**PROJECT BUSINESS REPORT**

**FINANCIAL AND RISK ANALYTICS**

**PROJECT OBJECTIVE:**

This project on the Finance and Risk Analytics module has the objective of creating an **Indian credit default risk model** on the dataset provided, and evaluating the efficiency of the same using a validation dataset given. Since the objective deals with credit risk, we are asked to provide a good model that helps us identify the people or companies **that will default in future**, using the variables given for the process. This will be our importance in doing the analysis for the benefit of identifying defaulters in future

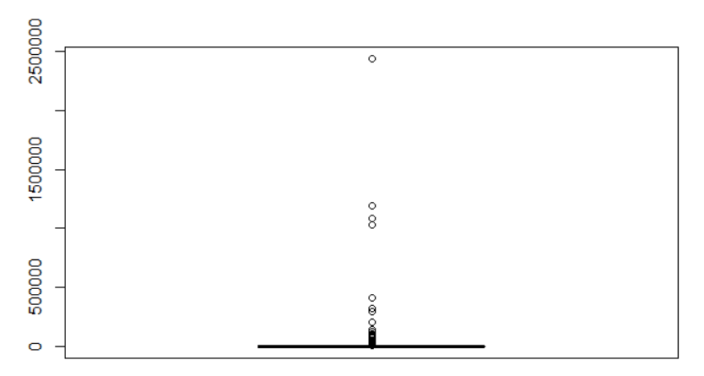
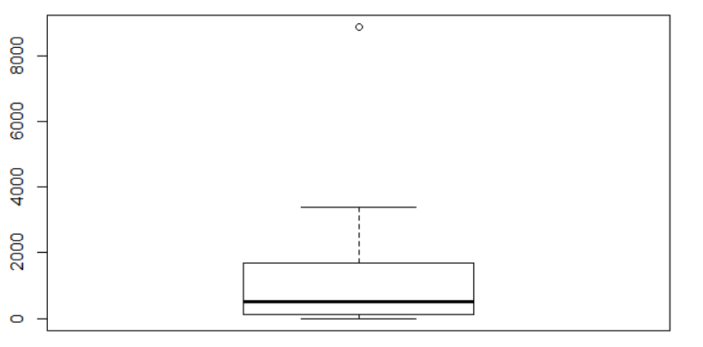
1. **EXPLORATORY DATA ANALYSIS (EDA)**
2. **Outlier Treatment:**

In this dataset, upon loading the raw data (model development dataset), we look at the summary of the data to find the lower, upper, averages, interquartile ranges, etc… to see if there are extreme values. The problem with extreme values is that **they tend to manipulate the overall look of the data.** It is an important step to either **remove or treat** outliers in order to build a fair model.

In our case, the dataset we used had a lot of outliers for almost all the variables. An example is below which shows a **box plot of total income**. Likewise we viewed the same for all the variables. Since the number of outliers was so much, we sought the **imputation** process. We used a **function** to define such that if the values are below and above the 25th and 75th percentile, we would replace them with a constant value in the 5th and 95th percentile range. Likewise we did for all the variables, which resulted in the below summaries. The below shows the before and after outlier treatment images of the summary and box plots . This helps in building a good model without having to worry about the extreme values.

Moreover, we also implied a function using the **car** library, called as outlier test. Here we can find the observations that have the most number of extreme values. We will have to build a regression model and run the model on the function **outlierTest**, which helps identify the rows with extreme values. We can then remove them to carry on with the replacement or imputation for the other outliers.

1. **Missing Value Treatment:**

Once upon treating the outliers, our next task is to either remove or treat the NA or null values present in the data. We found that the dataset provided had a lot of NA values, in each of the columns. We also had a variable called deposits which was totally filled with NAs. Hence we have to treat them and cannot remove them. Since the deposits variable did not provide any information, we have removed the column. We have also removed the numbers (1st column of the dataset) as it is unnecessary for the model formulation or any analysis.

