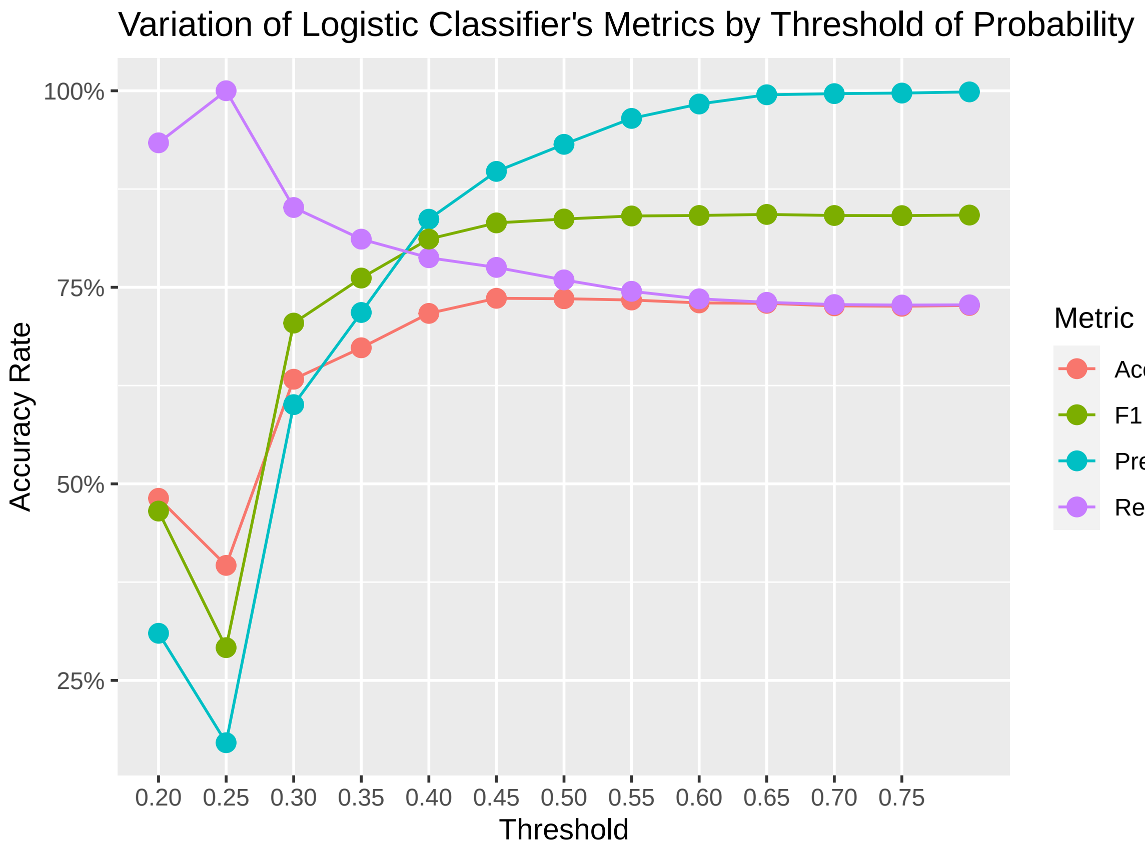
So, The Selected model is good.

CALCULATE PD (SEARCH OPTIMAL THRESOLD OF PROBABILITY THAT MAXIMIZE ACCURACY)

Accuracy criterion is widely used for evaluating model performance in context of credit scoring. However this measure is totally affected by threshold of probability that we select for classifier. I use grid search method for finding optimal threshold that maximizes accuracy, As, I deal with Imbalance data, the optimum thresold may not be 0.5. So, I will find the Optimum thresold. The graph shown below:

```
## Loading required package: lattice
```

```
##
## Attaching package: 'caret'
## The following object is masked from 'package:purrr':
##
##     lift
```



Here, 0.5 is the best for the thresold.

Now, the we construct the confusion matrix,

```
##
## fraud_or_not_STEP    0    1
##                 0 1272  403
##                 1   93  107
```

```
##  Accuracy
## 0.7354667
```

```
## Precision
##  0.759403
```

```
##    Recall
## 0.9318681
```

```
##        F1
## 0.8368421
```

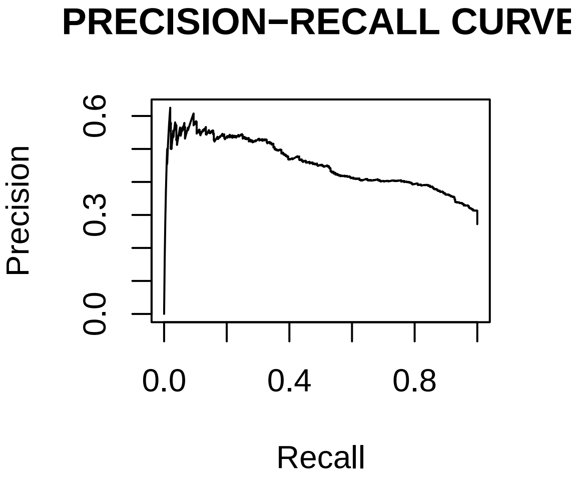
The F1-Score of the model is 0.8384. So, our model is GOOD.

