FINANCIAL RISK ANALYSIS

Milestone 1

NOVEMBER, 2023

DIVYA VIKRAM

Table of Contents

List of Figures

List of Tables

Problem1:

We will check the descriptive statistics and do the null value and duplicate value check. Let's see the top 5 and bottom 5 records of the dataset

	Co_Code	Co_Name	_Operating_Expense_Rate	_Research_and_development_expense_rate	_Cash_flow_rate	_Interest_bearing_debt_interest_rate	_Tax_rate_
0	16974	Hind.Cables	8.820000e+09	0.000000e+00	0.462045	0.000352	0.00141
1	21214	Tata Tele. Mah.	9.380000e+09	4.230000e+09	0.460116	0.000716	0.00000
2	14852	ABG Shipyard	3.800000e+09	8.150000e+08	0.449893	0.000496	0.00000
3	2439	GTL	6.440000e+09	0.000000e+00	0.462731	0.000592	0.00931
4	23505	Bharati Defence	3.680000e+09	0.000000e+00	0.463117	0.000782	0.40024

5 rows × 58 columns

Figure 1:Top 5 records of the dataset

	Co_Code	Co_Name	_Operating_Expense_Rate	_Research_and_development_expense_rate	_Cash_flow_rate	_Interest_bearing_debt_interest_rate	_Tax_r
2053	2743	Kothari Ferment.	3.021580e-04	6.490000e+09	0.477066	0.000000	0.1
2054	21216	Firstobj.Tech.	1.371450e-04	0.000000e+00	0.465211	0.000658	0.0
2055	142	Diamines & Chem.	2.114990e-04	8.370000e+09	0.480248	0.000502	0.0
2056	18014	IL&FS Engg.	3.750000e+09	0.000000e+00	0.474670	0.000578	0.3
2057	43229	Channel Nine	2.981110e-04	0.000000e+00	0.467203	0.000826	0.0

5 rows × 58 columns

Figure 2: Bottom 5 records of the dataset

The dataset contains 2058 rows and 58 columns.

Explorative Analysis of the Dataset:

Table below gives us information of data like number of record entries, number of columns, column data types, number of non-null values in each column.