Removing the NA value reduced the data to just 15% which was very less for analysis or model training. Hence removing them would cause too much data loss, thereby would be of no help in building a good model. Hence we have done imputation using the mean of the required variable, which replaces the NAs with the mean of the variable to fill the values. Later, we have checked the sum of NAs and obtained a 0 meaning that the NAs have been replaced accordingly.

We have not only done the missing value and outlier treatment for the raw data alone, but for the validation set as well.

1. **Multicollinearity Treatment:**

In the next step, we are going about building a corrplot. This helps us to find the correlation between the independent variables. While doing so, we remove the categorical and the dependent variables. In our case, we have no categorical variable but our dependent variable is the Net worth next year variable. A variable named Default is created using the dependent variable in a logical format (0,1) and taken as the dependent variable. So we will remove these variables from the corrplot to find the relation between the other independent variables.

Upon building the corrplot, we find that most of the variables are correlated with each other, like the total assets which are correlated with many variables. This is called as **multicollinearity.**

To remove multicollinearity, we have to build a regression model with the variables available. Using the models’ R2 value and with the formula (1/1-R2), we will calculate the **variance inflation factor (VIF)** which will help us in eliminating the highly correlated variables. If the VIF is anything more than 10, we have to remove that variable. We will then construct another model without that variable and check the VIF for that. Likewise the process is repeated until all the VIF values are within 10. The final variables that lead to a model with VIF value less than 10 proves that the variables in the data are not correlated and perfect for regression.

In our case, the first model with all variables had VIF values too high and distributed. Later upon eliminating non significant variables, we had 7 variables which provided a significant regression model. The purpose is that regression models cannot handle multicollinearity and may provide misleading results. Hence this step is essential to build robust models.

1. **Univariate Data Analysis:**

The first thing we see about a data is the structure and summary of the dataset. Likewise in our data we have summarized some points based on our earlier findings and the treatments done.

* Firstly, as we have removed the many outliers we found in each of the variables, we are now having the data within good limits.
* Secondly, we are free from the trouble of having NA values. We have treated the same by imputation with the mean values.
* Lastly, we have found the presence of multicollinearity which has been treated by VIF method. We have used the logistic model after treating with VIF as our final model for analysis.

**Basic Data Analysis:**

* Firstly, our raw dataset has **3541 rows and 52 columns.** This includes the 51 independent variables and 1 dependent variable (**Net worth next year)**
* We made us of the dimensions function to get the number of rows and columns.
* Using the names function, we viewed the column names to facilitate our study.
* We are now removing a couple of the variables: Number and Deposits. Number is just for identification and the deposits variable only contains NA values which provide no scope for imputation.
* Next was the **structure** of the data. All the variables, including the target variable, were **numeric in nature** except for a few of them which needed change.
* Following that, we were asked to create a dependent variable **Default** basis the net worth next year variable,which is of **binary** nature.

Hence we passed a command and converted this dependent variable to a logical type (factor) by assigning all **negative net worth next year values as 1 and the others to 0**. This means that the negative net worth next year companies are classified as **potential defaulting companies.**

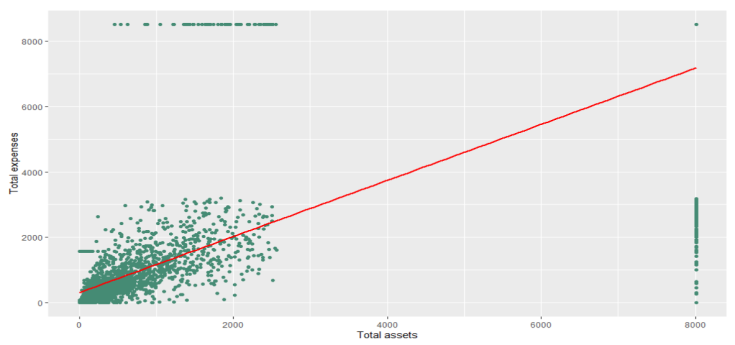
We can say that one of the main factors for predicting a company’s success will be their incomes and profits. We have a sum of all the companies’ profits, incomes, expenses, etc.. which are all the values that have been calculated after the outlier and imputation treatment.