Now we check the PRECISION-RECALL curve.

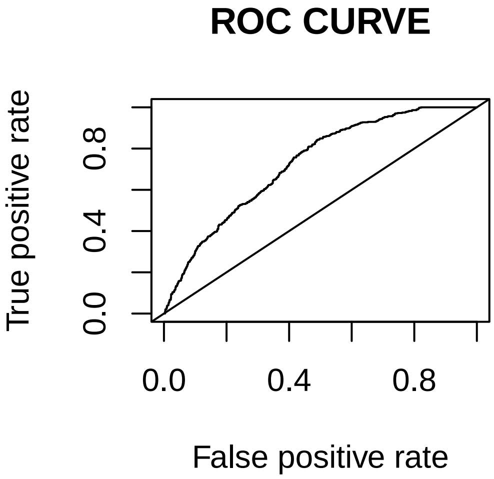
```
## Type 'citation("pROC")' for a citation.
```

```
##
```

```
## Attaching package: 'pROC'

## The following object is masked from 'package:colorspace':
##
##     coords

## The following objects are masked from 'package:stats':
##
##     cov, smooth, var
```



Now, we check the ROC curve



The AUC is

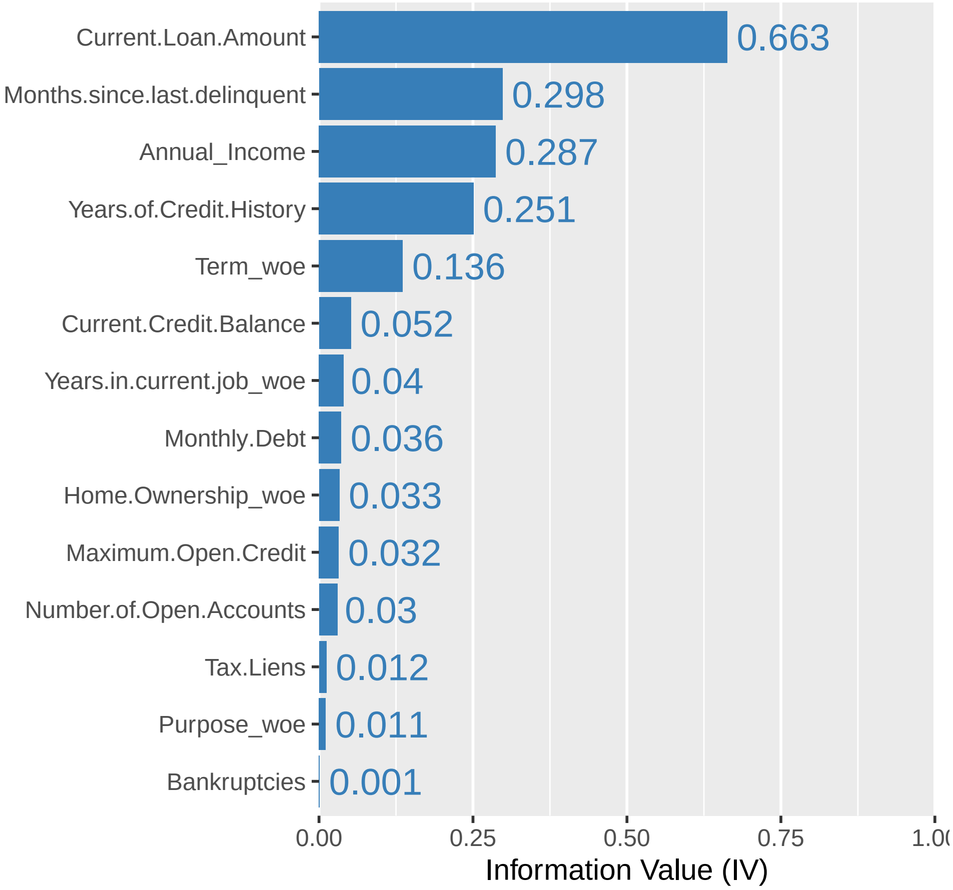
```
## Setting levels: control = 0, case = 1

## Setting direction: controls < cases

## Area under the curve: 0.7266
```