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2058 entries, 0 to 2057
Data columns (total 58 columns):
 #   Column                                                  Non-Null Count  Dtype
---  ------                                                  --------------  -----
 0   Co_Code                                                 2058 non-null   int64
 1   Co_Name                                                 2058 non-null   object
 2   _Operating_Expense_Rate                                 2058 non-null   float64
 3   _Research_and_development_expense_rate                  2058 non-null   float64
 4   _Cash_flow_rate                                         2058 non-null   float64
 5   _Interest_bearing_debt_interest_rate                    2058 non-null   float64
 6   _Tax_rate_A                                             2058 non-null   float64
 7   _Cash_Flow_Per_Share                                    1891 non-null   float64
 8   _Per_Share_Net_profit_before_tax_Yuan_                  2058 non-null   float64
 9   _Realized_Sales_Gross_Profit_Growth_Rate                2058 non-null   float64
 10  _Operating_Profit_Growth_Rate                           2058 non-null   float64
 11  _Continuous_Net_Profit_Growth_Rate                      2058 non-null   float64
 12  _Total_Asset_Growth_Rate                                2058 non-null   float64
 13  _Net_Value_Growth_Rate                                  2058 non-null   float64
 14  _Total_Asset_Return_Growth_Rate_Ratio                   2058 non-null   float64
 15  _Cash_Reinvestment_perc                                 2058 non-null   float64
 16  _Current_Ratio                                          2058 non-null   float64
 17  _Quick_Ratio                                            2058 non-null   float64
 18  _Interest_Expense_Ratio                                 2058 non-null   float64
 19  _Total_debt_to_Total_net_worth                          2037 non-null   float64
 20  _Long_term_fund_suitability_ratio_A                     2058 non-null   float64
 21  _Net_profit_before_tax_to_Paid_in_capital               2058 non-null   float64
 22  _Total_Asset_Turnover                                   2058 non-null   float64
 23  _Accounts_Receivable_Turnover                           2058 non-null   float64
 24  _Average_Collection_Days                                2058 non-null   float64
 25  _Inventory_Turnover_Rate_times                          2058 non-null   float64
 26  _Fixed_Assets_Turnover_Frequency                        2058 non-null   float64
 27  _Net_Worth_Turnover_Rate_times                          2058 non-null   float64
 28  _Operating_profit_per_person                            2058 non-null   float64
 29  _Allocation_rate_per_person                             2058 non-null   float64
 30  _Quick_Assets_to_Total_Assets                           2058 non-null   float64
 31  _Cash_to_Total_Assets                                   1962 non-null   float64
 32  _Quick_Assets_to_Current_Liability                      2058 non-null   float64
 33  _Cash_to_Current_Liability                              2058 non-null   float64
 34  _Operating_Funds_to_Liability                           2058 non-null   float64
 35  _Inventory_to_Working_Capital                           2058 non-null   float64
 36  _Inventory_to_Current_Liability                         2058 non-null   float64
 37  _Long_term_Liability_to_Current_Assets                  2058 non-null   float64
 38  _Retained_Earnings_to_Total_Assets                      2058 non-null   float64
 39  _Total_income_to_Total_expense                          2058 non-null   float64
 40  _Total_expense_to_Assets                                2058 non-null   float64
 41  _Current_Asset_Turnover_Rate                            2058 non-null   float64
 42  _Quick_Asset_Turnover_Rate                              2058 non-null   float64
 43  _Cash_Turnover_Rate                                     2058 non-null   float64
 44  _Fixed_Assets_to_Assets                                 2058 non-null   float64
 45  _Cash_Flow_to_Total_Assets                              2058 non-null   float64
 46  _Cash_Flow_to_Liability                                 2058 non-null   float64
 47  _CFO_to_Assets                                          2058 non-null   float64
 48  _Cash_Flow_to_Equity                                    2058 non-null   float64
 49  _Current_Liability_to_Current_Assets                    2044 non-null   float64
 50  _Liability_Assets_Flag                                  2058 non-null   int64
 51  _Total_assets_to_GNP_price                              2058 non-null   float64
 52  _No_credit_Interval                                     2058 non-null   float64
 53  _Degree_of_Financial_Leverage_DFL                       2058 non-null   float64
 54  _Interest_Coverage_Ratio_Interest_expense_to_EBIT       2058 non-null   float64
 55  _Net_Income_Flag                                        2058 non-null   int64
 56  _Equity_to_Liability                                    2058 non-null   float64
 57  Default                                                 2058 non-null   int64
dtypes: float64(53), int64(4), object(1)
memory usage: 932.7+ KB
```

Table 1: Data Information

There are 58 columns of which 53 are type float , 4 are type int64 and 1 object type variable. 'Default' is our class of interest. It tells which indicates whether the Company has defaulted on its payments or not. It's a categorical variable with 1 meaning Defaulted and 0 meaning Not Defaulted.

We will be dropping the variables Co Code and Co Name as these columns are not useful for our model building

Null value check

There are null values for some variables in the dataset.

Duplicate check

There no duplicate records in the dataset.

Data Summary

Let's see the 5 number data summary of the numerical variables where the mean, standard deviation, minimum, maximum,25%,50%,75% of each numerical column.

	count	mean	std	min	25%	50%	75%	max
Co_Code	2058.0	1.757211e+04	2.189289e+04	4.000000	3.674000e+03	6.240000e+03	2.428075e+04	7.249300e+04
_Operating_Expense_Rate	2058.0	2.052389e+09	3.252624e+09	0.000100	1.578727e-04	3.330330e-04	4.110000e+09	9.980000e+09
_Research_and_development_expense_rate	2058.0	1.208634e+09	2.144568e+09	0.000000	0.000000e+00	1.994130e-04	1.550000e+09	9.980000e+09
_Cash_flow_rate	2058.0	4.652426e-01	2.266269e-02	0.000000	4.600991e-01	4.634450e-01	4.680691e-01	1.000000e+00
_Interest_bearing_debt_interest_rate	2058.0	1.113022e+07	9.042595e+07	0.000000	2.760280e-04	4.540450e-04	6.630660e-04	9.900000e+08
_Tax_rate_A	2058.0	1.147770e-01	1.524457e-01	0.000000	0.000000e+00	3.709890e-02	2.161909e-01	9.996963e-01
_Cash_Flow_Per_Share	1891.0	3.199856e-01	1.529979e-02	0.169449	3.149890e-01	3.206479e-01	3.259178e-01	4.622268e-01
_Per_Share_Net_profit_before_tax_Yuan_	2058.0	1.769673e-01	3.015730e-02	0.000000	1.666039e-01	1.756421e-01	1.858854e-01	7.923477e-01
_Realized_Sales_Gross_Profit_Growth_Rate	2058.0	2.276117e-02	2.170104e-02	0.004282	2.205831e-02	2.210001e-02	2.215200e-02	1.000000e+00
_Operating_Profit_Growth_Rate	2058.0	8.481083e-01	4.589093e-03	0.736430	8.479740e-01	8.480386e-01	8.481147e-01	1.000000e+00
_Continuous_Net_Profit_Growth_Rate	2058.0	2.173915e-01	5.678779e-03	0.000000	2.175741e-01	2.175961e-01	2.176198e-01	2.332046e-01
_Total_Asset_Growth_Rate	2058.0	5.287663e+09	2.912615e+09	0.000000	4.315000e+09	6.225000e+09	7.220000e+09	9.980000e+09
_Net_Value_Growth_Rate	2058.0	5.189504e+06	2.077918e+08	0.000000	4.362833e-04	4.554170e-04	4.883758e-04	9.330000e+09
_Total_Asset_Return_Growth_Rate_Ratio	2058.0	2.641004e-01	2.415661e-03	0.251620	2.637383e-01	2.640161e-01	2.643097e-01	3.586288e-01
_Cash_Reinvestment_perc	2058.0	3.771970e-01	2.737311e-02	0.025828	3.707295e-01	3.789678e-01	3.855575e-01	1.000000e+00
_Current_Ratio	2058.0	1.336249e+06	6.061917e+07	0.000000	6.567062e-03	8.945370e-03	1.350542e-02	2.750000e+09
_Quick_Ratio	2058.0	2.775510e+07	4.448654e+08	0.000000	2.946399e-03	5.284241e-03	8.902983e-03	9.230000e+09
_Interest_Expense_Ratio	2058.0	6.312913e-01	6.785512e-03	0.525126	6.306116e-01	6.307999e-01	6.317437e-01	8.121652e-01
_Total_debt_to_Total_net_worth	2037.0	1.071429e+07	2.696960e+08	0.000000	3.924894e-03	7.270721e-03	1.306869e-02	9.940000e+09
_Long_term_fund_suitability_ratio_A	2058.0	8.973310e-03	3.485186e-02	0.004129	5.162031e-03	5.517000e-03	6.415301e-03	1.000000e+00
_Net_profit_before_tax_to_Paid_in_capital	2058.0	1.753994e-01	2.622348e-02	0.000000	1.658623e-01	1.745683e-01	1.844450e-01	7.921047e-01
_Total_Asset_Turnover	2058.0	1.286405e-01	1.006216e-01	0.000000	6.146927e-02	1.034483e-01	1.679160e-01	9.190405e-01
_Accounts_Receivable_Turnover	2058.0	4.159864e+07	5.047673e+08	0.000000	7.446260e-04	1.081432e-03	1.854463e-03	9.740000e+09
_Average_Collection_Days	2058.0	2.629786e+07	4.109967e+08	0.000000	3.576384e-03	6.001272e-03	8.638997e-03	8.800000e+09
_Inventory_Turnover_Rate_times	2058.0	2.030227e+09	3.077250e+09	0.000000	1.909297e-04	1.910000e+07	3.815000e+09	9.990000e+09
_Fixed_Assets_Turnover_Frequency	2058.0	1.230898e+09	2.649289e+09	0.000000	2.278950e-04	5.995245e-04	8.423224e-03	9.990000e+09
_Net_Worth_Turnover_Rate_times	2058.0	3.957710e-02	4.239591e-02	0.008871	2.048387e-02	2.870968e-02	4.435484e-02	1.000000e+00
_Operating_profit_per_person	2058.0	4.036693e-01	5.358970e-02	0.000000	3.913864e-01	3.950781e-01	4.008927e-01	1.000000e+00
_Allocation_rate_per_person	2058.0	5.725559e+06	1.979500e+08	0.000000	4.671612e-03	1.062969e-02	2.457491e-02	8.260000e+09

Table 2 Data summary of the numerical variables

Outlier Treatment