One of the things we observe is that we find some profitability variables clearly having a negative minimum value. For example, the reserves and funds are at a negative 389.5 which tells that a company of this nature having either no reserves or loss is likely on the defaulter side.

1. **Bivariate Data Analysis:**

We are now moving onto the variable relationships or bi-variate analysis.

* **Correlation check**: We did the correlation plot to find variable relationships and upon doing that we found a sheer amount of **multi-collinearity.** For example, the total assets variable was explained by a lot of other variables such as total income, net worth, total liabilities, capital employed, total expenses, etc… By this we confirmed the presence of multi collinearity and treated the same.
* **Point Graphs:** Since all of our variables are numbers, we used a simple **point graph** to explain the relationship between those variables.
* First one between **Total Income and Net worth** on the y and x axis accordingly. We found a **positive correlation** between the two variables. The **lm abline** gave a better picture and showed a direct relationship. They looked clustery until 1000 and then a few values were missed. But the values at the extremes related to the outliers that were trimmed to the lower or higher ends of the chart.
* Second point graph between **Total expenses and Total Assets** gave the same picture as well. Also the next point-abline graph on the **Creditors and Debtors turnover** show the same. Though it is positive, we do see that the Creditors turnover is more than the Debtors turnover. This may make the company a little lagged behind when compared to its creditors.
* The final chart between **Finished goods and Net working capital** seemed to show a declining lm line. The finished goods turnover did not seem to be that good with increasing net working capital. This may yield a loss to the company and eventually make it to the default list.
* **Box plots:** The next two plots are box plots on one, **Debt equity ratio (DER)** and two, **Current ratio (CR)** with relation to **Default probabilities.** The **first** plot is self explanatory, which shows that a higher DER of more than 1 (in our case it reaches to about 6.5) is a red flag as it points out directly to defaulting. Whereas when lower than a median 1, the chances are ruled out. The **second** plot on the current ratio also tells the same story. A higher value of median 1 or more (**meaning assets are more than liabilities**) can save a company from defaulting than otherwise.



**Point graph:**

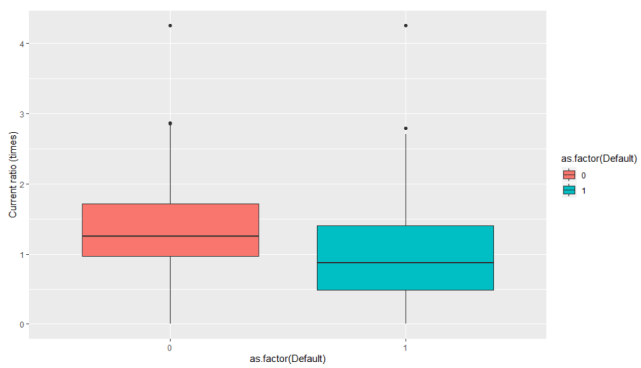
Total expenses vs Total assets

**Box plot:**

Current ratio with default

**Density plot:**

Distribution of PAT as a % of N.W

* **Density plot:** Our last plot is the density plot of **PAT as a % of net worth.** This plot shows if PAT% of greater than or equal to 0 can help a company survive default. Whereas anything less than 0 raises defaulting chances. Though this is the overall image, we also find some companies that have defaulted or survived in opposite conditions. This may suggest that apart from PAT%, there are some other important factors that determine a company’s default status.

1. **New Variable Creation:**

We have used the variables in the data for the creation of the below ratios.

* **Return on Assets (ROA) for Profitability:**

The return on assets is calculated by dividing **Profit after tax by Total assets**. This measures the earning per rupee of asset invested in the company. Better the value, better the company status.

* **Equity ratio for Leverage:**

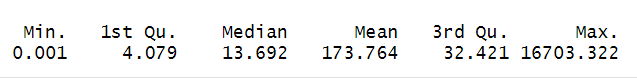
The equity ratio is calculated by dividing the Shareholders funds by the Capital employed. This measure gives the total owner contribution in the company. A higher rate indicates a company's long term solvency position, if otherwise the company is at a higher risk of defaulting.