Now, I use the variable selection using INFORMATION VALUE.

```
##                       variable   info_value
##  1:          Current.Loan.Amount 0.6630385201
##  2: Months.since.last.delinquent 0.2982963591
##  3:                Annual_Income 0.2868380680
##  4:       Years.of.Credit.History 0.2509911204
##  5:                     Term_woe 0.1358883134
##  6:        Current.Credit.Balance 0.0524097701
##  7:      Years.in.current.job_woe 0.0402699763
##  8:                 Monthly.Debt 0.0358296516
##  9:            Home.Ownership_woe 0.0326084409
## 10:           Maximum.Open.Credit 0.0324787293
## 11:       Number.of.Open.Accounts 0.0301734740
## 12:                   Tax.Liens 0.0117135874
## 13:                 Purpose_woe 0.0114600038
## 14:                 Bankruptcies 0.0008856759
```

The Graphical view of IV values

Figure 7: Information Value (IV) for All Va



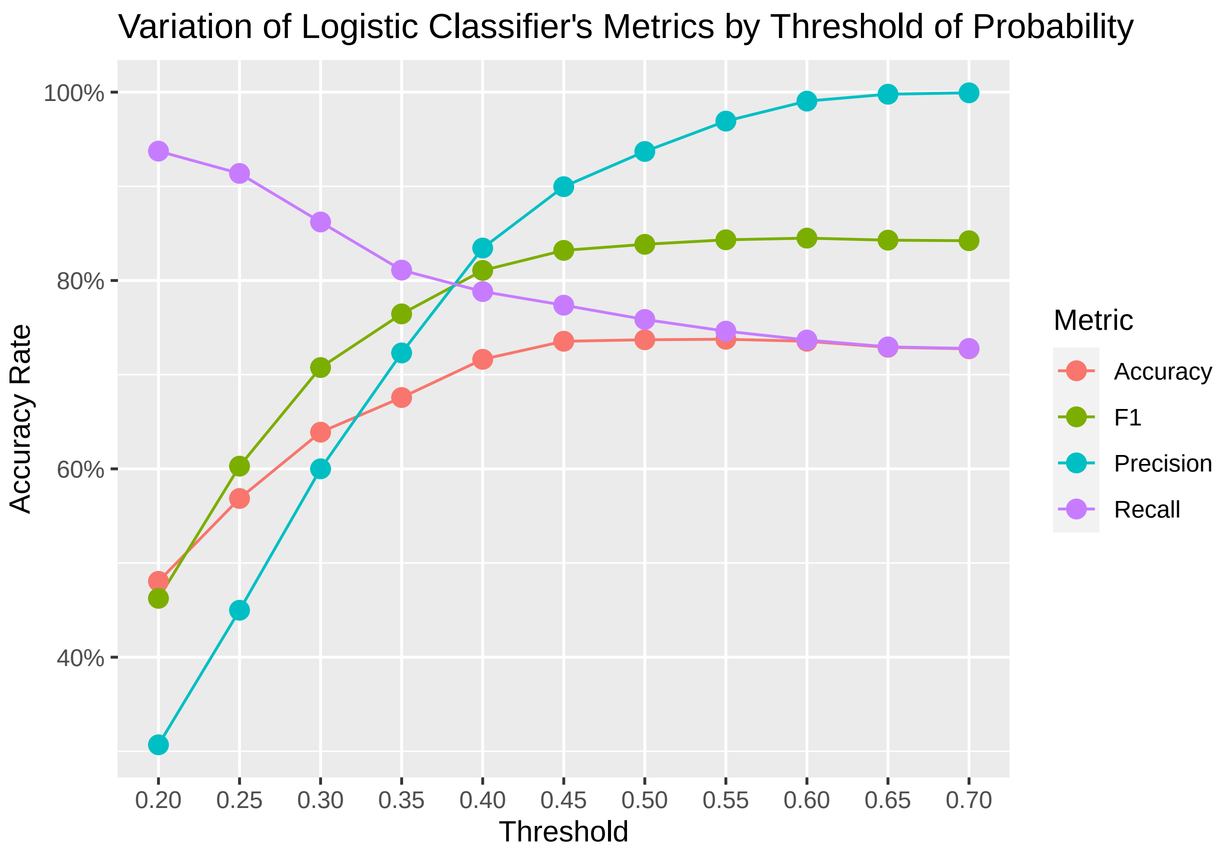
From the graph we can see the IV value of TAX LIENS, PURPOSE and BANKRUPTCIES are less than 0.02, According to SIDDIQI, these 3 are not useful for prediction purpose, so we remove these 3 variables from my model.