To detect the outliers, we can check the boxplot of the variables . Boxplots graphically represent the distribution of the data and provide a lot of statistical information, including — medians, ranges, and outliers.. When reviewing a box plot, an outlier is defined as a data point that is located outside the whiskers of the box plot. For example, outside 1.5 times the interquartile range above the upper quartile and below the lower quartile (Q1 – 1.5 * IQR or Q3 + 1.5 * IQR)

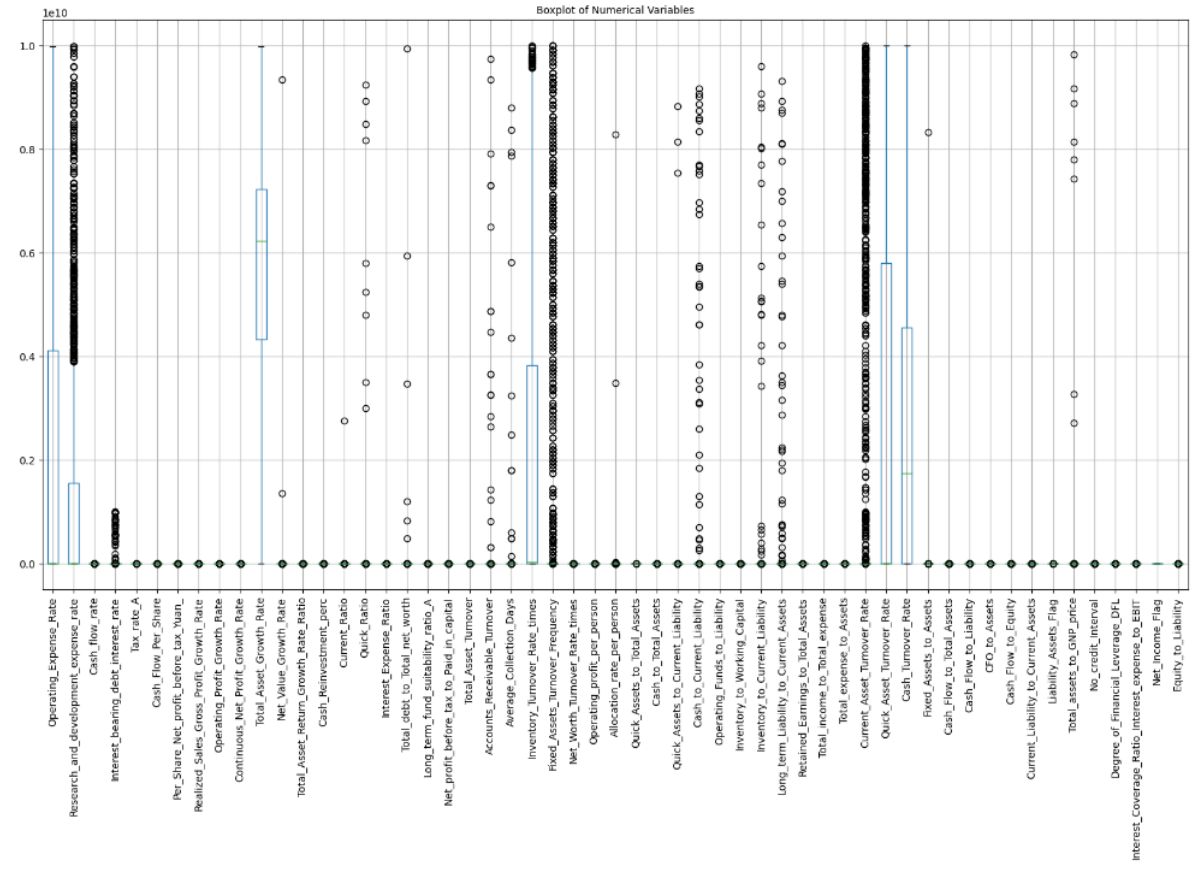


Figure 3: Boxplot of Numerical variables with outliers

As we can see from the boxplot of Numerical variables above ,most of the variables have outliers.

Extreme and heavy outliers can be seen in fields like: '_Research_and_development_expense_rate', '_Interest_bearing_debt_interest_rate', '_Fixed_Assets_Turnover_Frequency', '_Current_Asset_Turnover_Rate'.

Let's see the count of outliers in the dataset for all the columns.

_Operating_Expense_Rate	0
_Research_and_development_expense_rate	264
_Cash_flow_rate	206
_Interest_bearing_debt_interest_rate	94
_Tax_rate_A	42
_Cash_Flow_Per_Share	146
_Per_Share_Net_profit_before_tax_Yuan_	186
_Realized_Sales_Gross_Profit_Growth_Rate	283
_Operating_Profit_Growth_Rate	317
_Continuous_Net_Profit_Growth_Rate	340
_Total_Asset_Growth_Rate	0
_Net_Value_Growth_Rate	304
_Total_Asset_Return_Growth_Rate_Ratio	226
_Cash_Reinvestment_perc	220
_Current_Ratio	193
_Quick_Ratio	190
_Interest_Expense_Ratio	328
_Total_debt_to_Total_net_worth	105
_Long_term_fund_suitability_ratio_A	234
_Net_profit_before_tax_to_Paid_in_capital	173
_Total_Asset_Turnover	101
_Accounts_Receivable_Turnover	281
_Average_Collection_Days	77
_Inventory_Turnover_Rate_times	29
_Fixed_Assets_Turnover_Frequency	501
_Net_Worth_Turnover_Rate_times	165
_Operating_profit_per_person	357
_Allocation_rate_per_person	200
_Quick_Assets_to_Total_Assets	4
_Cash_to_Total_Assets	163
_Quick_Assets_to_Current_Liability	185
_Cash_to_Current_Liability	253
_Operating_Funds_to_Liability	219
_Inventory_to_Working_Capital	247
_Inventory_to_Current_Liability	129
_Long_term_Liability_to_Current_Assets	213
_Retained_Earnings_to_Total_Assets	208
_Total_income_to_Total_expense	136
_Total_expense_to_Assets	168
_Current_Asset_Turnover_Rate	464
_Quick_Asset_Turnover_Rate	0
_Cash_Turnover_Rate	0
_Fixed_Assets_to_Assets	10
_Cash_Flow_to_Total_Assets	317
_Cash_Flow_to_Liability	407
_CFO_to_Assets	110
_Cash_Flow_to_Equity	306
_Current_Liability_to_Current_Assets	121
_Liability_Assets_Flag	7
_Total_assets_to_GNP_price	235
_No_credit_Interval	396
_Degree_of_Financial_Leverage_DFL	438
_Interest_Coverage_Ratio_Interest_expense_to_EBIT	376
_Net_Income_Flag	0
_Equity_to_Liability	190
dtype: int64	

Table 3: Number of Outliers in the columns.

There are 9.6% outliers in the dataset .

One of the most important steps as part of data preprocessing is detecting and treating the outliers as they can negatively affect the statistical analysis and the training process of a machine learning algorithm resulting in lower accuracy.

Below are some of the methods of treating the outliers:

Option 1: Trimming/Remove the outliers

In this technique, we remove the outliers from the dataset. Although it is not a good practice to follow.

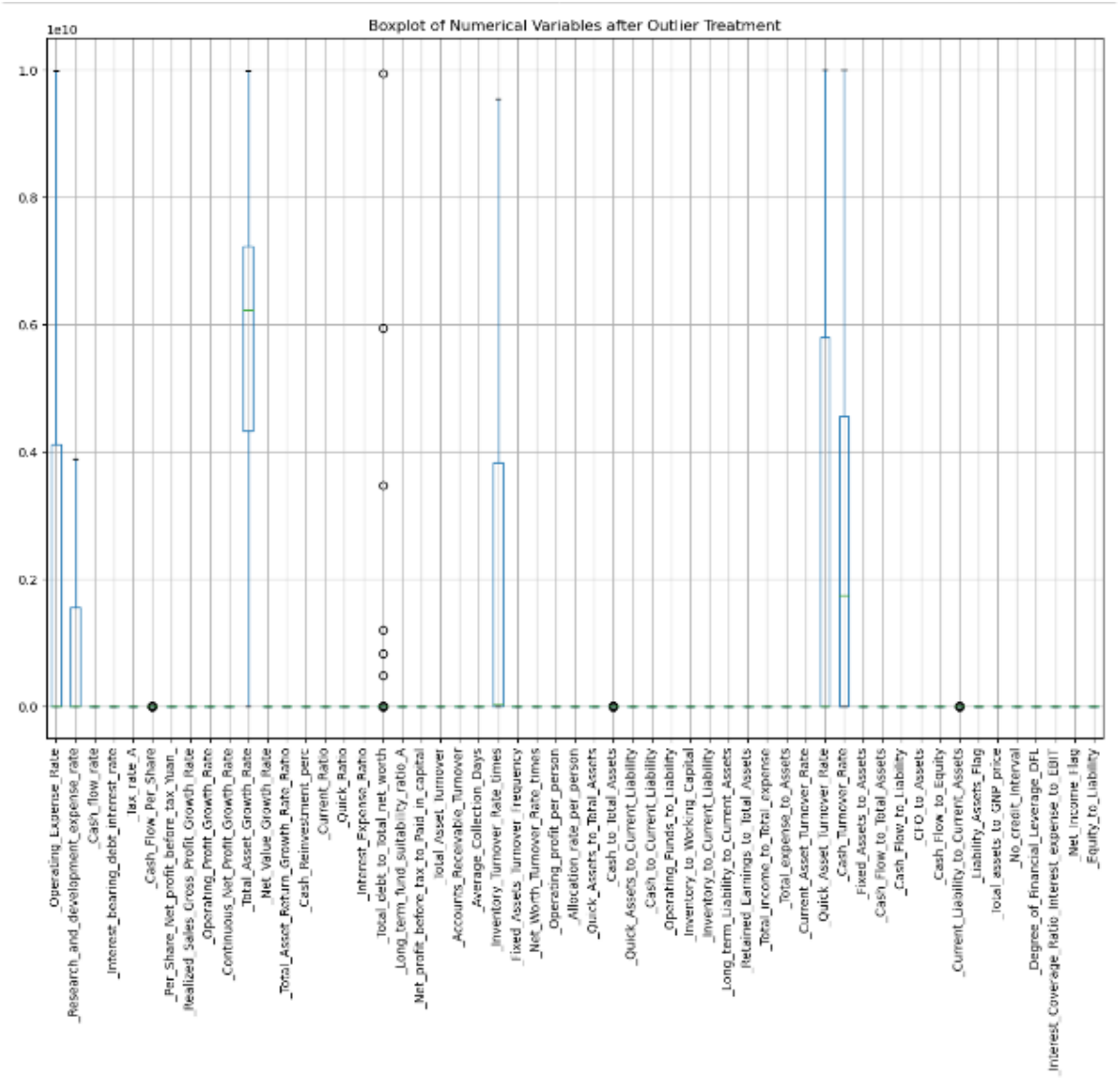
Option 2: Quantile Based Flooring and Capping

In this technique, the outlier is capped at a certain value above the 75th percentile + 1.5 * Inter Quartile Range(Upper Limit) or floored at a factor below the 25th percentile – 1.5 * IQR(Lower Limit).The data points that are lesser than the Lower Limit are replaced with the Lower Limit value and the data points that are greater than the Upper Limit are replaced with Upper Limit value.

Option 3: Mean/Median Imputation

Imputation techniques include replacing outliers with the mean, median, or another suitable value based on the characteristics of the data .As the mean value is highly influenced by the outliers, it is advised to replace the outliers with the median value.

We will use the **Quantile Based Flooring and Capping** method to remove the outliers in the dataset . Let's see the boxplot after removing the outliers in the variables .



After treating outliers, we observe that there are few more outliers left for variables 'Cash_Flow_Per_Share', 'Total_debt_to_Total_net_worth', 'Cash_to_Total_Assets' and 'Current_Liability_to_Current_Assets' that is because these variables have null values .

Missing Value Treatment

Incomplete data can bias the results of the machine learning models and/or reduce the accuracy of the model and hence need to treated correctly

There are 2 primary ways of handling missing values:

- Deleting the Missing values
- Imputing the Missing Values

Deleting the Missing value

It is one of the quick and dirty techniques one can use to deal with missing values. The disadvantage of this method is one might end up deleting some useful data from the dataset.

There are 2 ways one can delete the missing data values:

Deleting the entire row :

If a row has many missing values, you can drop the entire row. If every row has some (column) value missing, you might end up deleting the whole data.

Deleting the entire column**:**

If a certain column has many missing values, then you can choose to drop the entire column. The code to drop the entire column is as follows:

Imputing the Missing Value

There are many imputation methods for replacing the missing values. Some of the ways of replacing the missing values include :

- Replacing with an arbitrary value
- Replacing with the mean
- Replacing with the mode (in case of categorical variables)
- Replacing with the median(It's better to use the median value for imputation in the case of outliers)
- Nearest Neighbours Imputations (KNNImputer)

In KNNImputer missing values are imputed using the k-Nearest Neighbours approach, where a Euclidean distance is used to find the nearest neighbours' is a method which works on the basic approach of the KNN algorithm rather than the naive approach of filling all the values with mean or the median. The idea in KNN methods is to identify 'k' samples in the dataset that are similar or close in the space. Then we use these 'k' samples to estimate the value of the missing data points. Each sample's missing values are imputed using the mean value of the 'k'-neighbours found in the dataset.

For our dataset , we will be imputing the missing values using KNN Imputer. For our imputation we use number of neighbours as 5.

We will be imputing the Train and Test set separately . We, perform imputation on the training set first, and then apply it on the test set separately to lessen the risk of overfitting and data leakage problems which may result in developing models that can't be used in production.

There are null values in the variables 'Cash_Flow_Per_Share', 'Total_debt_to_Total_net_worth', 'Cash_to_Total_Assets' and 'Current_Liability_to_Current_Assets'.
There are 298 missing values in the dataset which make up 0.25% of the dataset

We can visually represent the missing values in the plot below

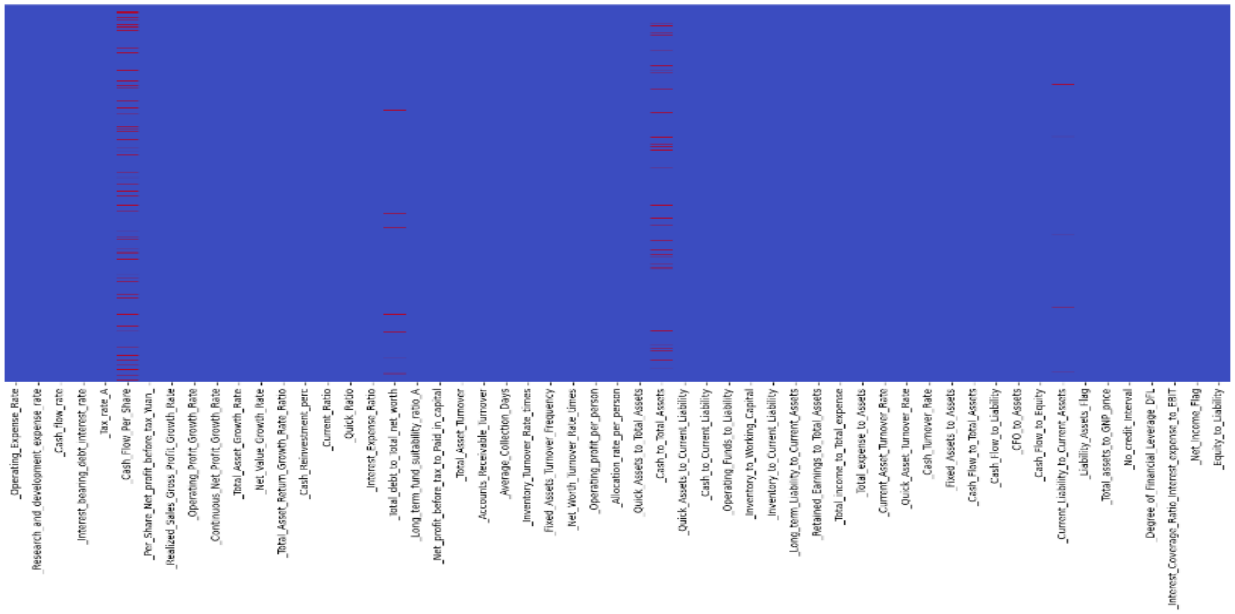


Figure 4:Visual representation of Missing Values in the dataset

As we can see from the plot above, the red lines indicate the missing data .
There are 167 null values in variable 'Cash_Flow_Per_Share', 21 null values in variable 'Total_debt_to_Total_net_worth', 96 in 'Cash_to_Total_Assets' and 14 in 'Current_Liability_to_Current_Assets'

After imputing the null/missing values using KNN Imputer, Lets visualize the train and test dataset

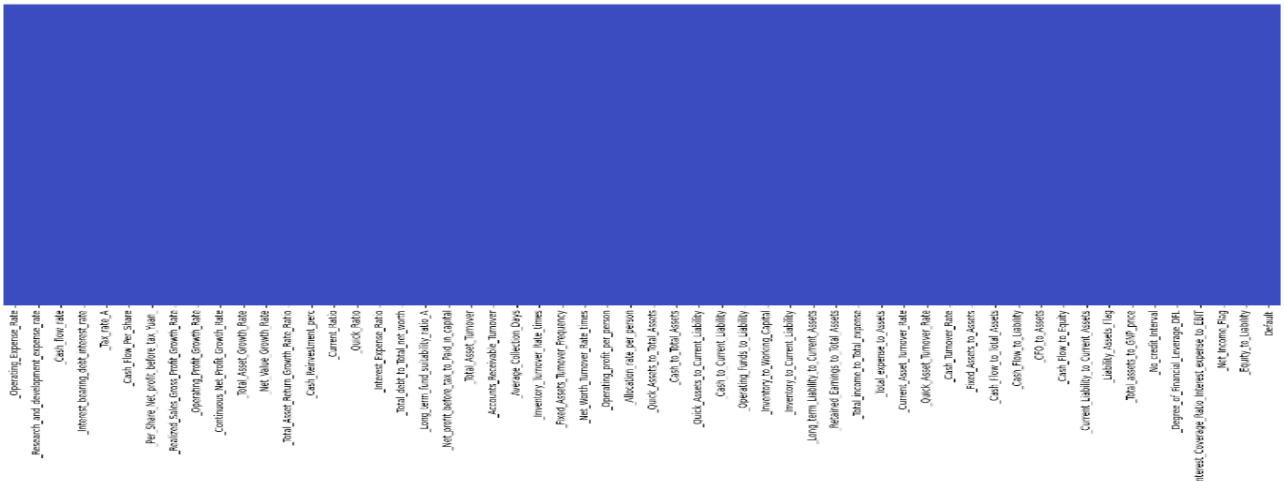


Figure 5: Training Dataset after Imputing Missing Values

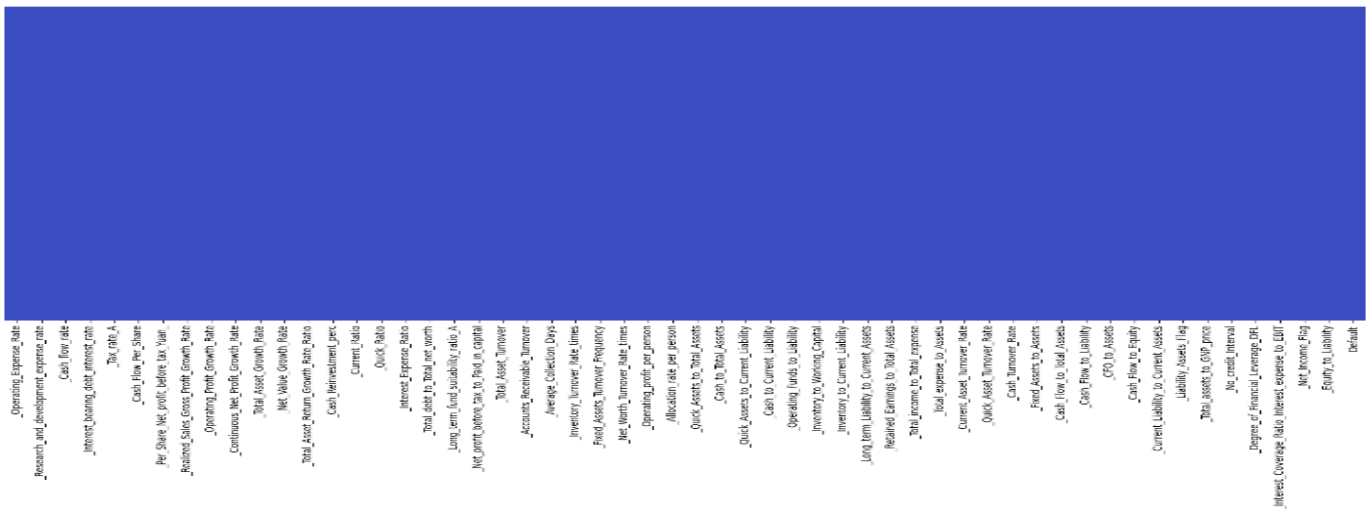


Figure 6: Testing Dataset after Imputing Missing Values

As we see that there are no null values in the training and testing dataset as there are no red lines as before

Univariate & Bivariate analysis with proper interpretation. (You may choose to include only those variables which were significant in the model building)

There are 58 variables in the dataset and there are many variables that are highly correlated and insignificant in predicting the dependent variable class. Let's see the heatmap of correlation between all the 57 independent variables

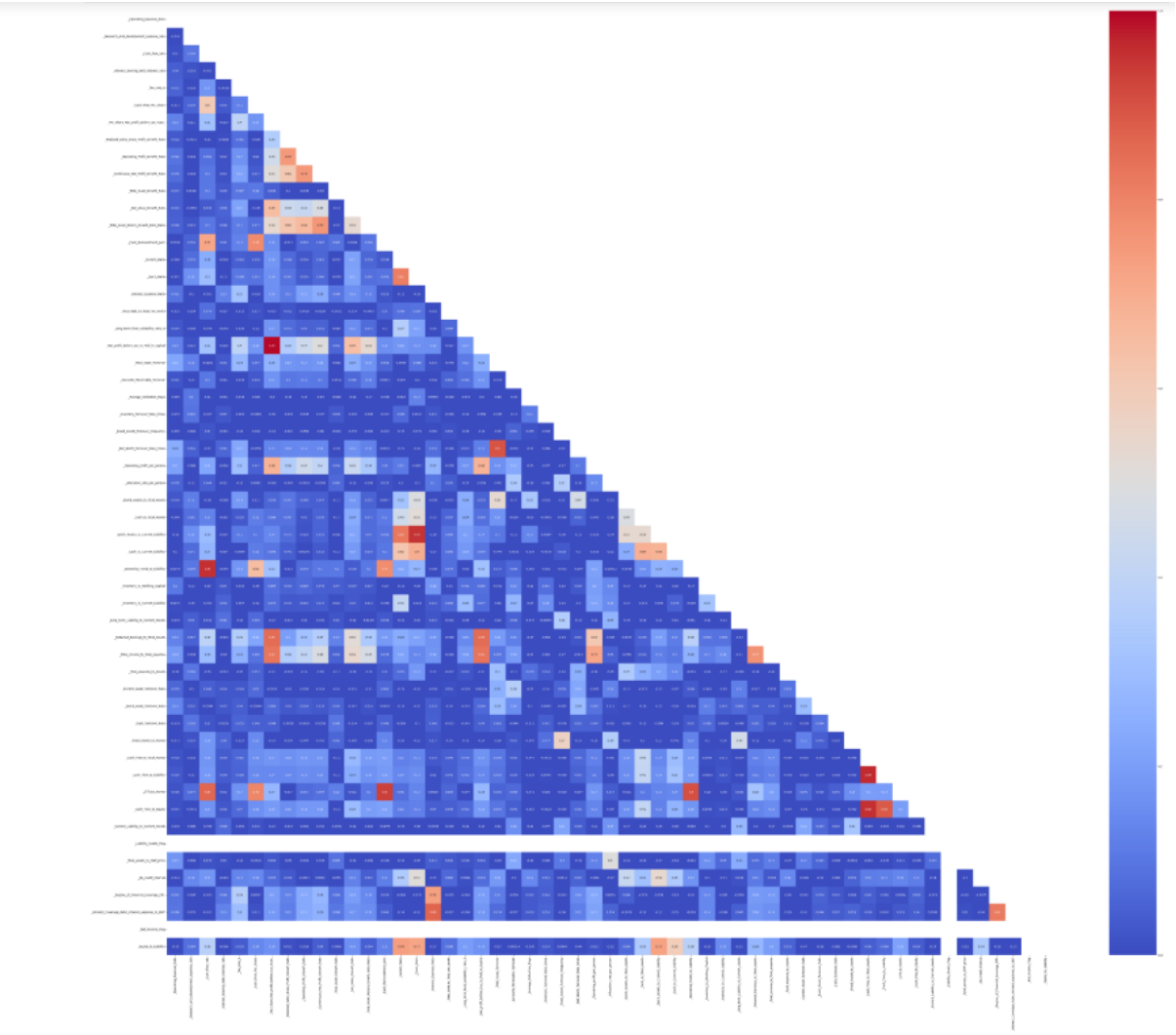


Figure 7:Heatmap of Independent Variables

As we can see there are many variables that are highly correlated as indicated by colours at far end of spectrum. There is a strong positive correlation as well as negative correlation existing among several variables which creates a problem of 'Multi-collinearity'. Hence, we need to treat the collinearity to obtain an accurate and efficient model .We will eliminate those variables and then do a univariate and Bivariate analysis

1. We will be dropping variables 'Co_Code','Co_Name' because they are insignificant as they are just company Id and company name .Then we eliminate all the correlated features in the dataset using Variance Inflation Factor method . Any variable with VIF more than 5 will be dropped from analysis .

After VIF method we are left with 44 variables , we also drop '_Liability_Assets_Flag','_Net_Income_Flag' from our analysis because they only have 0's

So, we are left with 42 variables .

2. Next , we use Recursive Feature Elimination(RFE) which is a popular feature selection algorithm.

RFE is effective at selecting those features (columns) in a training dataset that are most relevant in predicting the target variable.

There are two important configuration options when using RFE: the choice in the number of features to select and the choice of the algorithm used to help choose features.

For our analysis , we will be selecting Logistic Regression as the algorithm and number of features as 11 .