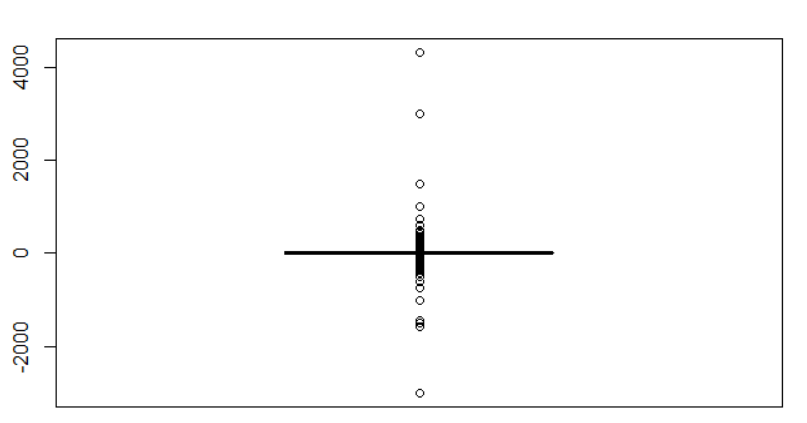
* **Contingent liabilities/Net working capital for Liquidity:**

As named, this ratio is calculated by dividing Contingent liabilities by Net working capital. This ratio helps to identify if the company is able to meet their emergency liabilities. Higher the value, higher the chances of default.

* **Sales/Total capital for Company size:**

The ratio as the name suggests, is the division of Sales by Total capital. This helps us to assign the position of the company if they are well to do or not. Higher the value, better is the position of the company.





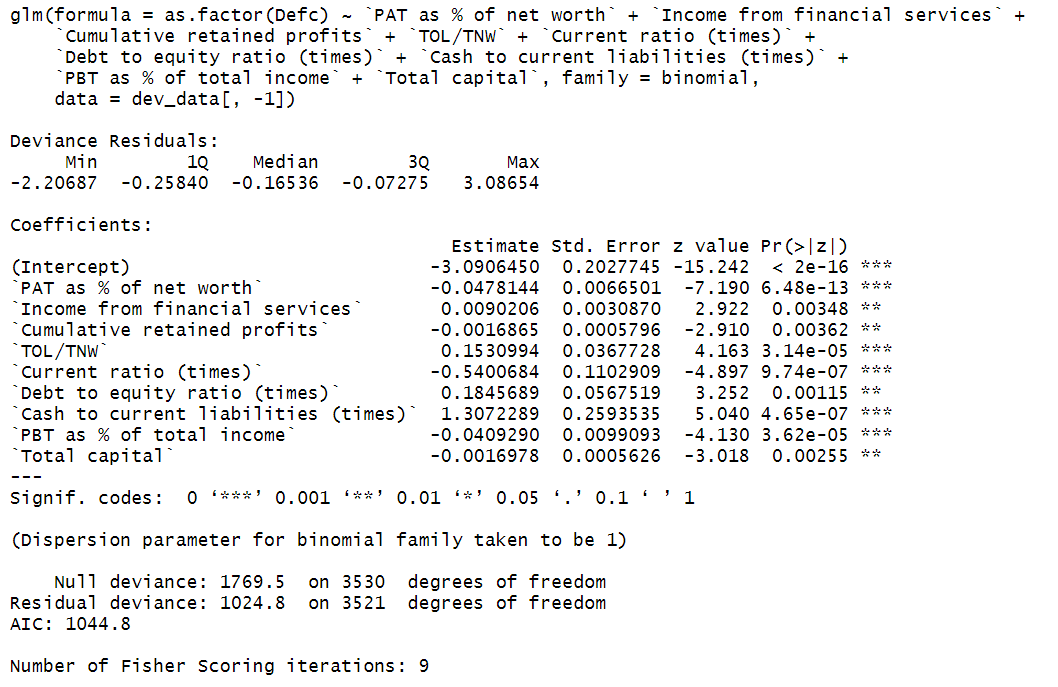
1. **MODELING:**

Our next important step is model building, which involves developing of multiple logistic regression models with important variables.