Next come to CURRENT LOAN AMOUNT whose IV is greater than 0.5. According to SIDDIQI, this indicates a suspicious relationship. This idicates a LEAKAGE, that means this variable may be a representative of Dependent variable. But in our case the IV value of CURRENT LOAN AMOUNT not too much from 0.05 thresold and for loan default prediction purpose, CUURENT LOAN AMOUNT plays an important role, So, I consider this variable in the model.

```
##
## Call:  glm(formula = Y ~ ., family = binomial, data = Train_IV_model)
##
## Coefficients:
##                 (Intercept)            Home.Ownership_woe
##                  -3.508e-01                     1.043e+00
## Months.since.last.delinquent      Years.in.current.job_woe
##                  -1.244e-03                     9.798e-01
##               Annual_Income                   Monthly.Debt
##                  -6.506e-07                     2.031e-05
##       Current.Credit.Balance           Current.Loan.Amount
##                   5.396e-07                    -7.157e-08
##                    Term_woe          Maximum.Open.Credit
##                   1.078e+00                    -4.451e-07
##      Years.of.Credit.History       Number.of.Open.Accounts
##                   5.905e-04                     2.312e-02
##
## Degrees of Freedom: 5624 Total (i.e. Null);  5613 Residual
## Null Deviance:       6723
## Residual Deviance: 5879  AIC: 5903
```

CALCULATE PD (SEARCH OPTIMAL THRESOLD OF PROBABILITY THAT MAXIMIZE ACCURACY)

Accuracy criterion is widely used for evaluating model performance in context of credit scoring. However this measure is totally affected by threshold of probability that we select for classifier. I use simulation method for finding optimal threshold that maximizes accuracy. As, I deal with Imbalance data, the optimum thresold may not be 0.5. So, I will find the Optimum thresold. The graph shown below:



from the graph 0.6 be the best choice for thresold.

Now, the we construct the confusion matrix,

```
##
## fraud_or_not_IV    0    1
##               0 1352  483
##               1   13   27
```

```
##  Accuracy
## 0.7354667
```

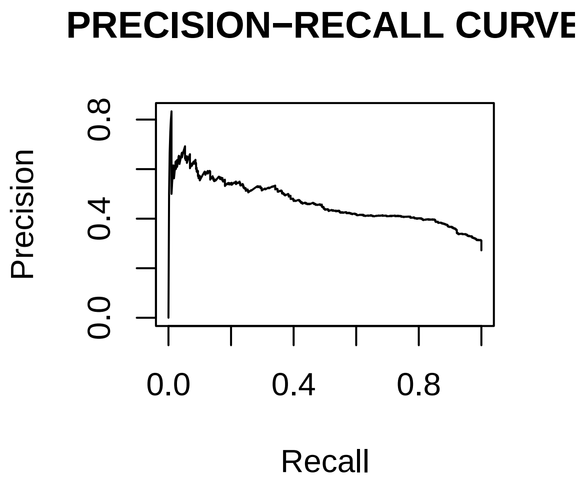
```
## Precision
## 0.7367847
```

```
##    Recall
## 0.9904762
```

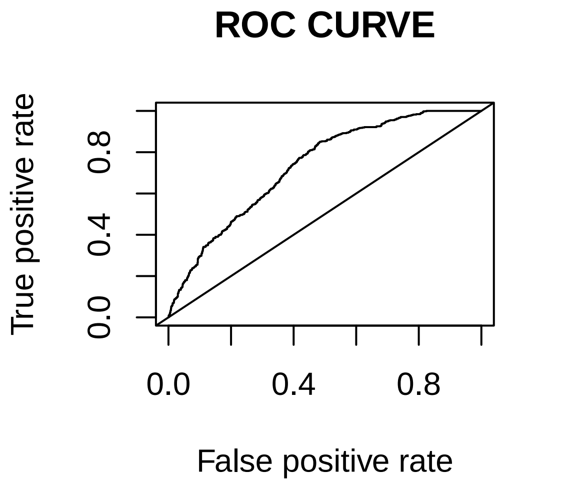
```
##    F1
## 0.845
```

The F1-Score of the model is 0.845. So, our Model is GOOD.

Now we check the PRECISION-RECALL curve.



Now, we check the ROC curve



The AUC is

```
## Setting levels: control = 0, case = 1
```

```
## Setting direction: controls < cases
```

```
## Area under the curve: 0.7302
```

So, both models selected by IV and STEPWISE method are almost similar, but F1 score is slight better for IV model. And in Credit default prediction purpose, most of the banking institute use IV model. So, we select finally the IV model.

Now, I check different MODEL DIAGONISTICS for validity of our model.

PHI COEFFICIENT

```
##
## Attaching package: 'psych'
## The following object is masked from 'package:car':
##
##     logit
```

```
## The following object is masked from 'package:scorecard':
##
##     describe
```

```
## The following object is masked from 'package:rcompanion':
##
##     phi
```

```
## The following objects are masked from 'package:ggplot2':
##
##     %+%, alpha
```

```
## [1] 0.13
```

GOODNESS OF FIT

Now we are going to test the following hypothesis:

H_0 : $\hat{Y}$ and Y are independent VS

H_1 : $\hat{Y}$ and Y are not independent

```
## X-squared
##      TRUE
```

CONTINGENCY COEFFICIENT

```
## X-squared
## 0.1284896
```

The observed value of P indicates that the association between Y and Yb is moderate

MACHINE LEARNING APPROACH FOR PREDICTING PD

Banks can also use Probability of Default (PD) for decisiong-making process in conjunction with scorecard points. Many empirical evidences has been providing the superiority of using Machine Learning Approach to predicting PD (Ben-David & Frank (2009), Van Gestel et al. (2005). In this section I will show R codes for training and selecting best Machine Learing Model based on cross validation approach.