After eliminating the insignificant variables using RFE we are left with following 11 important variables that are the significant in model building

'_Research_and_development_expense_rate',

'_Interest_bearing_debt_interest_rate', '_Quick_Ratio',

'_Total_debt_to_Total_net_worth', '_Accounts_Receivable_Turnover',

'_Average_Collection_Days', '_Allocation_rate_per_person',

'_Retained_Earnings_to_Total_Assets',

'_Total_income_to_Total_expense', '_Cash_Turnover_Rate',

'_Equity_to_Liability'

And 'Default' Is the dependent variable.

Let's do descriptive analysis of these variables.

5 number summary of the independent variables is as below

	count	mean	std	min	25%	50%	75%	max
Default	2058.0000	0.1069	0.3091	0.0000	0.0000	0.0000	0.0000	1.0000
_Research_and_development_expense_rate	2058.0000	1208634256.5598	2144568158.0809	0.0000	0.0000	0.0002	1550000000.0000	9980000000.0000
_Interest_bearing_debt_interest_rate	2058.0000	11130223.5191	90425949.0445	0.0000	0.0003	0.0005	0.0007	990000000.0000
_Quick_Ratio	2058.0000	27755102.0498	444865390.4716	0.0000	0.0029	0.0053	0.0089	9230000000.0000
_Total_debt_to_Total_net_worth	2037.0000	10714285.7258	269696017.5877	0.0000	0.0039	0.0073	0.0131	9940000000.0000
_Accounts_Receivable_Turnover	2058.0000	41598639.4598	504767266.5944	0.0000	0.0007	0.0011	0.0019	9740000000.0000
_Average_Collection_Days	2058.0000	26297862.0103	410996733.8326	0.0000	0.0036	0.0060	0.0086	8800000000.0000
_Allocation_rate_per_person	2058.0000	5725558.8211	197949961.0648	0.0000	0.0047	0.0106	0.0246	8280000000.0000
_Retained_Earnings_to_Total_Assets	2058.0000	0.9304	0.0298	0.0000	0.9279	0.9351	0.9409	0.9728
_Total_income_to_Total_expense	2058.0000	0.0024	0.0005	0.0000	0.0022	0.0023	0.0024	0.0103
_Cash_Turnover_Rate	2058.0000	2653695544.2182	2821244732.1900	0.0001	0.0017	1730000000.0000	4550000000.0000	9990000000.0000
_Equity_to_Liability	2058.0000	0.0425	0.0595	0.0039	0.0204	0.0285	0.0434	1.0000

Table 4: Data Summary of Important Variables

As we can see the variables are on different scales and they need to be scaled before modelling . Most variables above have highly right skewed distribution because mean is very high compared to median.

- Retained_earnings_to_total_assets, Total_Income_to_Total_expense are having a near normal distribution because the mean and median are very close
- 'Research_and_development_expense_rate','Interest_bearing_debt_interest_rate','Quick_Ratio',Total_debt_to_Total_net_worth','Accounts_Receivable_Turnover','Allocation_rate_per_person', Average_Collection_Days', 'Cash_Turnover_Rate', these variables have very high standard deviation indicating that these variables are very high variation is the data.

Univariate analysis

Let's look at the boxplot of the independent variables

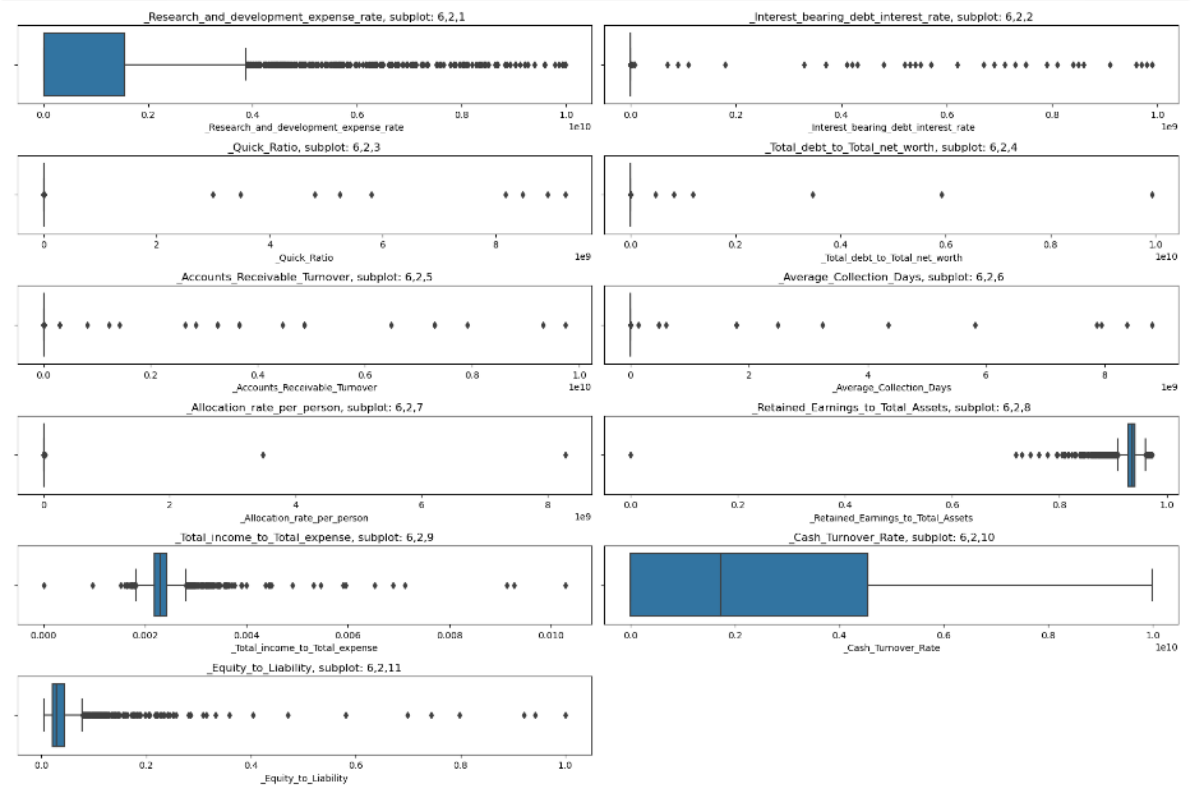


Figure 8: Boxplot of Important Independent variables

From the boxplot above, we see all variables except Cash_Turnover_Rate have outliers

Let's look at the distribution plot of the independent variables

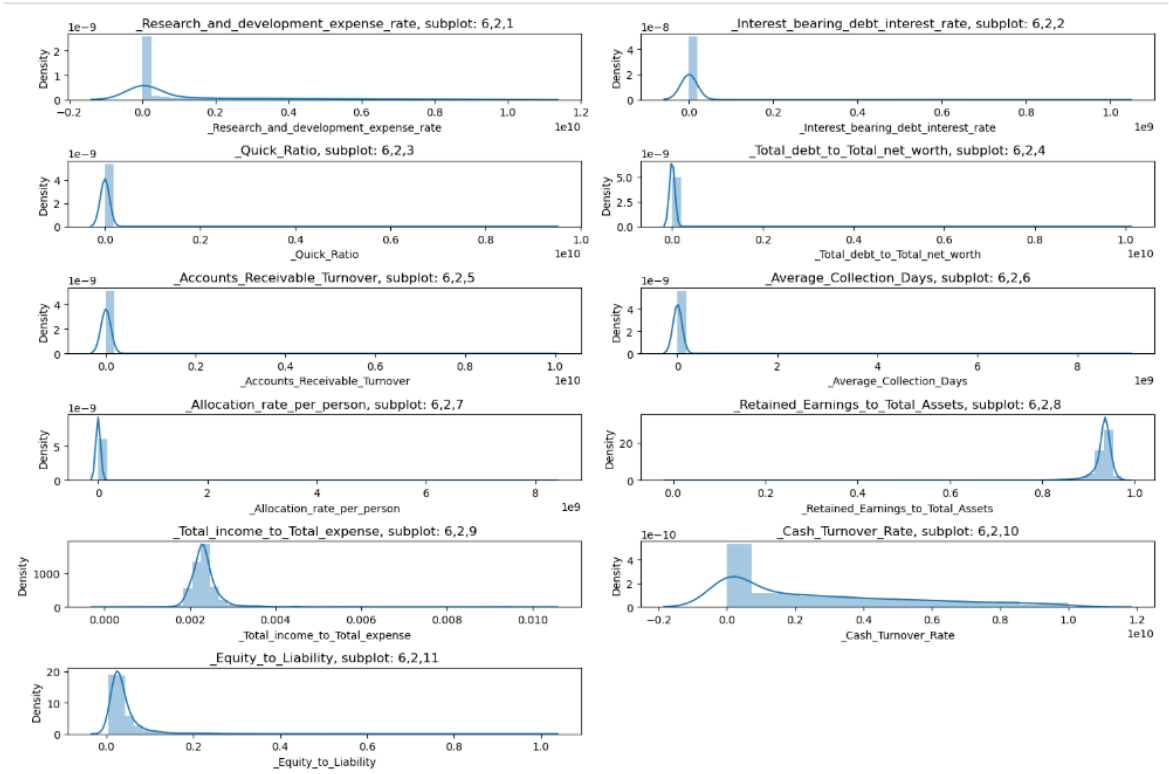


Figure 9:Countplot of Independent variables

Let's see the skewness of the variables. **Skewness** is a statistical term and it is a way to estimate or measure the shape of a distribution. It is an important statistical methodology that is used to estimate the asymmetrical behaviour rather than computing frequency distribution

Features	Skewness
_Research_and_development_expense_rate	1.99
_Interest_bearing_debt_interest_rate	8.67
_Quick_Ratio	17.33
_Total_debt_to_Total_net_worth	30.83
_Accounts_Receivable_Turnover	14.19
_Average_Collection_Days	17.99
_Allocation_rate_per_person	38.17
_Retained_Earnings_to_Total_Assets	-16.14
_Total_income_to_Total_expense	8.02
_Cash_Turnover_Rate	0.89
_Equity_to_Liability	9.14

Table 5 Skewness of variables

Distribution on the basis of skewness value:

- Skewness = 0: Then normally distributed.
- Skewness > 0: Then more weight in the left tail of the distribution.
- Skewness < 0: Then more weight in the right tail of the distribution.

- All the variables except Retained_earning_to_total_assets have skewness > 0. As we can see the variables Allocation_rate_per_person, Total_debt_to_Total_net_worth is highly right skewed with skewness of 38.17 and 30.83 respectively and Retained_earnings_to_total_assets are highly left skewed with skewness of -16.14

Let's see the Kurtosis of the Independent variables
Kurtosis is also a statistical term and an important characteristic of frequency distribution. It determines whether a distribution is heavy-tailed in respect of the normal distribution. It provides information about the shape of a frequency distribution.

- kurtosis for normal distribution is equal to 3.
- For a distribution having kurtosis < 3: It is called platykurtic .
- For a distribution having kurtosis > 3, It is called leptokurtic and it signifies that it tries to produce more outliers rather than the normal distribution.

Default	4.4881
_Research_and_development_expense_rate	3.2953
_Interest_bearing_debt_interest_rate	77.0317
_Quick_Ratio	314.5551
_Total_debt_to_Total_net_worth	1029.9200
_Accounts_Receivable_Turnover	218.8232
_Average_Collection_Days	340.3024
_Allocation_rate_per_person	1531.6993
_Retained_Earnings_to_Total_Assets	468.8240
_Total_income_to_Total_expense	105.1667
_Cash_Turnover_Rate	-0.3368
_Equity_to_Liability	115.4525
dtype: float64	

Table 6 Kurtosis of variables

None of the Variables have Kurtosis 3 . Reaserach_and_development_expense_rate is close to 3 indicating its near normal distribution.

Cash_Turnover_rate has Kurtosis less than 3 indicating that it has platykurtic distribution.
A platykurtic distribution have thinner tails than a normal distribution, leading to less extreme positive or negative events as also indicated in distribution plot above and as shown by boxplot, this variable doesn't have outliers .

All other variables have value > 3 indicating , It is called leptokurtic it is the opposite of a platykurtic distribution. It has an excess kurtosis that is positive and it signifies that it tries to produce more outliers rather than the normal distribution. As we can see in distribution plot these variables are more peaked than a normal distribution with longer tails and as can be seen in boxplot , all these variables have outliers

Let's look at the distribution of 'Default' is the dependent variable

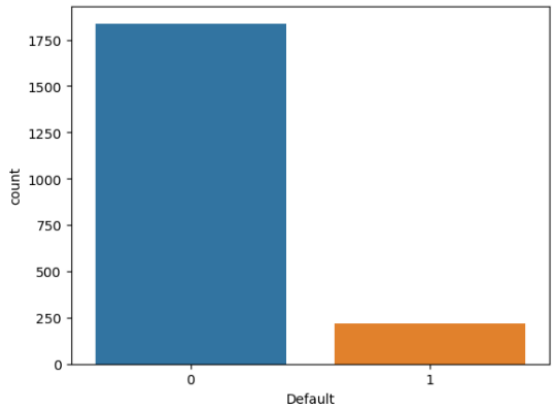


Figure 10:Count plot of Default Variable

89% of records in the dataset having Default class as 0 and 11% of records in the dataset are having Default class as '1'. There is a class imbalance in this is dataset with majority of records having default class as 0.

Bivariate Analysis of Independent Variables against Dependent variable

This is a bivariate analysis of the independent variables vis-à-vis the dependent variable 'Default'

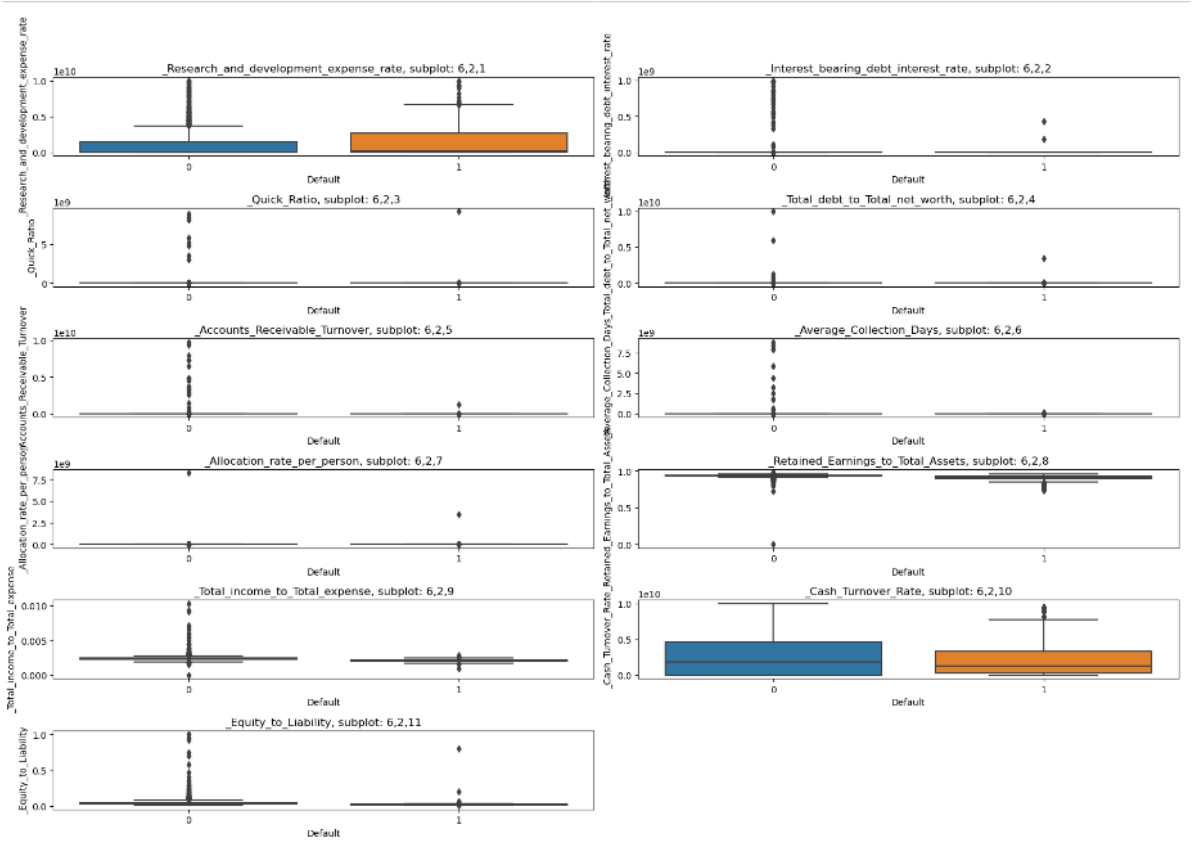


Figure 11:Bivariate Analysis of independent vs dependent variables

This is a pairwise analysis of the independent variables with hue as 'Default'

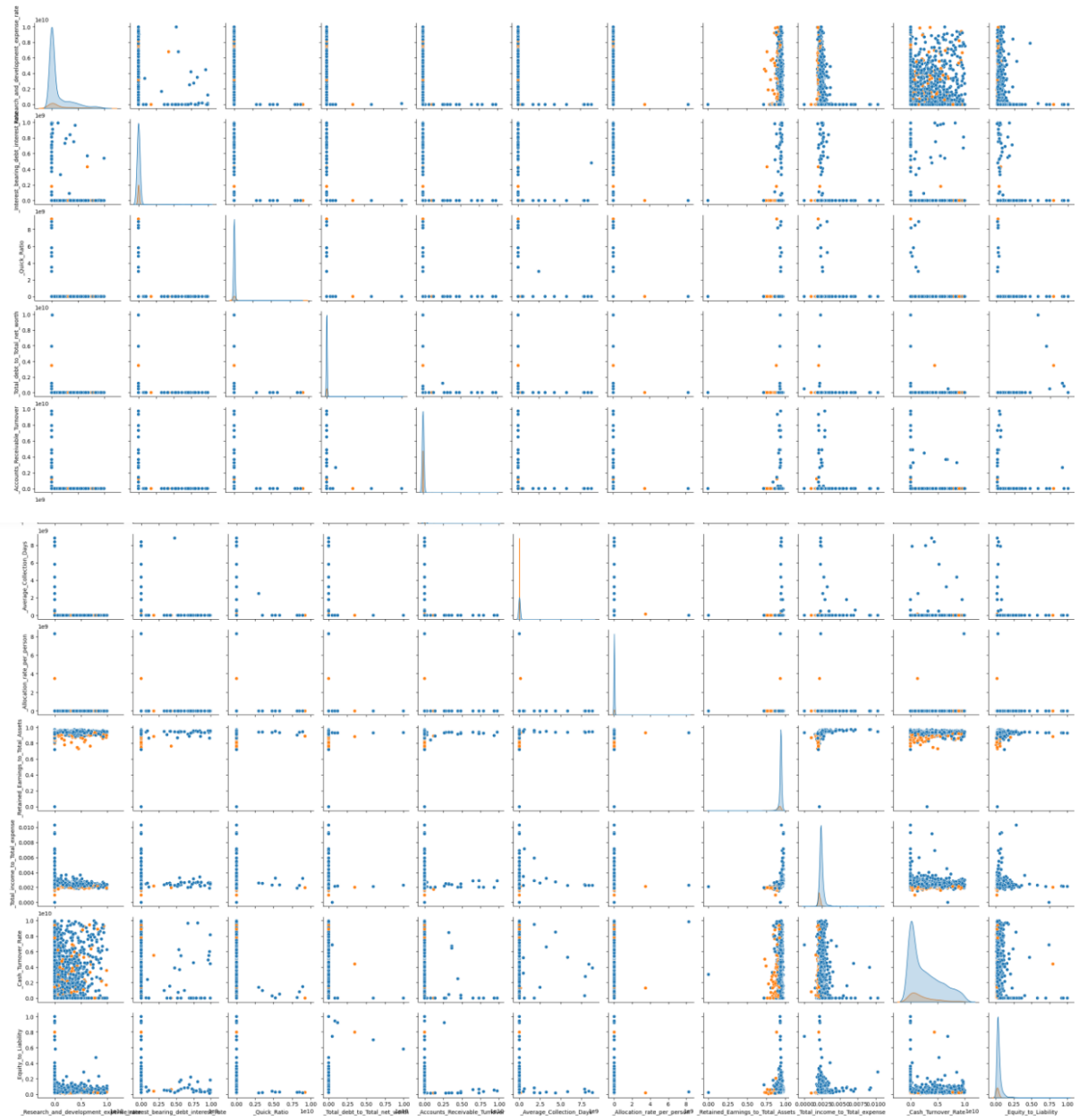


Figure 12 pair plot of Independent variables

Inference from Bivariate Analysis

- Average of _Research_and_development_expense_rate for companies that are defaulting is higher than average of _Research_and_development_expense_rate ratio for companies that are not defaulting.
- Average of _Equity_to_Liability, for companies that are not defaulting is slightly higher than average of _Equity_to_Liability, ratio for companies that are defaulting.
- Average of _Cash_Turnover_Rate ratio for companies that are not defaulting is slightly higher than average of _Cash_Turnover_Rate for companies that are defaulting.
- Average of Total_Income_to_total_expense for companies that are not defaulting is Higer than average of Total_Income_to_total_expense for companies that are defaulting.
- Average of Retained_Earnings_to_Total_Assets for companies that are not defaulting is higher than average of Retained_Earnings_to_Total_Assets ratio for companies that are defaulting.

- Bivariate analysis of the pairwise plot of independent variables indicates that independent variables are not displaying any correlation between themselves
- Equity_to_Liability, Retained_Earnings_to_Total_Assets ratio shows clear separation in classes of 0 and 1 as shown in the pair plot, so these could one of the most important predictors

Train Test Split.

We will be dropping the columns 'Co_Code','Co_Name' because these 2 columns are not of any significance for our model building .

After we split the dataset, we have to scale the dataset to ensure that no single feature dominates the distance calculations in an algorithm, and can help to improve the performance of the algorithm

We will scale the dataset using StandardScaler() function .

We will also impute null values in the dataset after the train test split . Since we are not imputing null values with constant values like mean or median instead using KNN imputer , the missing values need to be handled after splitting the training and test sets. Otherwise, the model will be given information about the test set which causes data leakage.

Using the train-test split function we split the dataset in ratio 67:33 at random state 42. It is desirable to split the dataset into train and test sets in a way that preserves the same proportions of examples in each class as observed in the original dataset. Therefore, we pass the parameter stratify in the Train-test split function

After the split Training dataset has 1378 observations and 56 columns and Training dataset has 680 observations and 56 columns .

Training set has proportion of both classes as below