1. **Building Logistic Regression Models:**

We have built 4 models with PAT as % of net worth, Debt to equity ratio, TOL/TNW and Sales on each of the 4 factors of Profitability, Leverage, Liquidity and Company size and two more models combining them. Since, the model on all 4 made one variable insignificant, hence we removed that variable in our final model.

We have made use of the below model to validate our findings on the test data. Our model 1 was on all the independent variables. It had too many insignificant variables which were removed for the next model 2, which seemed to be only with the significant variables. The summary is as follows.

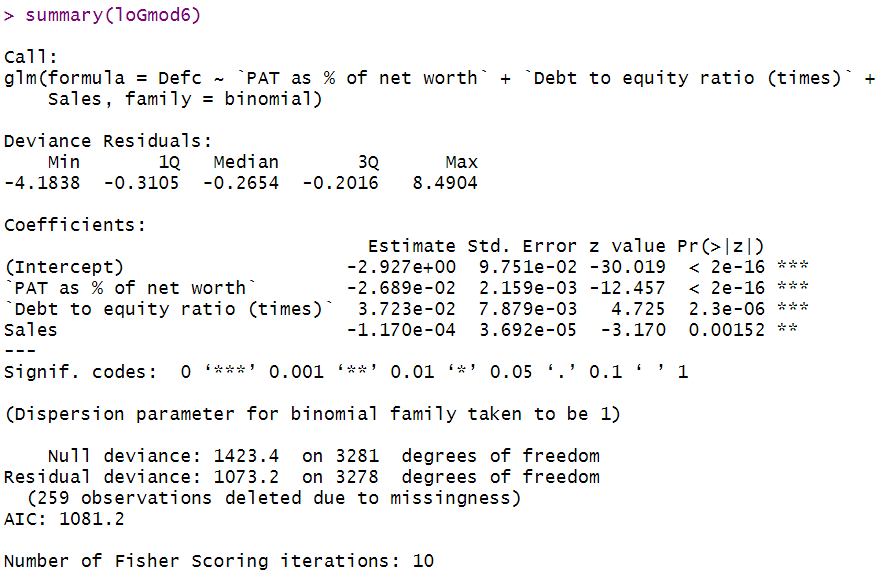


1. **Coefficients and sign analysis:**

All of the 4 models had similarities in coefficients, P Values, standard errors and standard deviation values.

* **P** **values:** The first 3 models gave very low values making them highly significant models. And the last model gave a significant p value of 0.0038. This proves that the variables play an important role in determining the defaulters.
* **Coefficients:** The coefficients of models 1 and 4 are negative, whilst those of models 2 and 3 are positive. This explains a higher value of debt equity ratio has increased default chances.
* **Standard error**: The standard errors of all the models are less than 0.008 which is pretty good. That of the fourth model on Sales has given a very low standard error of 4.555e-05.

The last model with PAT%, Debt equity and Sales combined gives more significant coefficients. The standard error values have also gone down more. The p value of Sales has further become better (0.0015) than earlier which was about 0.003. This states that the variables are very much significant together.



The summary of model 2, which is going to be fed to the validation data, has also given some specific values. The Akaike information criterion (AIC) is very low for this model which is something that makes the model significant.

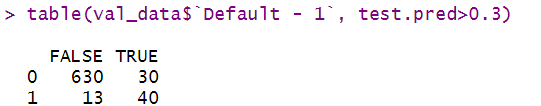
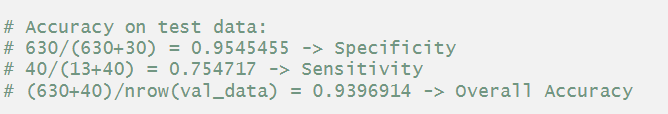
1. **MODEL PERFORMANCE MEASURES:**

As discussed above, we have used the model2 to test the validation dataset.