I apply EXTREME GRADIENT BOOST(XGBOOST) Algorithm for improvement in Accuracy and F1 Score of my model.

Firstly I set a set of Hyperparameters for training the model.

```
## [1] 576
```

So total combinations of Hyperparameters are .

Next, we train our model using 5-fold Cross-validation.

```
##
## Attaching package: 'xgboost'
```

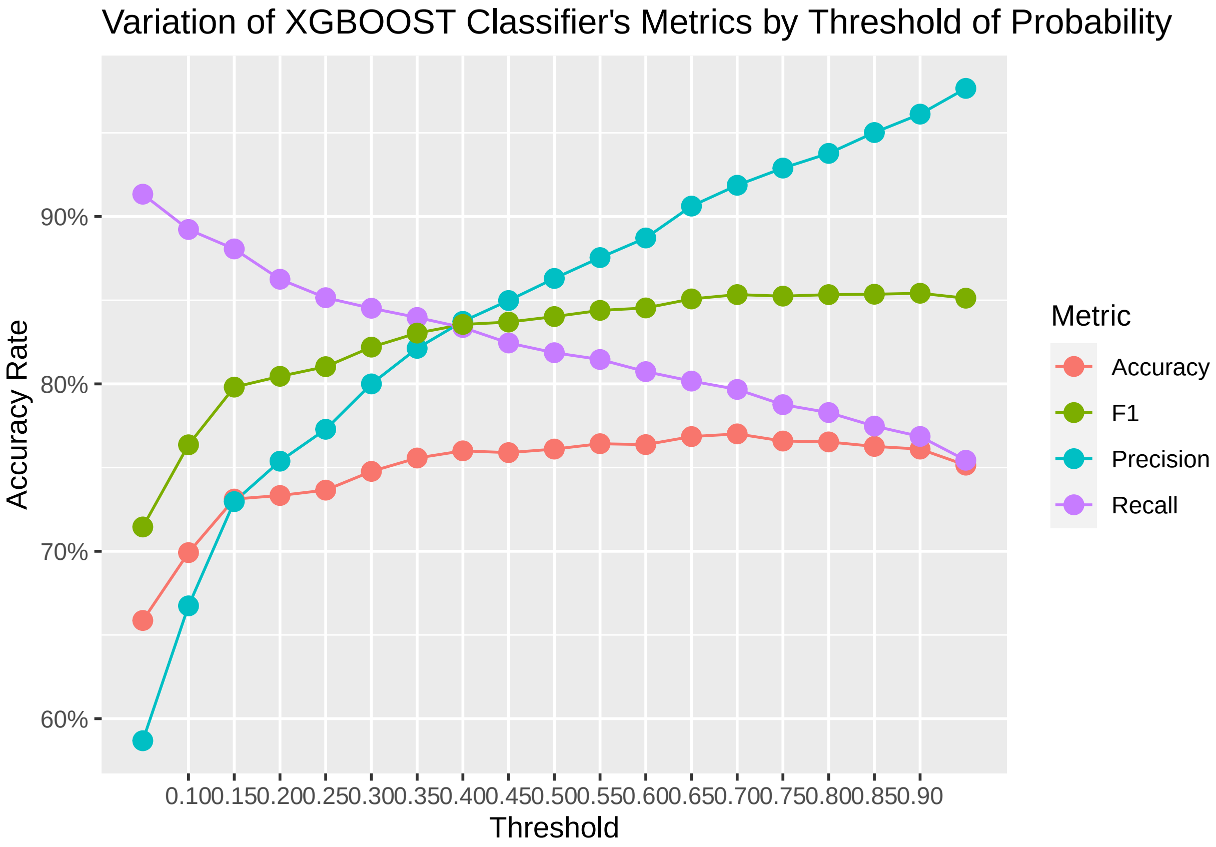
```
## The following object is masked from 'package:dplyr':
##
##     slice
##    eta max_depth min_child_weight subsample colsample_bytree optimal_trees
## 1 0.05         7                5       0.8              0.8           184
## 2 0.05         5                5       0.8              0.9           268
## 3 0.05         5                5       0.8              0.8           268
## 4 0.05         7                7       0.8              0.9           143
## 5 0.05         7                1       0.8              0.8           178
##   min_ERROR
## 1 0.4677014
## 2 0.4683918
## 3 0.4687450
## 4 0.4688201
## 5 0.4688205
```