```
0    0.893324
1    0.106676
```

Testing set has proportion of both classes as below

```
0    0.892647
1    0.107353
```

As we can see both training and test data have proportions of classes same as the original data

Let’s see the sample of Training and Testing Data set

	_Operating_Expense_Rate	_Research_and_development_expense_rate	_Cash_flow_rate	_Interest_bearing_debt_interest_rate	_Tax_rate_A	_Cash_Flow_Per_
0	-0.63	-0.65	2.09	1.73	-0.82	
1	-0.63	0.57	0.26	-0.82	1.52	
2	-0.63	-0.65	-0.72	0.73	-0.82	
3	1.20	1.34	-0.25	-0.71	-0.82	
4	-0.63	-0.19	-0.52	0.28	-0.82	
...	...	...	...	...	...	
1373	0.68	-0.65	0.12	-1.15	1.02	
1374	-0.63	0.31	-0.19	-0.20	0.78	
1375	1.20	-0.65	0.01	0.99	0.53	
1376	-0.63	2.04	0.67	0.30	0.77	
1377	-0.63	-0.57	-0.80	0.03	-0.82	

1378 rows × 56 columns

rice	_No_credit_Interval	_Degree_of_Financial_Leverage_DFL	_Interest_Coverage_Ratio_Interest_expense_to_EBIT	_Net_Income_Flag	_Equity_to_Liability	Default
-0.67	-0.31	-0.47	-0.52	0.00	0.07	0.00
-0.57	0.18	-0.30	-0.33	0.00	-0.23	0.00
1.63	0.01	1.81	1.88	0.00	-0.63	0.00
0.90	0.52	-1.36	-1.77	0.00	-0.69	0.00
0.45	-0.36	-1.75	-1.77	0.00	-0.44	0.00
...	...	...	...	...	...	...
-0.47	1.92	-0.29	-0.32	0.00	0.29	1.00
0.24	-0.06	-0.11	-0.12	0.00	-0.66	0.00
2.15	-0.81	1.64	1.10	0.00	-0.47	1.00
2.15	0.18	-0.21	-0.22	0.00	1.01	0.00
2.15	-1.81	-0.43	-0.47	0.00	-0.88	0.00

Figure 13:Scaled Training Dataset

	_Operating_Expense_Rate	_Research_and_development_expense_rate	_Cash_flow_rate	_Interest_bearing_debt_interest_rate	_Tax_rate_A	_Cash_Flow_Per_S
0	-0.64	0.16	1.29	1.46	1.14	
1	-0.64	-0.66	-0.97	-0.50	-0.72	
2	-0.64	-0.66	-0.35	-1.61	-0.82	
3	-0.64	-0.66	2.09	0.96	1.35	
4	0.56	-0.66	-0.39	-1.61	1.38	
...	...	...	...	...	...	
675	-0.64	-0.66	-0.26	1.09	-0.82	
676	2.30	1.42	1.10	1.04	-0.82	
677	1.21	-0.62	0.26	-0.60	-0.79	
678	1.75	-0.66	-0.23	-0.88	-0.67	
679	-0.64	-0.66	0.37	1.17	-0.82	

680 rows × 56 columns

price	_No_credit_Interval	_Degree_of_Financial_Leverage_DFL	_Interest_Coverage_Ratio_Interest_expense_to_EBIT	_Net_Income_Flag	_Equity_to_Liability	Default
-0.82	0.15	0.16	0.13	0.00	-0.18	0.00
-0.50	-1.81	-0.20	-0.22	0.00	-0.75	0.00
-0.89	0.30	-0.36	-0.40	0.00	2.22	0.00
-0.21	0.12	-0.29	-0.31	0.00	1.82	0.00
-0.68	0.98	-0.36	-0.39	0.00	-0.17	0.00
...	...	...	...	...	...	...
-0.47	-0.25	1.81	1.88	0.00	-0.79	0.00
-0.65	-0.15	-1.75	1.88	0.00	-0.22	0.00
-0.78	1.92	-0.29	-0.32	0.00	2.22	0.00
1.75	-1.81	-0.36	-0.39	0.00	-0.56	0.00
-0.90	-0.07	-0.08	-0.10	0.00	-0.76	0.00

Figure 14:Scaled Testing Dataset

As we can see the training and testing dataset is now ready for model building with all the data pre-processing activities like outlier treatment, scaling and missing values treatment done

Build Logistic Regression Model (using statsmodels library) on most important variables on train dataset and choose the optimum cut-off. Also showcase your model building approach.

Logistic Regression using statsmodels:

To conduct detailed statistical analysis and hypothesis testing related to Logistic regression, statsmodels is more suitable.

Steps in Building Logistic Regression Model

1. **Eliminate features with Multicollinearity**

Variance Inflation Factor (VIF) is a measure to detect multicollinearity in multiple regression models.

It quantifies how much the variance of a regression coefficient is inflated due to correlations with other predictors. Variance Inflation Factor tells- How good an independent variable can be explained as a Linear combination of other independent variables (i.e., to identify Multicollinearity)

We will use VIF when building a multiple regression model to identify highly correlated variables, which may cause instability and hinder accurate interpretation.

High VIF values indicate significant multicollinearity, requiring attention, such as removing correlated variables or applying advanced techniques like PCA or regularization to mitigate the issue.

We have 55 independent variables and we must remove the redundant columns to avoid multicollinearity before model building

Variables with VIF > 5 are not suitable as it is mostly compensated by other Independent variables .

Using VIF method , we check the multicollinearity between the predictors and eliminate the variables one by one .Any variable with VIF more than 5 has been dropped until we reach a stage where we are left with variables which don’t have VIF greater than 5.

Listed below are the VIF values of variables in descending order of VIF value

	variables	VIF
19	_Net_profit_before_tax_to_Paid_in_capital	82.18
6	_Per_Share_Net_profit_before_tax_Yuan_	82.00
43	_Cash_Flow_to_Total_Assets	67.10
45	_CFO_to_Assets	31.90
30	_Quick_Assets_to_Current_Liability	23.71
44	_Cash_Flow_to_Liability	22.83
46	_Cash_Flow_to_Equity	22.67
32	_Operating_Funds_to_Liability	22.01
2	_Cash_flow_rate	16.54
13	_Cash_Reinvestment_perc	13.67
14	_Current_Ratio	12.43
15	_Quick_Ratio	11.76
20	_Total_Asset_Turnover	11.21
25	_Net_Worth_Turnover_Rate_times	10.93
28	_Quick_Assets_to_Total_Assets	6.88
52	_Interest_Coverage_Ratio_Interest_expense_to_EBIT	6.41
36	_Retained_Earnings_to_Total_Assets	5.45
54	_Equity_to_Liability	5.40
42	_Fixed_Assets_to_Assets	5.20
37	_Total_income_to_Total_expense	5.11
16	_Interest_Expense_Ratio	4.62
51	_Degree_of_Financial_Leverage_DFL	4.09
8	_Operating_Profit_Growth_Rate	3.81
9	_Continuous_Net_Profit_Growth_Rate	3.61
31	_Cash_to_Current_Liability	3.48
12	_Total_Asset_Return_Growth_Rate_Ratio	3.47
34	_Inventory_to_Current_Liability	3.36
5	_Cash_Flow_Per_Share	3.34
26	_Operating_profit_per_person	3.25

Table 7: VIF values of the variables

After eliminating using VIF method we get the following variables

	variables	VIF
35	_Fixed_Assets_to_Assets	4.76
12	_Cash_Reinvestment_perc	4.26
30	_Total_income_to_Total_expense	4.20
13	_Quick_Ratio	3.98
41	_Equity_to_Liability	3.82
7	_Operating_Profit_Growth_Rate	3.76
2	_Cash_flow_rate	3.72
29	_Retained_Earnings_to_Total_Assets	3.62
8	_Continuous_Net_Profit_Growth_Rate	3.56
25	_Cash_to_Current_Liability	3.21
11	_Total_Asset_Return_Growth_Rate_Ratio	3.18
22	_Operating_profit_per_person	3.12
5	_Cash_Flow_Per_Share	3.04
6	_Realized_Sales_Gross_Profit_Growth_Rate	2.92
23	_Allocation_rate_per_person	2.90
16	_Long_term_fund_suitability_ratio_A	2.84
17	_Accounts_Receivable_Turnover	2.78
21	_Net_Worth_Turnover_Rate_times	2.76
14	_Interest_Expense_Ratio	2.76
40	_Degree_of_Financial_Leverage_DFL	2.72
24	_Cash_to_Total_Assets	2.52
18	_Average_Collection_Days	2.46
10	_Net_Value_Growth_Rate	2.41
31	_Total_expense_to_Assets	2.23
20	_Fixed_Assets_Turnover_Frequency	1.97
38	_Total_assets_to_GNP_price	1.74
37	_Current_Liability_to_Current_Assets	1.67
27	_Inventory_to_Current_Liability	1.67
28	_Long_term_Liability_to_Current_Assets	1.66
39	_No_credit_Interval	1.55
32	_Current_Asset_Turnover_Rate	1.52
4	_Tax_rate_A	1.49

Table 8:VIF after elimination

As we can see now all variables are having VIF less than 5

2. **Using above predictors and dependent variable as 'Default' we build the logistic regression model using statsmodels**.

The most important variables after eliminating using VIF are

'_Operating_Expense_Rate', '_Research_and_development_expense_rate','_Cash_flow_rate', '_Interest_bearing_debt_interest_rate', '_Tax_rate_A', '_Cash_Flow_Per_Share', '_Realized_Sales_Gross_Profit_Growth_Rate','_Operating_Profit_Growth_Rate', '_Continuous_Net_Profit_Growth_Rate','_Total_Asset_Growth_Rate', '_Net_Value_Growth_Rate', '_Total_Asset_Return_Growth_Rate_Ratio', '_Cash_Reinvestment_perc', '_Quick_Ratio', '_Interest_Expense_Ratio','_Total_debt_to_Total_net_worth', '_Long_term_fund_suitability_ratio_A', '_Accounts_Receivable_Turnover', '_Average_Collection_Days', '_Inventory_Turnover_Rate_times', '_Fixed_Assets_Turnover_Frequency', '_Net_Worth_Turnover_Rate_times', '_Operating_profit_per_person', '_Allocation_rate_per_person', '_Cash_to_Total_Assets','_Cash_to_Current_Liability', '_Inventory_to_Working_Capital', '_Inventory_to_Current_Liability', '_Long_term_Liability_to_Current_Assets', '_Retained_Earnings_to_Total_Assets', '_Total_income_to_Total_expense', '_Total_expense_to_Assets', '_Current_Asset_Turnover_Rate', '_Quick_Asset_Turnover_Rate', '_Cash_Turnover_Rate', '_Fixed_Assets_to_Assets', '_Cash_Flow_to_Liability', '_Current_Liability_to_Current_Assets', '_Total_assets_to_GNP_price', '_No_credit_Interval', '_Degree_of_Financial_Leverage_DFL' ,'_Equity_to_Liability'

We build the Logistic regression model Model 1 using above predictors and 'default' variable as target variable

Dep. Variable:	Default	**No. Observations:**	1378
Model:	Logit	**Df Residuals:**	1333
Method:	MLE	**Df Model:**	44
Date:	Mon, 06 Nov 2023	**Pseudo R-squ.:**	0.4552
Time:	18:28:32	**Log-Likelihood:**	-254.89
converged:	True	**LL-Null:**	-467.84
Covariance Type:	nonrobust	**LLR p-value:**	5.547e-64

	coef	std err	z	P>|z|	[0.025	0.975]
Intercept	-4.3518	0.311	-13.985	0.000	-4.962	-3.742
_Operating_Expense_Rate	0.1634	0.143	1.146	0.252	-0.116	0.443
_Research_and_development_expense_rate	0.4085	0.119	3.436	0.001	0.175	0.642
_Cash_flow_rate	0.1807	0.333	0.542	0.588	-0.473	0.834
_Interest_bearing_debt_interest_rate	0.4371	0.152	2.881	0.004	0.140	0.734
_Tax_rate_A	-0.2137	0.183	-1.170	0.242	-0.572	0.144
_Cash_Flow_Per_Share	-0.2210	0.286	-0.773	0.440	-0.782	0.340
_Realized_Sales_Gross_Profit_Growth_Rate	-0.0924	0.150	-0.615	0.538	-0.386	0.202
_Operating_Profit_Growth_Rate	0.0661	0.169	0.391	0.696	-0.266	0.398
_Continuous_Net_Profit_Growth_Rate	-0.2021	0.147	-1.374	0.170	-0.490	0.086
_Total_Asset_Growth_Rate	-0.0766	0.143	-0.536	0.592	-0.356	0.203
_Net_Value_Growth_Rate	-0.1846	0.179	-1.031	0.303	-0.536	0.166
_Total_Asset_Return_Growth_Rate_Ratio	0.0927	0.183	0.508	0.611	-0.265	0.451
_Cash_Reinvestment_perc	-0.0246	0.264	-0.093	0.926	-0.542	0.493

Table 9:Model1Logit

_Long_term_fund_suitability_ratio_A	0.2235	0.173	1.289	0.197	-0.116	0.563
_Accounts_Receivable_Turnover	-0.4301	0.181	-2.371	0.018	-0.786	-0.075
_Average_Collection_Days	0.3595	0.159	2.261	0.024	0.048	0.671
_Inventory_Turnover_Rate_times	-0.0852	0.133	-0.641	0.521	-0.346	0.175
_Fixed_Assets_Turnover_Frequency	0.2173	0.130	1.670	0.095	-0.038	0.472
_Net_Worth_Turnover_Rate_times	-0.0393	0.190	-0.207	0.836	-0.411	0.332
_Operating_profit_per_person	0.1552	0.244	0.636	0.525	-0.323	0.633
_Allocation_rate_per_person	0.2321	0.165	1.403	0.161	-0.092	0.556
_Cash_to_Total_Assets	0.0288	0.254	0.113	0.910	-0.469	0.527
_Cash_to_Current_Liability	0.1901	0.145	1.308	0.191	-0.095	0.475
_Inventory_to_Working_Capital	-0.0296	0.114	-0.260	0.795	-0.252	0.193
_Inventory_to_Current_Liability	-0.0317	0.187	-0.169	0.866	-0.399	0.336
_Long_term_Liability_to_Current_Assets	-0.1935	0.137	-1.413	0.158	-0.462	0.075
_Retained_Earnings_to_Total_Assets	-0.8495	0.200	-4.241	0.000	-1.242	-0.457
_Total_income_to_Total_expense	-0.4664	0.345	-1.351	0.177	-1.143	0.210
_Total_expense_to_Assets	0.2924	0.181	1.615	0.106	-0.063	0.647
_Current_Asset_Turnover_Rate	-0.0981	0.143	-0.687	0.492	-0.378	0.182
_Quick_Asset_Turnover_Rate	0.0029	0.145	0.020	0.984	-0.281	0.287
_Cash_Turnover_Rate	-0.2895	0.144	-2.015	0.044	-0.571	-0.008
_Fixed_Assets_to_Assets	0.2057	0.185	1.115	0.265	-0.156	0.567
_Cash_Flow_to_Liability	0.1890	0.345	0.548	0.584	-0.487	0.865
_Cash_Flow_to_Equity	-0.3090	0.239	-1.290	0.197	-0.778	0.160
_Current_Liability_to_Current_Assets	-0.0182	0.128	-0.142	0.887	-0.269	0.233
_Total_assets_to_GNP_price	0.1616	0.135	1.199	0.230	-0.103	0.426
_No_credit_Interval	0.0182	0.125	0.146	0.884	-0.227	0.263
_Degree_of_Financial_Leverage_DFL	0.1213	0.144	0.843	0.399	-0.161	0.403
_Interest_Coverage_Ratio_Interest_expense_to_EBIT	-0.0772	0.191	-0.404	0.686	-0.451	0.297
_Equity_to_Liability	-1.4097	0.463	-3.045	0.002	-2.317	-0.502

Table 10: Model 1 Summary

3. Eliminate more dependent variables that are not good predictors using p-value and arrive at the final model

The p values in regression help determine whether the relationships that you observe in your sample also exist in the larger population. The linear regression p value for each independent variable tests the null hypothesis that the variable has no correlation with the dependent variable. If there is no correlation, there is no association between the changes in the independent variable and the shifts in the dependent variable. In other words, there is insufficient evidence to conclude that there is an effect at the population level.

If the p-value for a variable is less than the significance level, your sample data provide enough evidence to reject the null hypothesis for the entire population. Your data favour the hypothesis that there *is* a non-zero correlation. Changes in the independent variable *are* associated with changes in the dependent variable at the population level. This variable is statistically significant and probably a worthwhile addition to your regression model.

On the other hand, when a p value in regression is greater than the significance level, it indicates there is insufficient evidence in your sample to conclude that a non-zero correlation exists.

With a significance value as 0.05, we can see there are lot of variables with p value> significance level of 0.05, this indicates that these variables are not very good predictors .

We will be dropping variables one by one and check the P values after each iteration ,

After 32 iterations we have a final model with just 11 predictors that have significant p values

Logit Regression Results

Dep. Variable:	Default	**No. Observations:**	1378
Model:	Logit	**Df Residuals:**	1366
Method:	MLE	**Df Model:**	11
Date:	Wed, 15 Nov 2023	**Pseudo R-squ.:**	0.4326
Time:	15:46:47	**Log-Likelihood:**	-265.45
converged:	True	**LL-Null:**	-467.84
Covariance Type:	nonrobust	**LLR p-value:**	5.862e-80

	coef	std err	z	P>|z|	[0.025	0.975]
Intercept	-4.0985	0.255	-16.069	0.000	-4.598	-3.599
_Research_and_development_expense_rate	0.4492	0.112	4.025	0.000	0.230	0.668
_Interest_bearing_debt_interest_rate	0.3042	0.140	2.169	0.030	0.029	0.579
_Continuous_Net_Profit_Growth_Rate	-0.3630	0.114	-3.187	0.001	-0.586	-0.140
_Quick_Ratio	-1.0579	0.261	-4.058	0.000	-1.569	-0.547
_Total_debt_to_Total_net_worth	0.2227	0.063	3.542	0.000	0.099	0.346
_Accounts_Receivable_Turnover	-0.7586	0.156	-4.877	0.000	-1.063	-0.454
_Allocation_rate_per_person	0.5777	0.145	3.992	0.000	0.294	0.861
_Retained_Earnings_to_Total_Assets	-1.1189	0.156	-7.192	0.000	-1.424	-0.814
_Total_expense_to_Assets	0.3858	0.155	2.491	0.013	0.082	0.689
_Cash_Turnover_Rate	-0.3579	0.132	-2.709	0.007	-0.617	-0.099
_Equity_to_Liability	-1.0412	0.261	-3.990	0.000	-1.553	-0.530

Table 11: Model 32 Summary

Let's predict on train data using the above model . The performance of this model on train data is as follows

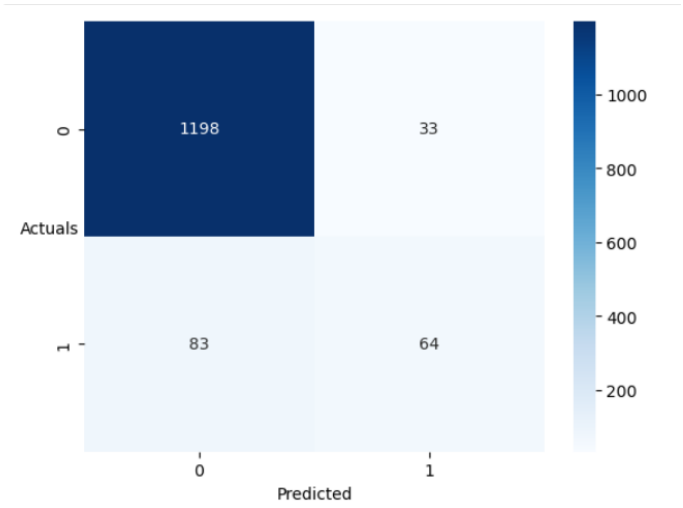


Figure 15: Confusion Matrix of Logistic regression Train Model

	precision	recall	f1-score	support
0.0	0.935	0.973	0.954	1231