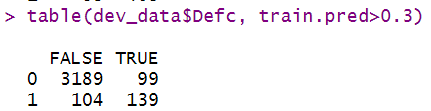
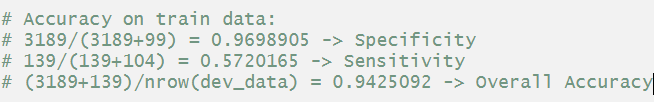
1. **Accuracy predictions on train and validation set:**

We make use of the predict function to do the prediction on the validation dataset with the type as **response** usually used for binomial results. We store this function under ***test.pred*** . Later we make a table, usually referred to as a confusion matrix, with the original default values in the validation dataset and the newly predicted values. The threshold is set to probability of 0.3 which tells that any value above probability of 0.3 is considered default. The threshold can be tweaked per convenience. A higher threshold may make it hard for the algorithm to find the defaulters.

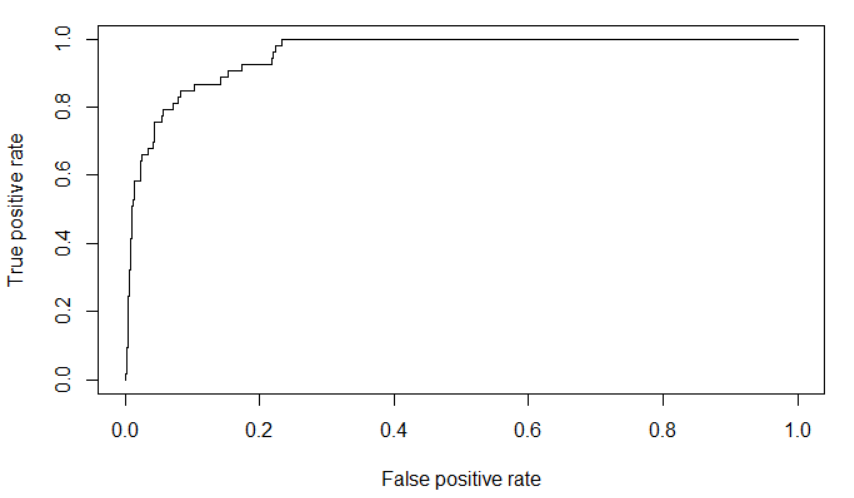
* The model is first tested on the validation dataset, which has been treated for outliers and NA values. The result is as follows; we see it has predicted almost 630 non defaulters correctly and 40 defaulters correctly giving an overall accuracy of almost 94% on the validation data. The picture to the right gives the **specificity, sensitivity and overall accuracy** of the model on the test data.

* Then the model is tested on the train data itself (the model development data). The model predicted 3189 non defaulters and 139 defaulters correctly with overall accuracy of 94% similar to the test data. Though the overall accuracy is good and the same for both, the prediction on the number of defaulters on the train set is lower of about only 57% whereas it was good on the test data (75% correct prediction). With the same threshold of > 0.3 for both, the accuracies on the train data are as follows:

We also plotted an ROCR performance curve with the rocr predictions as given below. The area under the curve (AUC) model has also confirmed that 95.6% of the times, the model has correctly predicted the values.