Now, we use the combination of hyperparameters which causes least Logloss, that I obtained after model training using Cross-validation. And train the model.

```
## [23:41:35] WARNING: amalgamation/../src/learner.cc:627:
## Parameters: { "optimal_trees" } might not be used.
##
##   This could be a false alarm, with some parameters getting used by language bindings but
##   then being mistakenly passed down to XGBoost core, or some parameter actually being used
##   but getting flagged wrongly here. Please open an issue if you find any such cases.
```

```
## ##### xgb.Booster
## raw: 9.2 Mb
## call:
##   xgb.train(params = params, data = dtrain, nrounds = nrounds,
##     watchlist = watchlist, verbose = verbose, print_every_n = print_every_n,
##     early_stopping_rounds = early_stopping_rounds, maximize = maximize,
##     save_period = save_period, save_name = save_name, xgb_model = xgb_model,
##     callbacks = callbacks, objective = "binary:logistic")
## params (as set within xgb.train):
##   eta = "0.05", max_depth = "7", min_child_weight = "5", subsample = "0.8", colsample_bytree = "0.8"
## xgb.attributes:
##   niter
## callbacks:
##   cb.evaluation.log()
## # of features: 11
## niter: 5000
## nfeatures : 11
## evaluation_log:
##     iter train_logloss
##        1    0.67488967
##        2    0.65881670
## ---
##     4999    0.01946866
##     5000    0.01946536
```

```
##    user  system elapsed
##    7.00    0.11    1.34
```



CALCULATE PD (SEARCH OPTIMAL THRESOLD OF PROBABILITY THAT MAXIMIZE ACCURACY)

Accuracy criterion is widely used for evaluating model performance in context of credit scoring. However this measure is totally affected by threshold of probability that we select for classifier. I use simulation method for finding optimal threshold that maximizes accuracy. The graph shown below:

As, I deal with Imbalance data, the optimum thresold may not be 0.5. So, I will find the Optimum thresold from the grid search method.

Now, the we construct the confusion matrix,

```
##
## fraud_or_not_BOOST    0    1
##                  0 1254  320
##                  1  111  190
```

MISCLASSIFICATION Rate

```
##         0
## 0.2298667
```

```
##  Accuracy
## 0.7701333
```

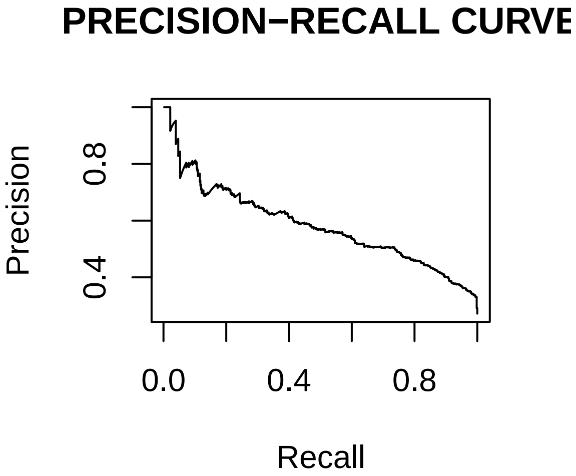
```
## Precision
## 0.7966963
```

```
##    Recall
## 0.9186813
```

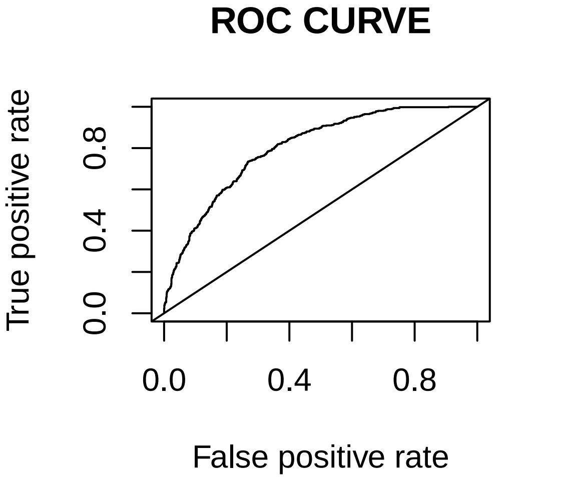
```
##        F1
## 0.8533515
```

The F1-Score of the model is 0.8563. And the ACCRACY IS 77.56% which is near about 4 % improvement from our Logistic model.

Now we check the PRECISION-RECALL curve.



Now, we check the ROC curve



The AUC is

```
## Setting levels: control = 0, case = 1
```

```
## Setting direction: controls < cases
```

```
## Area under the curve: 0.8005
```

##Probabilities of Default by Group for Test Data

Table 1: Probabilities of Default by Group for Test Data

Credit.Default	min	max	median	mean	n
Default	2e-04	0.9998	0.4839	0.4940	510
NonDefault	0e+00	0.9973	0.0241	0.1643	1365

So, we can say if PROBABILITY OF DEFAULT is above or equal 0.5, we can certainly say, Customer is a defaulter.