1.0	0.660	0.435	0.525	147
accuracy			0.916	1378
macro avg	0.798	0.704	0.739	1378
weighted avg	0.906	0.916	0.908	1378

Table 12: Classification Matrix of Logistic regression Train Model

Let's predict on test data using the above model . The performance of this model on test data is as follows

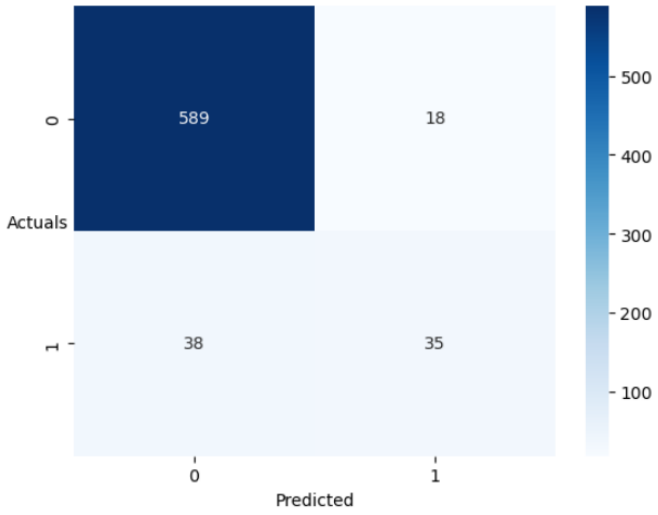


Figure 16: Confusion Matrix of Logistic regression Test Model

	precision	recall	f1-score	support
0.0	0.939	0.970	0.955	607
1.0	0.660	0.479	0.556	73
accuracy			0.918	680
macro avg	0.800	0.725	0.755	680
weighted avg	0.909	0.918	0.912	680

Table 13: Classification Matrix of Logistic regression Test Model

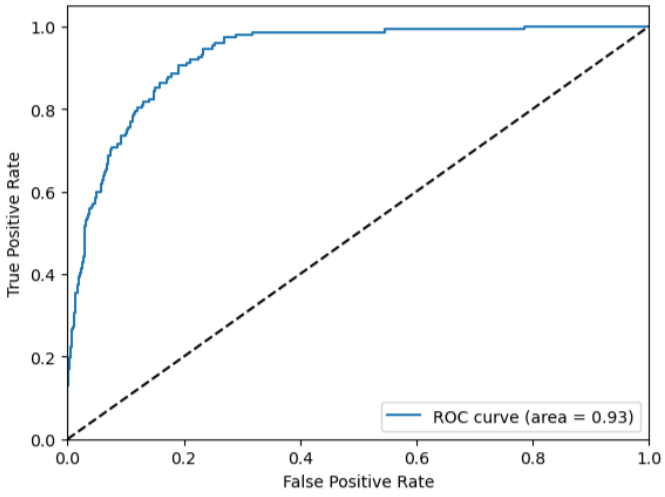


Figure 17 ROC curve of Logistic Regression Model on Train data with 0.5 threshold

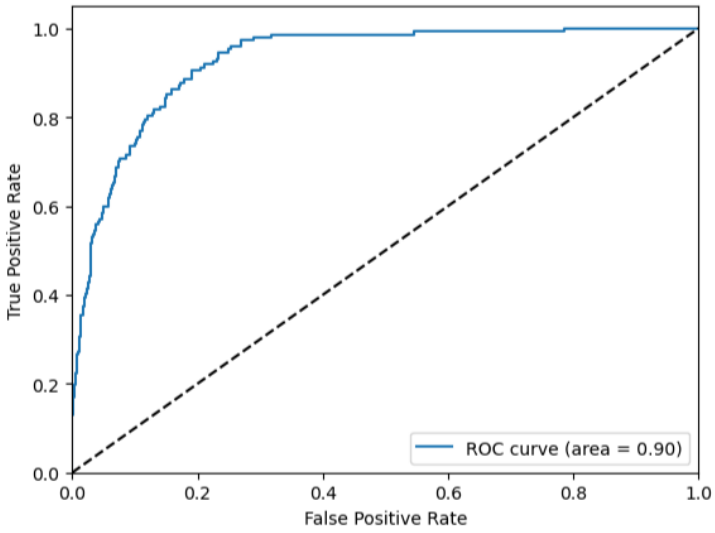


Figure 18 ROC curve of Logistic Regression Model on Test data with 0.5 threshold

Logistic regression predicts the probabilities of an object belonging to each class and makes binary classification based on the probabilities**.** The probability cut off threshold is by default 0.5 . If the probability is less than 0.5 its classified as 0 and If the probability is greater than 0.5 its classified as 1.

Interpretations

- With the cut off of 0.5 the accuracy on train test is 0.916 and on test set is 0.918 .
- We see that with this model the recall for class 0 is 0.973 on train data and for test data its 0.970
- We see that with this model the recall for class 1 is 0.435 on train data and for test data its 0.479
- The precision score with this model for class 1 is 0.66 for both train and test data .
- AUC score on Train data is 0.93 and on Test Data is 0.90.
- So, this model is neither overfitting nor underfitting which is good

- The significant difference in recall values between class 1 (default cases) and class 0 (non-default cases) indicates a severe class imbalance in the dataset. The model's ability to accurately identify non-default cases is much higher compared to default cases. This is because the dataset has a low number of default instances compared to non-default instances.
- The low recall of 0.48 for default cases implies a high false negative prediction. The model is missing a considerable number of companies that actually default on their debt obligations. This is a critical issue as identifying defaulting companies accurately is crucial for risk assessment and decision-making.

We are interested in default class 1 as we are interested to predict who will be potential defaulters We are interested in recall score more than precision score because we want to reduce the number of false negatives , that is we want to predict to number of customers who will be defaulting, so false negative will cost the bank more than false positive.

With this model the recall scores are very low and In order to increase the Recall score we need to reduce the probability threshold cut off.

To calculate the optimum cut-off, the steps are as below:

• We calculate the ROC curve using the true labels (the actual class values) and the predicted probabilities (the probabilities assigned to the positive class) for the training data.

• Using roc_curve function, we get three arrays: fpr, tpr, and thresholds. The fpr array contains the false positive rates at different classification thresholds, and the tpr array contains the corresponding true positive rates. The threshold array holds the actual classification thresholds.

• To find the optimal threshold, which maximizes the difference between the true positive rate and the false positive rate, we use the np.argmax function. This function helps us find the index of the maximum value in the expression tpr - fpr. By retrieving this index, we can determine the optimal threshold value.

• Finally, we extract the optimal threshold by indexing the threshold array with the index obtained from np. argmax. This value represents the threshold that strikes a balance between correctly classifying positive samples and avoiding false positives.

•The optimal threshold is 0.0836 at which the TPR is maximum and FPR is lowest .

Let see Model performance when we apply this threshold of 0.0836 on Train data

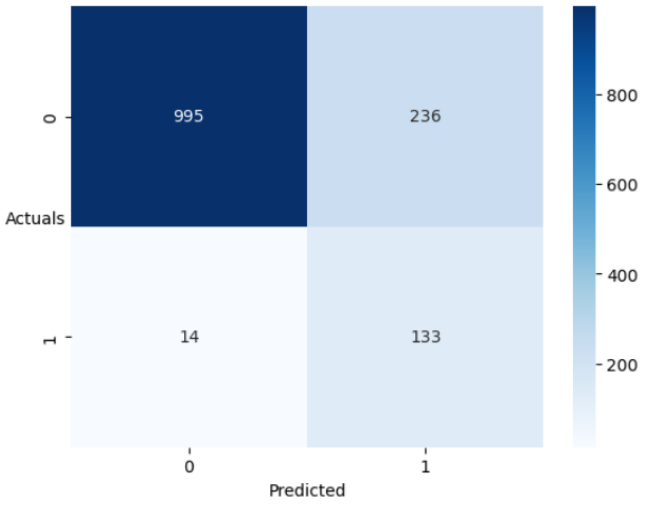


Figure 19: Confusion Matrix of Logistic regression Train Model with optimal threshold

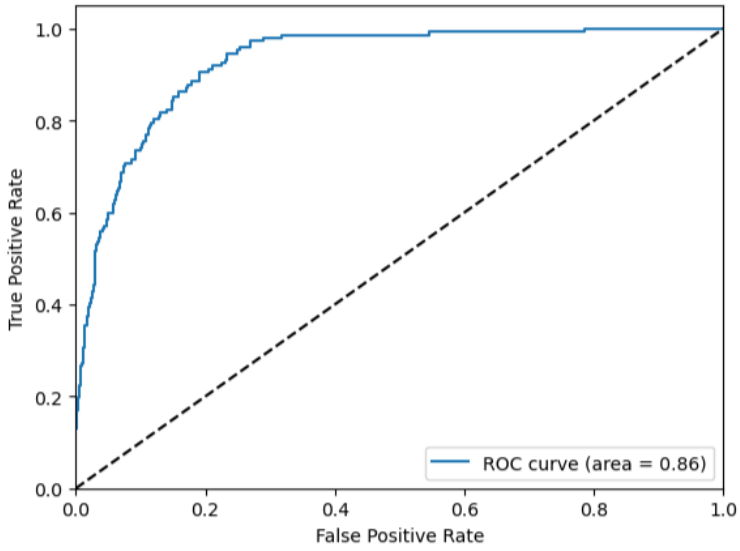


Figure 20 ROC curve of Logistic Regression Model on Train data with 0.0836 threshold

With this threshold the False negative has been brought down from 83 to 14 , but False positive has gone up from 18 to 236, which means we are predicting that more customers will default when they are not .

Let's see the classification report on train data

	precision	recall	f1-score	support
0.0	0.986	0.808	0.888	1231
1.0	0.360	0.905	0.516	147
accuracy			0.819	1378
macro avg	0.673	0.857	0.702	1378
weighted avg	0.919	0.819	0.849	1378

Figure 21: Classification Matrix of Logistic regression Train Model with optimal threshold

As we can see the Recall score for train data has gone up from 0.435 to 0.905.

But the precision has taken a hit and reduced to 0.36 from 0.66.

With this cut off the accuracy on train set is 0.82.

The AUC score on Train data is 0.86.

Recall scores of both Class 0 and that of Class 1 has improved with new threshold

Based on the recall score of 0.91 for Class1 , the model's performance can be considered very good. A higher recall score indicates a stronger ability to identify default cases accurately. As a result, the model has performed very well. Let's see the performance of this model on Test data

Validate the Model on Test Dataset and state the performance metrics. Also state interpretation from the model

Let's predict on test data using the above model with threshold 0.0836. The performance of this model on test data is as follows .

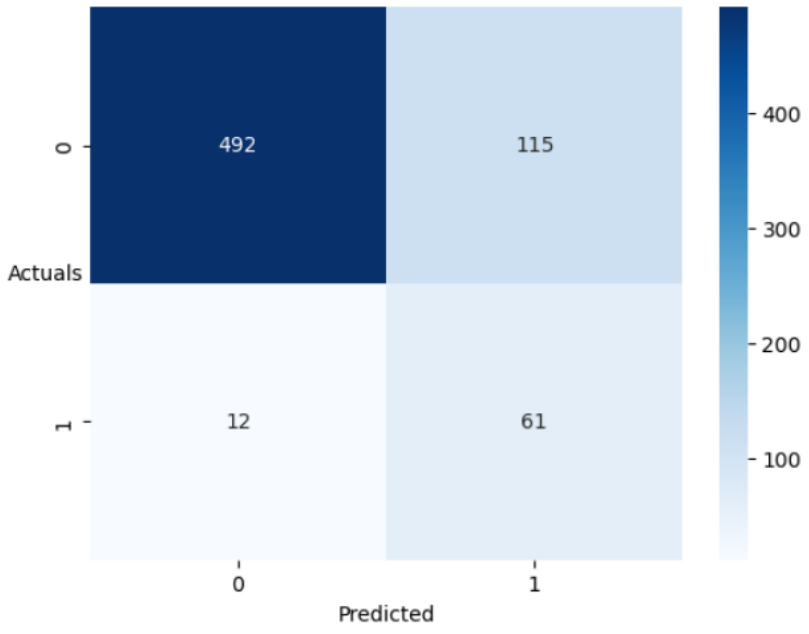


Figure 22: Confusion Matrix of Logistic regression test Model with optimal with threshold 0.0836

	precision	recall	f1-score	support
0.0	0.976	0.811	0.886	607
1.0	0.347	0.836	0.490	73
accuracy			0.813	680
macro avg	0.661	0.823	0.688	680
weighted avg	0.909	0.813	0.843	680

Table 14: Classification Matrix of Logistic regression Train Model with optimal with threshold 0.0836

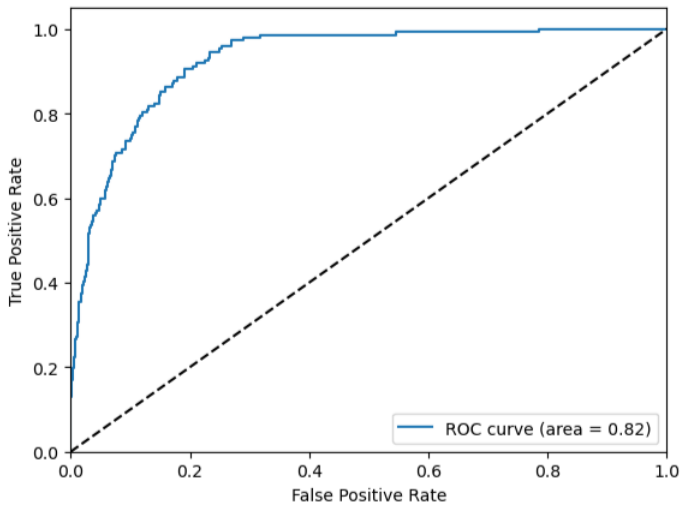


Figure 23 ROC curve of Logistic Regression Model on Test Data with optimal Threshold

Inference

The accuracy of the model is 0.813 at the optimal threshold

Recall score for test data has gone up to 0.836 for test set after decreasing the threshold from 0.5 to 0.0836.

But the precision has taken a hit, its reduced to 0.35 .

AUC score of this model is 0.82

After reducing the threshold from 0.5 to 0.0836, Logistic regression Model is able to predict defaulters with a recall score of .84 meaning approx. 84 % of the time , Model can predict the defaulters correctly which is better than the previous model with a threshold of 0.5.

But given the fact that Dataset has only 11% of defaulters , it's a pretty good model . Precision score is also very less with this model meaning , model is not very good at catching false positives .

Overall, it's a good model with no over-fitting or under fitting .

Interpretation

- For our model, Accuracy is 0.813. Accuracy is the ratio of the total number of correct predictions and the total number of predictions which is to say of all the default and non-default predictions made, model is correct 81% of the time . This is a decent score considering the class imbalance in the data .
- Our main agenda is to identify potential defaulters .Although we do aim for high precision and high recall value, achieving both at the same time is not possible. For example, if we change the model to one giving us a high recall, we might detect all the companies who will default, but we might end up classifying non defaulters as defaulters .Similarly, suppose we aim for high precision to avoid classifying non-defaulters as defaulters. In that case, we end up not identifying companies who actually will default.
- With a recall of 0.84 for class 1 (default cases) and 0.81 for class 0 (non-default cases), the model demonstrates reasonably good performance in identifying both default and non-default instances
- The recall of 0.84 for class 1 indicates that the model captures an 84% of the actual default cases. It effectively identifies companies that are at risk of defaulting on their debt obligations. This is critical to risk assessment since it identifies businesses that could need more oversight or steps to reduce the likelihood of financial instability.
- The recall of 0.81 for class 0 suggests that the model accurately identifies non-default cases 81 % of the time . It correctly classifies a significant proportion of companies that are not at risk of defaulting. This is helpful in lowering false positives and guaranteeing that businesses with solid financial stability are classified correctly.
- The Precision score of models is 0.35 for class 1 , meaning when it predicts a default , it is correct around 35% of the time. This is not a very good score .But this is the trade-off that one has to make to achieve higher recall score .
- From ROC curve we get a value of 0.82 as the AUC, which is a pretty good score! In simplest terms, this means that the model can distinguish the companies that default and those who don’t 82% of the time. .
- A balanced performance in capturing both default and non-default cases is indicated by the comparatively good recall for both class 0 and class 1. Maintaining an accurate risk assessment procedure and making defensible judgments based on the model's predictions depend on this.
- The model's ability to discriminate between default (class 1) and non-default cases (class 0) can be very useful when making investment decisions. It aids in the management of potential financial exposure, the determination of creditworthiness, and the analysis of the risk associated in investing in enterprises.

Let's see the important features used in model building

Feature importance:

Feature	Coefficients
Research_and_development_expense_rate	0.4492
_Interest_bearing_debt_interest_rate	0.3042
_Continuous_Net_Profit_Growth_Rate	-0.3630
_Quick_Ratio	-1.0579
_Total_debt_to_Total_net_worth	0.2227
_Accounts_Receivable_Turnover	-0.7586
_Allocation_rate_per_person	0.5777
_Retained_Earnings_to_Total_Assets	-1.1189
_Total_expense_to_Assets	0.3858
_Cash_Turnover_Rate	-0.3579
_Equity_to_Liability	-1.0412

Table 15 Feature Importance

From the model summary we have extracted some of the important features and their coefficients .

Here's what a Logistic Regression model equation looks like

$\text{logit}(p) = a + bX_1 + cX_2$ and so on ……

logit(p) is just a shortcut for log(p/1-p), where p = P{Y = 1}, i.e. the probability of "default", or the presence of an outcome. X_1 and X_2 are the predictor variables like Quick_Ratio and Alocation_rate_per_person, and b and c are their corresponding coefficients, each of which determines the emphasis X_1 and X_2 have on the final outcome Y (or p). Last, a is simply the intercept.

A 1 unit increase in Quick_Ratio will result in 1.0579 decrease in logit(p) assuming all other variables remain same and a 1 unit increase in Alocation_rate_per_person will result in 0.5777 increase in logit(p) indicating that increase in Quick Ratio bring down the log odds of default whereas increase in Alocation_rate_per_person increases log odds of default . This way we can interpret the log odds of default using the features and their coefficients.

Build a Random Forest Model on Train Dataset. Also showcase your model building approach

One can increase the model performance using hyperparameters. Finding the optimal hyperparameters would help us achieve the best-performing model

Hyperparameters for a model can be chosen using several techniques such as Random Search, Grid Search, Manual Search, Bayesian Optimizations, etc. For our model building we will use the GridSearchCV which uses the Grid Search technique for finding the optimal hyperparameters to increase the model performance. Hyperparameters are the variables that the user specifies while

building the Machine Learning model. Hyperparameters are used to evaluate optimal parameters of the model.

Grid Search uses a different combination of all the specified hyperparameters and their values and calculates the performance for each combination and selects the best value for the hyperparameters..

In GridSearchCV, along with Grid Search, cross-validation is also performed. Cross-Validation is used while training the model. In cross-validation, the process divides the train data further into two parts – the train data and the validation data.

The most popular type of Cross-validation is K-fold Cross-Validation. It is an iterative process that divides the train data into k partitions. Each iteration keeps one partition for testing and the remaining k-1 partitions for training the model. The next iteration will set the next partition as test data and the remaining k-1 as train data and so on. In each iteration, it will record the performance of the model and at the end give the average of all the performance. Thus, GridSearch along with cross-validation is used evaluate the best hyperparameters.

GridSearchCv(estimator,param_grid,cv,scoring) primarily takes 4 arguments
i.e., estimator, param_grid, cv, and scoring. The description of the arguments is as follows:

- estimator – A scikit-learn model
- param_grid – A dictionary with parameter names as keys and lists of parameter values.
- scoring – The performance measure. For example, 'r2' for regression models, 'precision' for classification models.
- cv – An integer that is the number of folds for K-fold cross-validation.

Steps in Random Forest Ensemble model Building

- We define param grid included hyperparameters like max_depth, min_samples_leaf, min_samples_split, and n_estimators with following values

  'max_depth':[3,5,7],

  'min_samples_leaf':[5,10,15],

  'min_samples_split':[15,30,45],

  'n_estimators':[25,50]

- We create an instance of the Random Forest classifier. This will serve as our base model for the grid search. We perform the grid search using the GridSearchCV and pass in the Random Forest classifier as the estimator and the hyperparameter grid we defined earlier.
- We then fit our model to the GridSearchCV by passing the train data with our independent variables and dependent variable.
- Once the grid search is completed, we get the best combination of hyperparameters that yields the highest performance. This is the best model with the optimal hyperparameters from the grid search. This model is then used to make predictions on Test data.

- Using GridSearchCV the best parameters to build a random forest model is as below