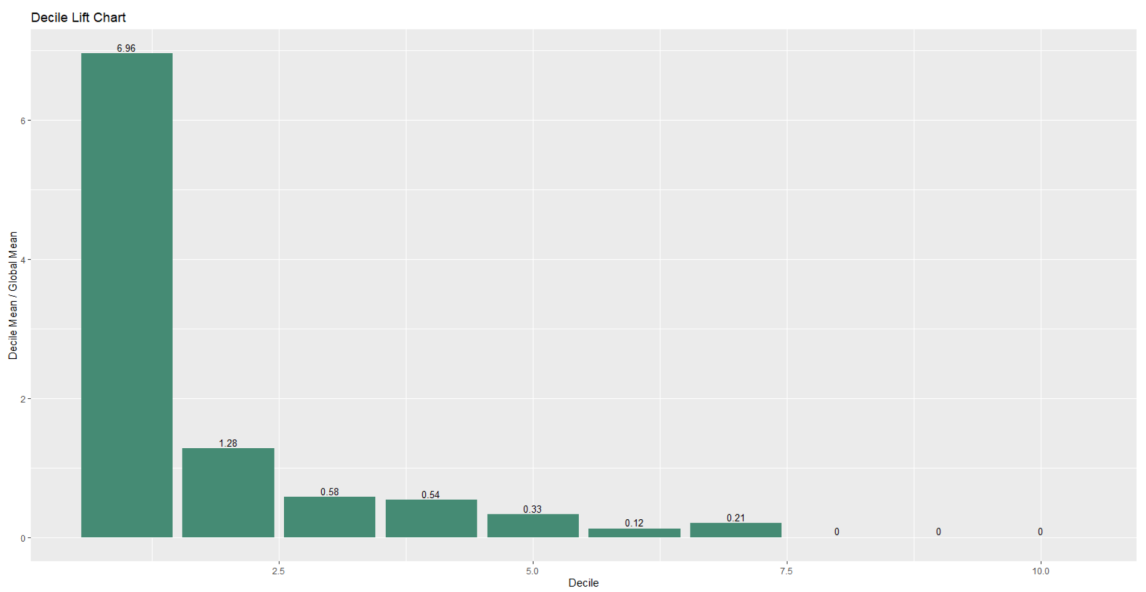




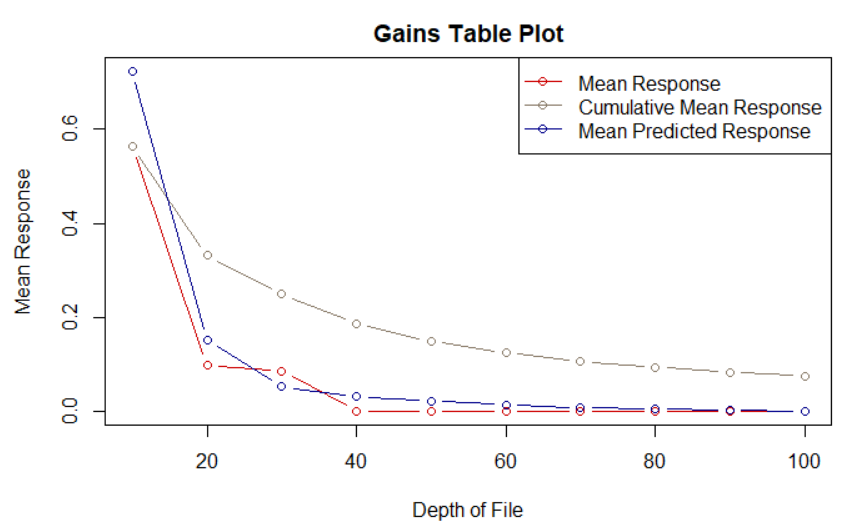
1. **Model performance basis Decile Analysis:**

Upon obtaining the accuracies, we figure out that the model has performed well on the validation dataset. But how well do we think the model has performed is to be questioned. To test that, we are using decile analysis. This is created to test the model’s ability to predict the desired outcome. The actual responses is compared to our predicted responses on a graph. The higher the bar, the better the prediction.

The first step in decile analysis would be to sort the data by predicted values in **descending order.** Then we need to divide into **10 deciles** or equal sized bins. The first bar or decile shows the value of companies most likely to default and the last decile shows the proportion of companies that are least likely to default. Our decile chart looks like this. From the graph, we see that the last 30% of the deciles are 0, meaning in the 70% of the data, we would have already predicted all the defaulters.



When we are looking at this, we want to see a staircase effect, which means that the values are indeed arranged in descending order. This ensures that the model is good to go with. If the deciles seem to not be in correct formation or portrays a flat image, it is a sign that the model is either under-performing or not good at all. Our graph looks fairly good and shows good results.



From the above gain table plot, we see that there is a correlation between mean responses and mean predicted responses. Hence we may conclude that this model is efficient enough to predict the defaulters.