Some Criteria for Model Evaluation in Context of Credit Scoring

It is impossible to use a scoring model effectively without knowing how accurate it is. First, one needs to select the best model according to some criteria for evaluating model performance. The methodology of credit scoring models and some measures of their quality have been discussed in surveys conducted by Hand and Henley (1997), Thomas (2000), and Crook at al. (2007). However, until just ten years ago, the general literature devoted to the issue of credit scoring was not substantial. Fortunately, the situation has improved in the last decade with the publication of works by Anderson (2007), Crook et al. (2007), Siddiqi (2006), Thomas et al. (2002), and Thomas (2009), all of which address the topic of credit scoring. The most used criteria in context of credit scoring are:

Gain or lift is a measure of the effectiveness of a classification model calculated as the ratio between the results obtained with and without the model. Gain and lift charts are visual aids for evaluating performance of classification models. However, in contrast to the confusion matrix that evaluates models on the whole population gain or lift chart evaluates model performance in a portion of the population.

Scorecards based on our model provide scores. A score is a measure that allows lenders to rank customers from high risk (low score) to low risk (high score) and as such provides a relative measure of credit risk. Scores are unlimited and can be measured within any range; they can even be negative. A score is not the same as a probability. A probability also allows us to rank, but on top of that, since it is limited between 0 and 1, it also gives an absolute interpretation of credit risk. Hence, probabilities provide more information than scores do. For application scoring, one does not need well-calibrated probabilities of default. However, for other application areas such as regulatory capital calculation in a Basel setting, as we will discuss later, calibrated default probabilities are needed (Van Gestel and Baesens 2009).

```
## [INFO] creating woe binning ...
```

```
## [INFO] creating woe binning ...
## [INFO] Binning on 1875 rows and 12 columns in 00:00:15
```

```
## [INFO] converting into woe values ...
```

```
## [INFO] converting into woe values ...
```

Predictor	Group	Scorecard
basepoints	NA	458
Home.Ownership	Rent	-31
Home.Ownership	Own Home	-16
Home.Ownership	Home Mortgage% to %Have Mortgage	33
Months.since.last.delinquent	From -Inf to 30	-4
Months.since.last.delinquent	From 30 to 32	73
Months.since.last.delinquent	From 32 to 36	2
Months.since.last.delinquent	From 36 to 64	-16
Months.since.last.delinquent	From 64 to Inf	13
Years.in.current.job	10+ years% to %< 1 year% to %1 year	-13

Predictor	Group	Scorecard
Years.in.current.job	2 years% to %6 years% to %7 years% to %4 years	-1
Years.in.current.job	3 years% to %5 years% to %9 years	11
Years.in.current.job	8 years	68
Annual_Income	From -Inf to 1000000	-28
Annual_Income	From 1000000 to 1150000	-84
Annual_Income	From 1150000 to 1900000	28
Annual_Income	From 1900000 to Inf	72
Monthly.Debt	From -Inf to 4000	24
Monthly.Debt	From 4000 to 8000	-4
Monthly.Debt	From 8000 to 11000	14
Monthly.Debt	From 11000 to 13000	-16
Monthly.Debt	From 13000 to 21000	1
Monthly.Debt	From 21000 to Inf	-4
Current.Credit.Balance	From -Inf to 60000	-7
Current.Credit.Balance	From 60000 to 280000	2
Current.Credit.Balance	From 280000 to 320000	-28
Current.Credit.Balance	From 320000 to 380000	-4
Current.Credit.Balance	From 380000 to 440000	26
Current.Credit.Balance	From 440000 to 740000	-3
Current.Credit.Balance	From 740000 to Inf	23
Current.Loan.Amount	From -Inf to 460000	-9
Current.Loan.Amount	From 460000 to 780000	-41
Current.Loan.Amount	From 780000 to Inf	237
Term	Long Term	-59
Term	Short Term	24
Maximum.Open.Credit	From -Inf to 200000	-32
Maximum.Open.Credit	From 200000 to 850000	-4
Maximum.Open.Credit	From 850000 to 1050000	45
Maximum.Open.Credit	From 1050000 to 1700000	14
Maximum.Open.Credit	From 1700000 to Inf	65
Years.of.Credit.History	From -Inf to 10	-33
Years.of.Credit.History	From 10 to 19	-2
Years.of.Credit.History	From 19 to 28	17
Years.of.Credit.History	From 28 to 32	-16
Years.of.Credit.History	From 32 to Inf	5
Number.of.Open.Accounts	From -Inf to 6	-3
Number.of.Open.Accounts	From 6 to 8	7
Number.of.Open.Accounts	From 8 to 9	20
Number.of.Open.Accounts	From 9 to 11	-3
Number.of.Open.Accounts	From 11 to 12	15
Number.of.Open.Accounts	From 12 to 18	-4
Number.of.Open.Accounts	From 18 to Inf	-19

SCORECARD POINTS BY GROUP FOR TEST DATA (SELECTION BASED ON IV)

Table 3: Scorecad Points by Group for Test Data

Credit.Default	min	max	median	mean	n
Default	172	904	410	418	510
NonDefault	188	1024	482	515	1365

This is a basic scorecard. This needs a great improvement.