```
{'max_depth': 7,
 'min_samples_leaf': 15,
 'min_samples_split': 45,
 'n_estimators': 50}
```

Let's see the model performance metrics on the training set

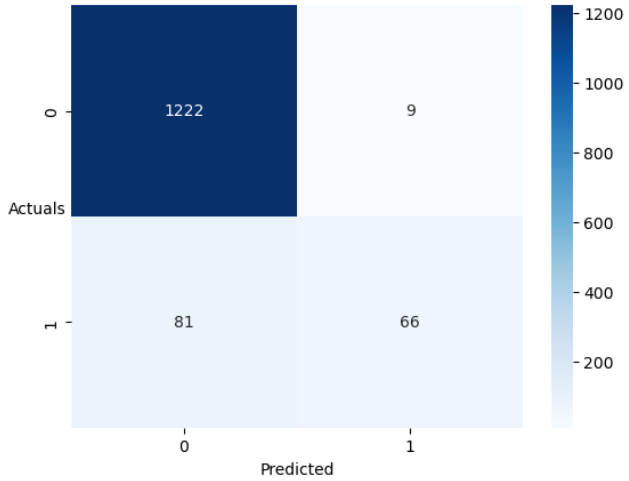


Figure 24: Confusion Matrix of Random Forest Model for Train Dataset

	precision	recall	f1-score	support
0.0	0.93	0.99	0.96	1231
1.0	0.86	0.39	0.54	147
accuracy			0.93	1378
macro avg	0.90	0.69	0.75	1378
weighted avg	0.92	0.93	0.92	1378

Table 16: Classification Matrix of Random Forest Model for Train Dataset

ROC curve of Train data with Random Forest Model

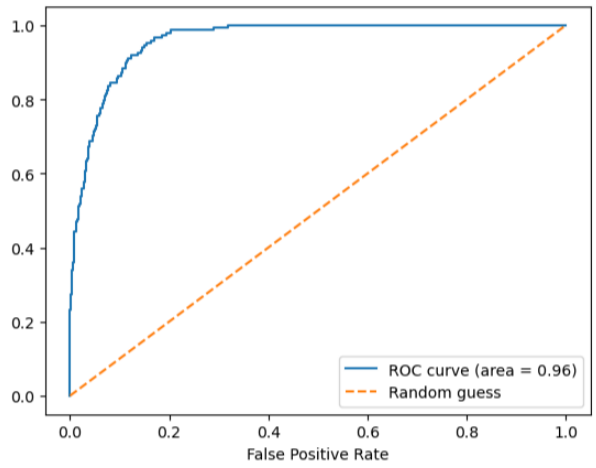


Figure 25:ROC curve for Random Forest Model Train Dataset.

The Accuracy on Train data is .93 .

The Recall score for Class 0 is .99 for Train data and the precision class 0 is .93

The Recall score for Class 1 is .39 for Train data and the precision class 1 is .86

AUC score is .96

Because of the Class Imbalance recall scores of Classes 0 is 0.99 whereas that of Class 1 is 0.39

Based on the recall score of 0.39 for Class1 , the model's performance can be considered moderate. A higher recall score indicates a stronger ability to identify default cases accurately. As a result, the model did not perform well. Therefore ,Random Forest may not be the most appropriate model for identifying defaulters .

Validate the Random Forest Model on test Dataset and state the performance metrics. Also state interpretation from the model

Let's see the model performance metrics on the training set

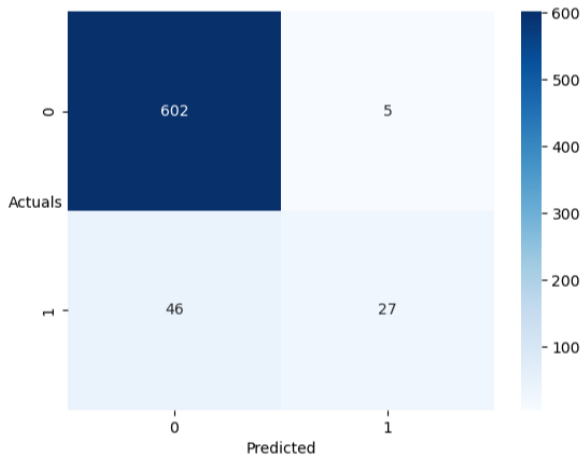


Figure 26: Confusion Matrix of Random Forest Model for Test Dataset

	precision	recall	f1-score	support
0.0	0.93	0.99	0.96	607
1.0	0.84	0.37	0.51	73
accuracy			0.93	680
macro avg	0.89	0.68	0.74	680
weighted avg	0.92	0.93	0.91	680

Figure 27: Classification Matrix of Random Forest Model for test Dataset

The Accuracy on Test data is .93

The Recall score for Class 0 is .99 for Test data and the precision done class 0 is .93 .

The Recall score for Class 1 is .37 for Test data and the precision done class 1 is .84 .

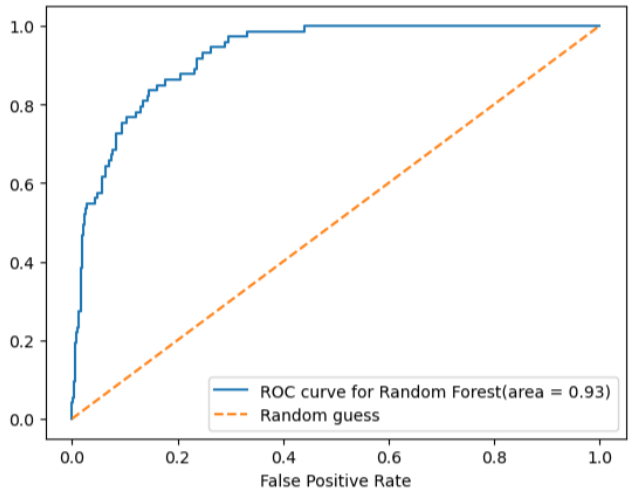


Figure 28: ROC curve for Random Forest Model Test Dataset.

Inferences

- The Accuracy on test data is .93 on test data and .93 on train data .
- The Recall score for Class 1 is .37 for Test data and .39 for Train data.
- The precision core on test data for class 1 is .84 and .86 on train data

Interpretation:

- For our model, Accuracy is 0.93. Accuracy is the ratio of the total number of correct predictions and the total number of predictions which is to say of all the default and non-default predictions made, model is correct 93% of the time . This is a very good score
- The model's performance has based on the recall of 0.37 on the test data. The model accurately identifies only 37% of genuine default situations in the test data and may be missing a large number of defaulting companies, which is undesirable for predicting defaults. As a result, the model's performance on the test data might be regarded as poor.
- The dataset has a relatively low number of default instances compared to non-default instances. There are only 11% of default cases. The significant difference in recall values between class 1 (default cases) and class 0 (non-default cases) is due to the class imbalance in the dataset and hence the model's ability to accurately identify non-default cases is much higher compared to default cases.
- The low recall of 0.37 for default cases implies that model has very high false negatives. The model is unable to correctly identify a considerable number of companies that actually default. This is a very severe problem since it might cost the bank a lot of money if they fail to identify the firms that would default.
- The Precision score of models is 0.84 for class 1 , meaning when it predicts a default , it is correct around 84% of the time. This is good score .But since its recall scores that we are most interested in , we need to tune the model further to improve Recall scores
- From ROC curve we get a value of 0.93 as the AUC, which is a pretty good score! In simplest terms, this means that the model can distinguish the companies that default and those who don't 93% of the time.

- To evaluate the risk involved in making investments in or extending loans to businesses, investors and financial institutions rely on precise recall prediction. The model may not be adequately reflecting the financial risk of default, as seen by the poor recall for default situations. This might result in less than ideal investment choices and heightened exposure to default risk.

Build a LDA Model on Train Dataset. Also showcase your model building approach

Steps in Building best LDA Model

- We define param grid with hyperparameters like solver, tol with following values

  'solver':['svd', 'lsqr', 'eigen'],

  'tol':[0.0001,0.00001]}

- We create an instance of the LinearDiscriminantAnalysis. This will serve as our base model for the grid search. We perform the grid search using the GridSearchCV and pass in the LinearDiscriminantAnalysis as the estimator and the hyperparameter grid we defined earlier.
- We then fit our model to the GridSearchCV by passing the train data with our independent variables and dependent variable.
- Once the grid search is completed, we get the best combination of hyperparameters that yields the highest performance. This is the best model with the optimal hyperparameters from the grid search. This model is then used to make predictions on Test data.

Using Grid search CV we get the the best parmeters for building LDA model as below .