KEY CHARACTERISTICS OF A USEFUL SCORECARD MODEL

Before bringing a scorecard into production, it needs to be thoroughly evaluated. Depending on the exact setting and usage of the model, different aspects may need to be assessed during evaluation in order to ensure the model is acceptable for implementation. Key characteristics of successful scorecard model are:

INTERPRETABILITY:

A scorecard needs to be interpretable. In other words, a deeper understanding of the detected default behavior is required, for instance to validate the scorecard before it can be used. This aspect involves a certain degree of subjectivism, since interpretability may depend on the credit expert's knowledge. The interpretability of a model depends on its format, which in turn is determined by the adopted analytical technique. Models that allow the user to understand the underlying reasons why the model signals a customer to be a defaulter are called white box models, whereas complex, incomprehensible, mathematical models are often referred to as black box models.

STATISTICAL ACCURACY

Refers to the detection power and the correctness of the scorecard in labeling customers as defaulters. Several statistical evaluation criteria exist and may be applied to evaluate this aspect, such as the hit rate, lift curves, area under the curve (AUC), and so on. Statistical accuracy may also refer to statistical significance, meaning that the patterns that have been found in the data have to be valid and not the consequence of noise. In other words, we need to make sure that the model generalizes well and is not overfitted to the historical data set.

ECONOMICAL COST

Developing and implementing a scorecard involves a significantcost to an organization. The total cost includes the costs togather, preprocess, and analyze the data, and the costs to putthe resulting scorecards into production. In addition, the softwarecosts as well as human and computing resources should betaken into account. Possibly also external (e.g., credit bureau)data has to be bought to enrich the available in-house data.Clearly it is important to perform a thorough cost-benefit analysisat the start of the credit scoring project, and to gain insight intothe constituent factors of the return on investment of building ascorecard system.

REGULATORY COMPLIANCES

A scorecard should be in line and compliant with all applicable regulations and legislation. In a credit scoring setting, the Basel Accords specify what information can or cannot be used and how the target (i.e., default) should be defined. Other regulations (e.g., with respect to privacy and/or discrimination) should also be respected.

SOME PRACTICAL ASPECTS OF SCORECARD MODEL USING BY BANK

The most important usage of application scores is to decide on loan approval. The scores can also be used for pricing purposes. Risk-based pricing (sometimes also referred to as risk-adjusted pricing) sets the price or other characteristics (e.g., loan term, collateral) of the loan based on the perceived risk as measured by the application score. A lower score will imply a higher interest rate and vice versa.

There are still many unresolved aspects of the credit rating (for example, selecting best model that maximizes profit or turning model parameter) but because my time resources is limited, these interesting issues will be presented in an upcoming post.

LIMITATIONS

Although credit scoring systems are being implemented and used by most banks nowadays, they do face a number of limitations. A first limitation concerns the data that is used to estimate credit scoring models. Since data is the major, and in most cases the only, ingredient to build these models, its quality and predictive ability is key to the models' success.

The quality of the data refers, for example, to the num- ber of missing values and outliers, and to the recency and representativity of the data. Data quality issues can be difficult to detect without specific domain knowledge, but have an important impact on the scorecard development and resulting risk measures. The availability of high-quality data is a very important prerequisite for building good credit scoring models. However, not only does the data need to be of high quality, but it should be predictive as well, in the sense that the captured characteristics are related to the customer's likelihood of defaulting.

In addition, before constructing a scorecard model, we need to thoroughly reflect on why a customer defaults and which characteristics could potentially be related to this. Customers may default because of unknown reasons or information not available to the financial institution, thereby posing another limitation to the performance of credit scoring models. The statistical techniques used in developing credit scoring models typically assume a data set of sufficient size containing enough defaults. This may not always be the case for specific types of portfolios where only limited data is available, or only a low number of defaults is observed. For these types of portfolios, one may have to rely on alternative risk assessment methods using, for example, expert judgment based on the five Cs, as discussed earlier.

REFERENCES

1. Introduction to Linear Regression Analysis by Douglas C Montgomery, Elizabeth A Peck, G. Geoffrey Vining
2. An Introduction to Statistical Learning: with Applications in R by Gareth James, Daniela Witten, Trevor Hastie, Robert Tibshirani
3. Siddiqi, N. (2012). Credit risk scorecards: developing and implementing intelligent credit scoring. John Wiley & Sons.
4. https://www.listendata.com/, Web blog