```
{'solver': 'svd', 'tol': 0.0001}
```

- We predict the Default parameter using a cut-off of 0.5. This means that if the predicted probability is above 0.5, then the default value will be 1, if the value is less than 0.5, the default value will be 0.

We build the LDA model using above params and see the performance on train data.

Let see the confusion matrix with LDA Model Predictions using 0.5 as the cut-off on train data:

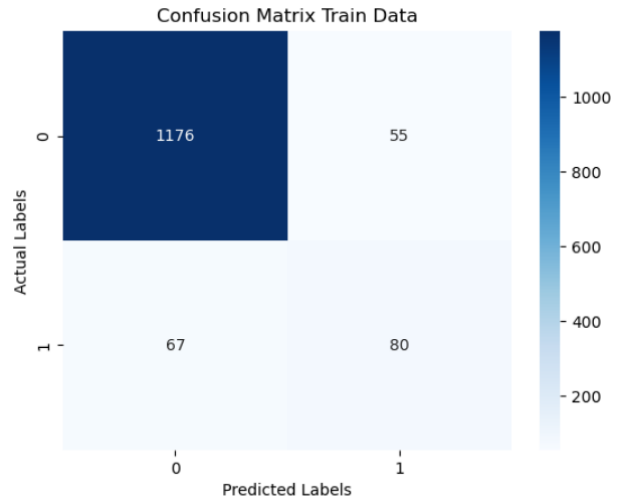


Table 17: Confusion Matrix for LDA Model on TrainSet

	precision	recall	f1-score	support
0.0	0.95	0.96	0.95	1231
1.0	0.59	0.54	0.57	147
accuracy			0.91	1378
macro avg	0.77	0.75	0.76	1378
weighted avg	0.91	0.91	0.91	1378

Table 18: Classification Matrix for LDA Model on TrainSet

The Accuracy on Train data is .91

The Recall score for Class 0 is .96 for Train data and Precision score for class 0 on train data is 0.95

The Recall score for Class 1 is .54 for Train data and Precision score for class 1 on train data is 0.59

Let see the confusion matrix with LDA Model Predictions using 0.5 as the cut-off on test data

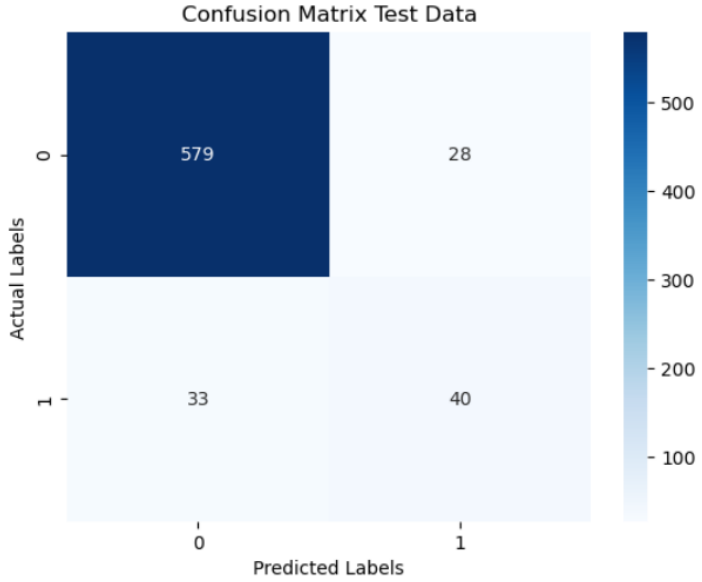


Figure 29 Confusion Matrix of LDA Model on Test Dataset

	precision	recall	f1-score	support
0.0	0.95	0.95	0.95	607
1.0	0.59	0.55	0.57	73
accuracy			0.91	680
macro avg	0.77	0.75	0.76	680
weighted avg	0.91	0.91	0.91	680

Table 19 Classification Matrix for LDA Model on Test Data

With threshold 0.5, the Accuracy on Test data is .91 .

The Recall score for Class 0 is .95 for Test data and the precision done class 0 is .95.

The Recall score for Class 1 is .55 for Test data and the precision done class 1 is .59

We are attempting to predict whether or not a company will default based on various personal data. We want our estimator to prioritize the true positive rate over the false positive rate. That is to say, correctly identifying an imminent default is more important than predicting a default that fails to materialize. It is clear from the above figures that the model did not perform very well in predicting the default parameter as indicated by low recall scores of 0.55 and we are interested to improve Class 1 recall scores. Hence, we need to change the cut-off value.

Let's the ROC curves for Train and Test data

AUC for Training Data: 0.916

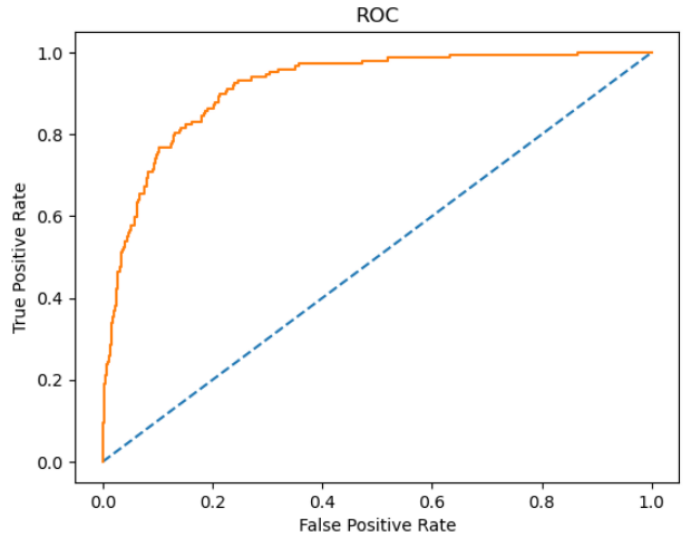


Figure 30:ROC curve of Train data LDA Model

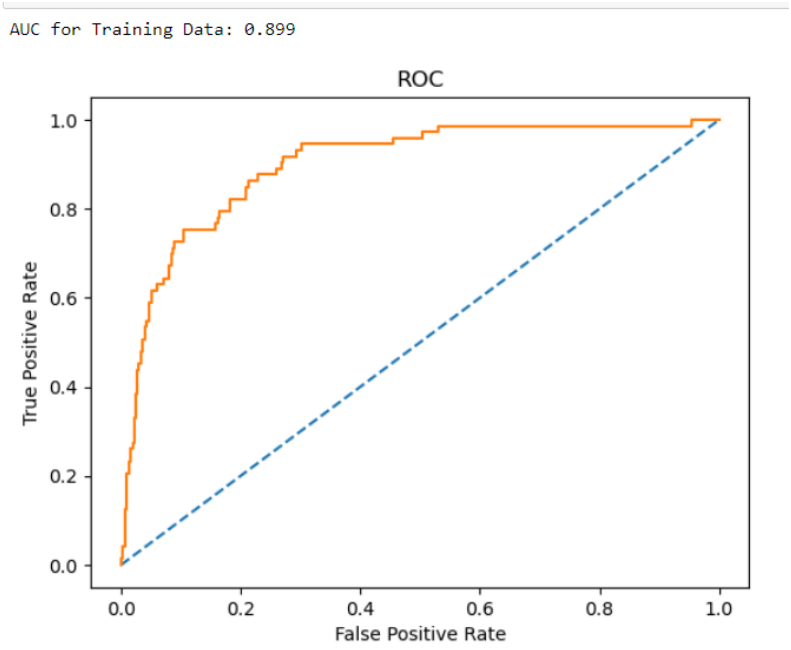


Figure 31 ROC curve of Test data LDA Model

From ROC-AUC curve we have to get the threshold which will have maximum True Positive Rate and lowest False Positive Rate.

To calculate the optimum cut-off, the steps are as below:

• We calculate the ROC curve using the true labels (the actual class values) and the predicted probabilities (the probabilities assigned to the positive class) for the training data.

• Using roc_curve function, we get three arrays: fpr, tpr, and thresholds. The fpr array contains the false positive rates at different classification thresholds, and the tpr array contains the corresponding true positive rates. The threshold array holds the actual classification thresholds.

• To find the optimal threshold, which maximizes the difference between the true positive rate and the false positive rate, we use the np.argmax function. This function helps us find the index of the maximum value in the expression tpr - fpr. By retrieving this index, we can determine the optimal threshold value.

• Finally, we extract the optimal threshold by indexing the threshold array with the index obtained from np. argmax. This value represents the threshold that strikes a balance between correctly classifying positive samples and avoiding false positives.

•The optimal threshold calculated is 0.0399

Let see the confusion matrix with LDA Model Predictions using 0.0399 as the cut-off on train data:

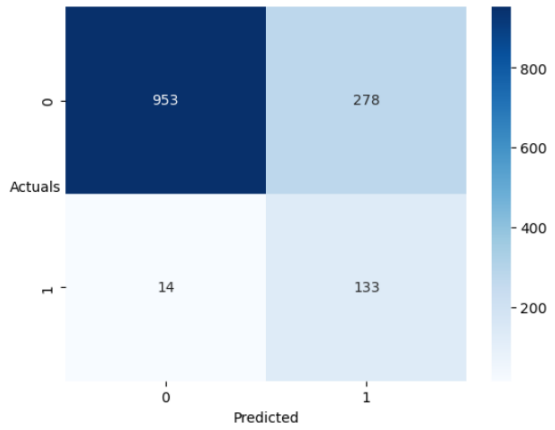


Table 20 Confusion Matrix of LDA Model with new threshold on Train Dataset

	precision	recall	f1-score	support
0.0	0.986	0.774	0.867	1231
1.0	0.324	0.905	0.477	147
accuracy			0.788	1378
macro avg	0.655	0.839	0.672	1378
weighted avg	0.915	0.788	0.826	1378

Table 21 Classification Matrix of LDA Model with optimal threshold on Train Dataset

Let's the ROC curves for Train data with optimal threshold

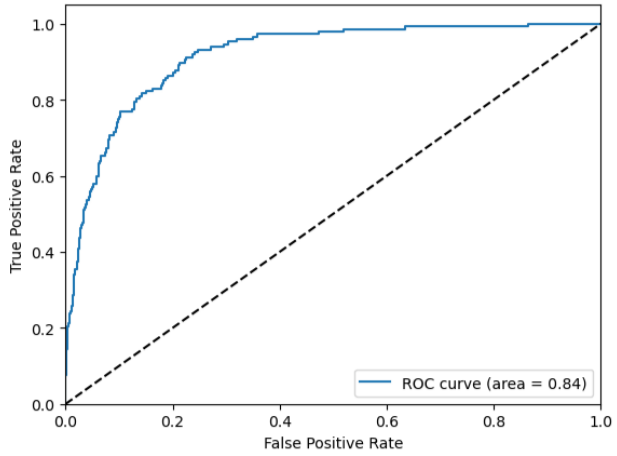


Figure 32 ROC curve of Train data LDA Model with new threshold

With threshold 0.0399, the Accuracy on Train data is 0.79 which is to say of all the default and non-default predictions made, model is correct 79% of the time

The Recall score for Class 0 is 0.774 and the precision done class 0 is 0.986 for Train data

The Recall score for Class 1 is 0.905 and the precision done class 1 is 0.324 for Train data

In summary, the model demonstrates a strong performance with a recall of 0.91 for class 1 (default cases). This high recall indicates that the model is effective in correctly identifying companies that are at risk of defaulting on their debt obligations.

From ROC curve we get a value of 0.84 as the AUC, which is a pretty good score! In simplest terms, this means that the model can distinguish the companies that default and those who don't 84% of the time.

Validate the LDA Model on test Dataset and state the performance metrics. Also state interpretation from the model

Let see confusion matrix at this threshold on test data at optimal threshold is `0.0399`

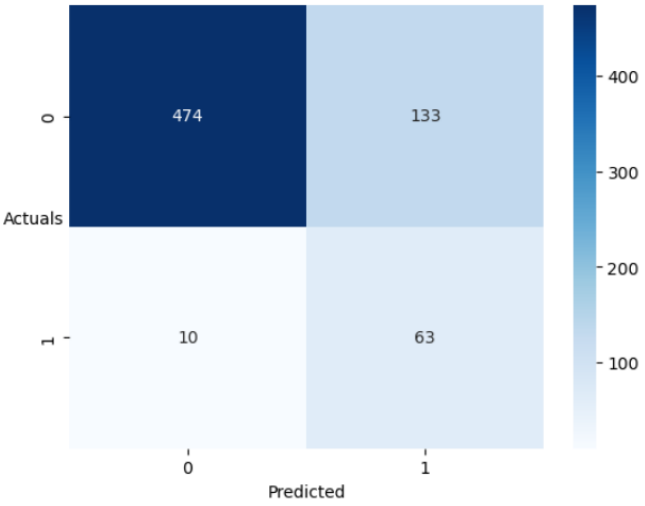


Figure 33: Confusion Matrix for LDA Model on test set after changing threshold

	precision	recall	f1-score	support
0.0	0.979	0.781	0.869	607
1.0	0.321	0.863	0.468	73
accuracy			0.790	680
macro avg	0.650	0.822	0.669	680
weighted avg	0.909	0.790	0.826	680

Table 22: Classification Matrix for LDA Model on test set after changing threshold

Let's the ROC curves for test data with optimal threshold

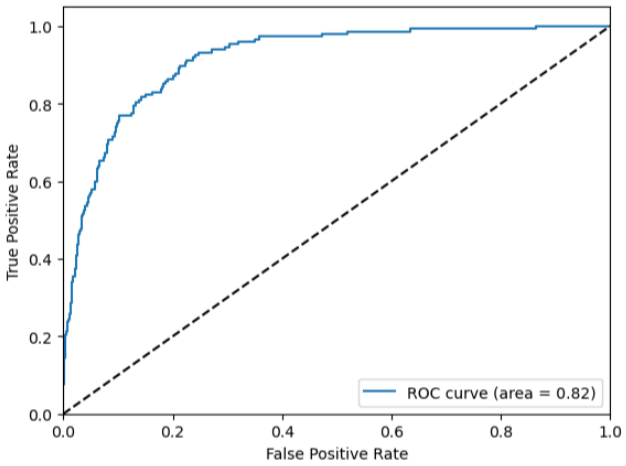


Figure 34 ROC curve of test data LDA Model with new threshold

Inference

- The LDA model looks a bit better than the Logistic Regression models in terms of the Recall value for Train and test data. However, the Accuracy for the test data is less compared to that of Logistic Regression Model. The AUC and ROC curves also do not show a significant difference compared to the other models built.
- With threshold 0.0399, the Accuracy on test data is .79 which was .93 with threshold 0.5 .
- The Recall score for Class 1 is .863 for Test data which was .55 with threshold 0.5.The Recall scores have improved a lot after changing the threshold . It means this model is good at predicting defaults 86.3% of the time .
- However, the precision scores for Class 1 have dropped from .59 to .32.
- Overall LDA is great model with good recall scores and no overfitting or underfitting .

Interpretation

- For our model, Accuracy is 0.79. Accuracy is the ratio of the total number of correct predictions and the total number of predictions which is to say of all the default and non-default predictions made, model is correct 79% of the time . This is a decent score considering the class imbalance in the data .
- Our main agenda is to identify potential defaulters .Although we do aim for high precision and high recall value, achieving both at the same time is not possible. For example, if we change the model to one giving us a high recall, we might detect all the companies who will default, but we might end up classifying non defaulters as defaulters .Similarly, suppose we aim for high precision to avoid classifying non-defaulters as defaulters. In that case, we end up not identifying companies who actually will default.
- With a recall of 0.86 for class 1 (default cases) and 0.78 for class 0 (non-default cases), the model demonstrates reasonably good performance in identifying both default and non-default instances
- The recall of 0.86 for class 1 indicates that the model captures an 86% of the actual default cases. It effectively identifies companies that are at risk of defaulting on their debt

obligations. This is critical to risk assessment since it identifies businesses that could need more oversight or steps to reduce the likelihood of financial instability.

- The recall of 0.78 for class 0 suggests that the model accurately identifies non-default cases 78 % of the time . It correctly classifies a significant proportion of companies that are not at risk of defaulting. This is helpful in lowering false positives and guaranteeing that businesses with solid financial stability are classified correctly.
- The Precision score of models is 0.321 for class 1 , meaning when it predicts a default , it is correct around 32.1% of the time. This is not a very good score .But this is the trade-off that one has to make to achieve higher recall score .
- From ROC curve we get a value of 0.82 as the AUC, which is a pretty good score! In simplest terms, this means that the model can distinguish the companies that default and those who don’t 82% of the time. .
- A balanced performance in capturing both default and non-default cases is indicated by the comparatively good recall for both class 0 and class 1. Maintaining an accurate risk assessment procedure and making defensible judgments based on the model's predictions depend on this.
- When making investment decisions, the model's capacity to distinguish between default circumstances (class 1) and non-default cases (class 0) can be quite helpful. It helps in managing possible financial exposure, determining creditworthiness, and analysing the risk involved in investing in businesses.

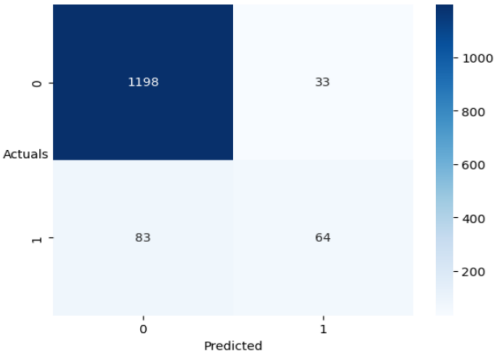
Compare the performances of Logistic Regression, Random Forest, and LDA models (include ROC curve)

Model Evaluation and Performance

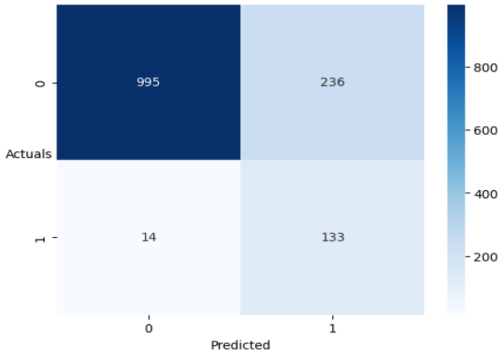
To check performance of Predictions of every model built on Train and Test datasets, Accuracy score is calculated. A Confusion Matrix, ROC curve and AUC-ROC score has been devised as well. We have considered the 'Default' i.e., both 0, 1 as the interest classes. Therefore, we will also look at the Accuracy scores of all models.

Comparing Confusion matrix of all models (Train data)

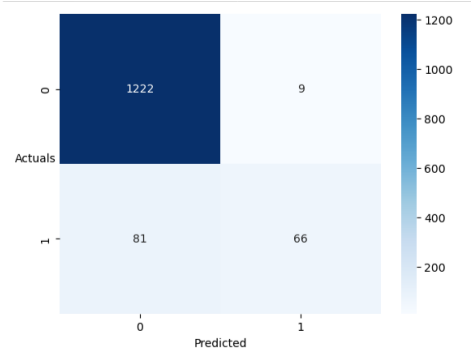
Logistic Regression Model



Tuned Logistic Regression Model

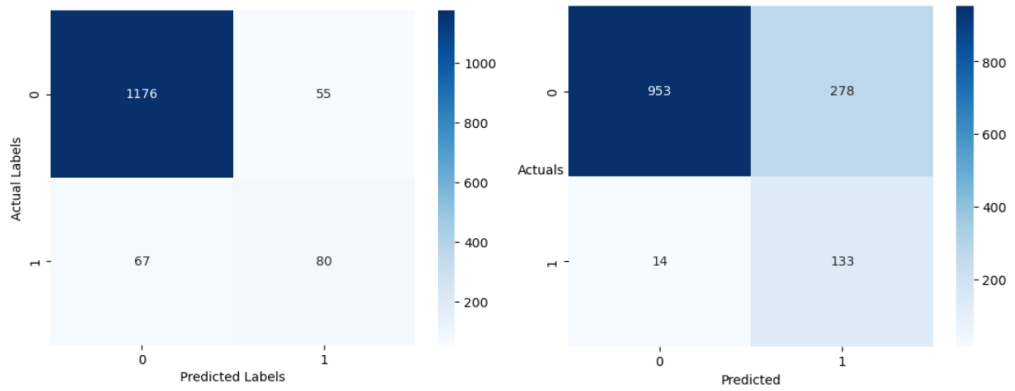


Random Forest Model



LDA Model

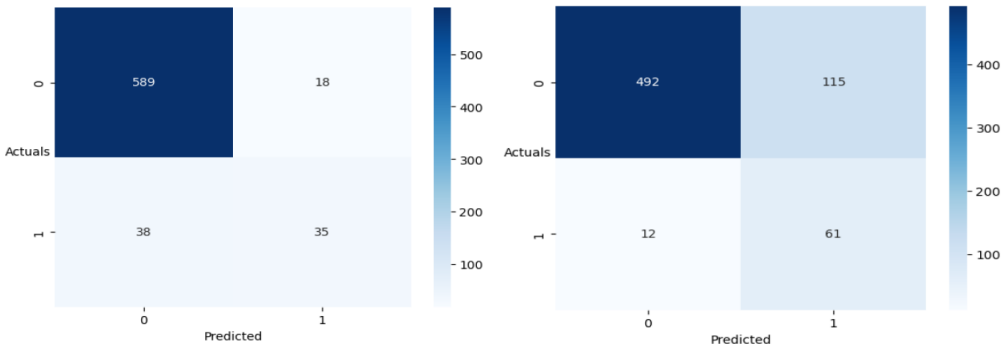
Tuned LDA Forest Model



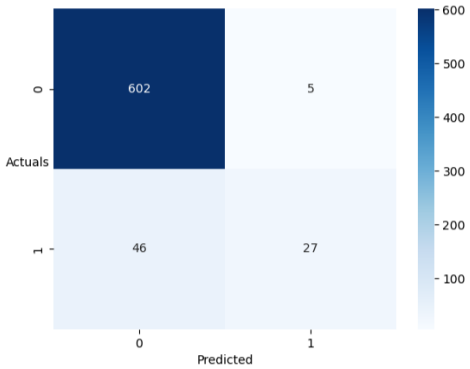
Comparing Confusion matrix of all models (Test data)

Logistic Regression Model

Tuned Logistic Regression Model

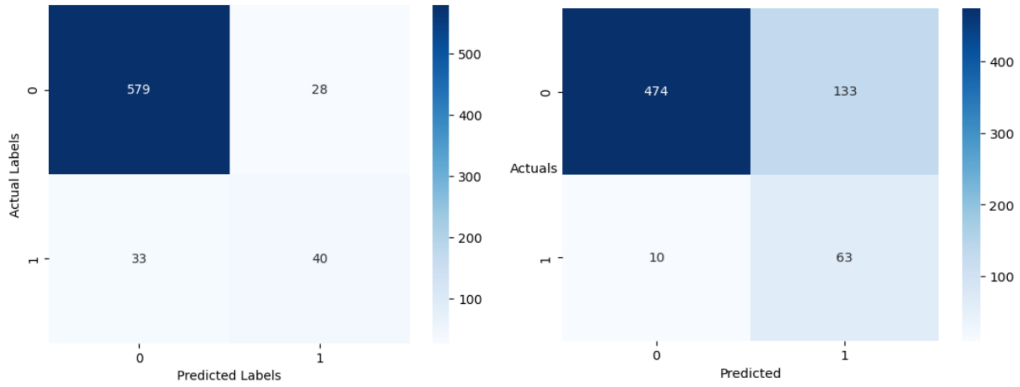


Random Forest Model

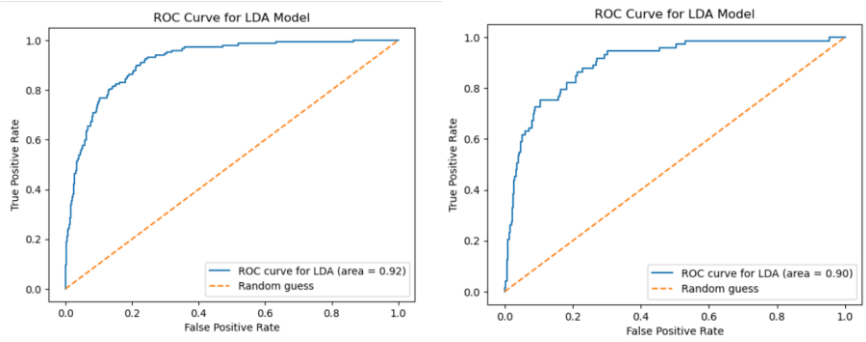


LDA Model

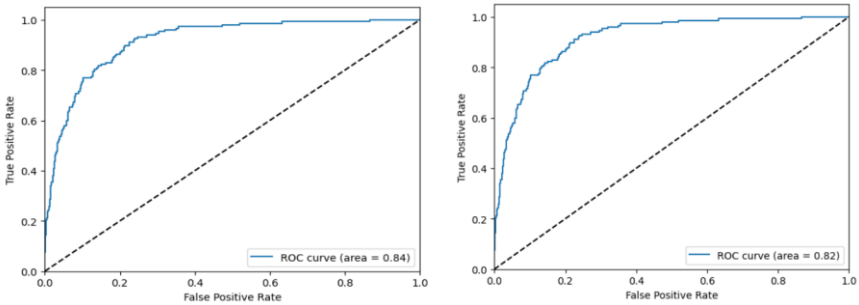
Tuned LDA Forest Model



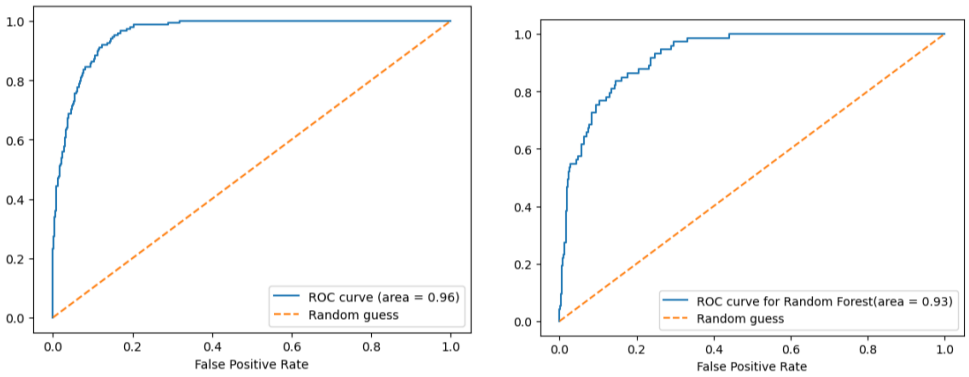
ROC curve of LDA Model Training and Test Data with default threshold



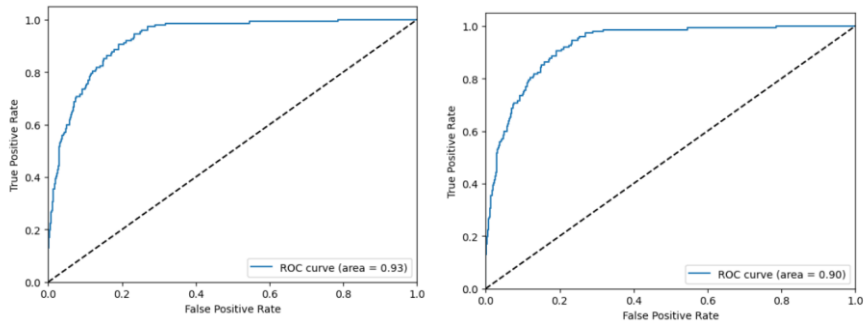
ROC curve of LDA Model Training and Test Data with optimal threshold



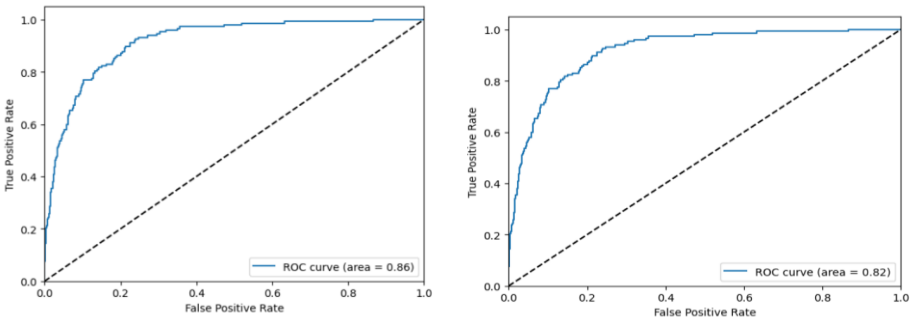
ROC curve of Random Forest Model Training and Test Data with default threshold



ROC curve of Logistic Regression Model on Training and Test Data with default threshold



ROC curve of Logistic Regression Model on Training and Test Data with optimal threshold



Summary of the perfomances of all Models

Model	Train Accuracy	Test Accuracy	Train Recall	Test Recall	Train Precision	Test Precision	Train ROC	Test ROC
Logistic Regression	0.92	0.92	0.44	0.48	0.66	0.66	0.93	0.90
Logistic with new threshold	0.82	0.81	0.91	0.84	0.36	0.35	0.86	0.82
RF Model	0.93	0.93	0.39	0.37	0.86	0.84	0.96	0.93
LDA	0.91	0.91	0.54	0.55	0.59	0.59	0.92	0.90
LDA with new threshold	0.79	0.79	0.91	0.86	0.32	0.32	0.84	0.82

- **Accuracy**: When comparing the Accuracy , Random Forest Model has the highest accuracy of all the models with 93%
- **Recall**: When we consider Recall score on Test data Class 1 which is our interest class, we see that LDA model has performed the best with recalls score of 0.86
- **AUC**: We see the AUC score is highest with Random Forest Model with score being 0.93

on test data

- With Random Forest , Accuracy and AUC scores are very high , but recall scores are very low compared to other models and there is overfitting when it comes to Recall and Precision scores, meaning model performs well on train data and performs poorly on test data, so it's not a good model even though the accuracy and AUC scores are high compared to other models .
- With LDA model with optimal threshold, accuracy is 0.79, recall is 0.86 for class 1 and recall is 0.781 for class 0, precision for class 1 is 0.32 and precision for class 0 is 0.98 and AUC is 0.82 on test data
- With Logistic Regression(LR) model with optimal threshold, accuracy is 0.81, recall is 0.84 for class 1 and recall for class 0 is 81.1 , precision for class 1 is 0.35 and precision for class 0 is 0.98 and AUC is 0.82 on test data .
- Except for Recall scores which are slightly less than LDA , Logistic Regression model is having better scores compared to LDA model.
- Among the three models, Logistic Regression and LDA perform better in terms of recall for default cases (class 1). However, LDA has a higher recall for default cases (class 1) compared to Logistic Regression.
- The Random Forest model, despite achieving a high accuracy, precision and AUC, performs poorly in identifying default cases (class 1) with a low recall of 0.37. This indicates that the model may miss a significant number of default instances, which is not desirable in the context of predicting defaults.
- The AUC score measures the overall performance of the models in terms of their ability to distinguish between the two classes. Both Logistic Regression and LDA have relatively high AUC scores of 0.82 , indicating good discriminatory power.
- Considering the factors of recall and AUC score, the Logistic Regression model appears to be the best-performing model. It demonstrates a highest recall for class 0 and similar recall scores as LDA for class 1, indicating a balanced performance in capturing instances from both classes.
- Overall, Logistic Regression model is the top-performing model because it has a good overall performance across all metrics, which indicates strong discriminating power, and balanced recall for both classes.

Conclusions and Recommendations

Conclusions

- Logistic regression is simple to interpret and it doesn't require much computational resources. This makes it a good choice when the target variable is binary and the relationship between the predictors and the response is linear. The model's coefficients provide insights into the significance and direction of each feature's impact on the target variable.
- The relative significance of various features in forecasting credit risk can be ascertained from the values of the logistic regression coefficient. By analysing the coefficients, we can ascertain variables that are influencing credit risk, and then concentrate on keeping an even closer eye on and assessing those particular variables.
- Linear Discriminant analysis can also be used to determine which set of variables discriminate between two or more naturally occurring groups and to classify an observation into these known groups. In order to achieve that discriminant analysis is based on the estimation of the orthogonal discriminant functions, the linear combination of the standardized independent predictor variables gives the greatest means differences between the existing groups. Thus, it can be proposed that both discriminant analysis and logistic regression can be used to predict the probability of a specified outcome using all or a subset of available variables.
- As per the Logistic Regression model, Research_and_development_expense_rate,Interest_bearing_debt_interest_rate, Continuous_Net_Profit_Growth_Rate, Quick_Ratio, Accounts_Receivable_Turnover, Allocation_rate_per_person, Retained_Earnings_to_Total_Assets, Total_income_to_Total_expense,Total_expense_to_Assets, Cash_Turnover_Rate, Equity_to_Liability are very important in deciding whether the company will default or not.
- The LDA model indicates that Research_and_development_expense_rate, Cash_to_Current_Liability, Total_expense_to_Assets , Allocation_rate_per_person ,Cash_flow_rate, Retained_Earnings_to_Total_Assets , Quick_Ratio , Net_Value_Growth_Rate, Accounts_Receivable_Turnover, Equity_to_Liability ,Interest_bearing_debt_interest_rate are important are very important in deciding whether the company will default or not.
- Ensemble techniques like Random Forest though offer a high accuracy in predictions , they are very complex and not very stable and there are high chances of overfitting with RF model. On the other hand, Logistic Regression is generally less complex with fewer features and more stable. It allows for constant and dependable risk evaluations throughout a number of time periods, this stability is important for credit risk modelling.
- While Random Forest isn't best-performing model in terms of default case recall, It is more of Accuracy focused algorithm. Random selection in individual decision trees of RFC can capture more complex feature patterns to provide the best accuracy. It can capture complex nonlinear relationships and identify potential interactions among features. RF can also tell us how much each feature contributes to class

prediction using Feature Importance and tree graphs for better interpretation.

- Overall Random Forest Classifier performs better with more categorical data than numeric. If the dataset has more Categorical data and consists of outliers it is better to use Random Forest Classifier. But in our dataset we have mostly numeric data, hence Logistic Regression is a better model
- In conclusion, Logistic Regression is recommended as the key model for credit risk assessment because of its interpretability, stability, and feature importance. Incorporating extra insights from Random Forest ensemble models, can provide a more complete knowledge of credit risk characteristics.

Recommendations based on feature Importance

- Equity vs. Liability: Equity is the ownership position that shareholders have in a corporation, whereas liabilities are debts owed to others. A high E/L ratio indicates that the company is using more equity financing, which can be less risky but may limit the company's growth potential. On the other hand, a low E/L ratio indicates that the company is relying heavily on debt to finance its operations. This can lead to a higher risk of default and bankruptcy if the company is unable to meet its debt obligations.
- Research and Development Expense Rate: Research and development (R&D) expenses are associated directly with the research and development of a company's goods or services and any intellectual property generated in the process. A company generally incurs R&D expenses in the process of finding and creating new products or services. A higher research and development spending rate is connected with a higher chance of default, implying that organizations who invest a considerable amount of their revenue in R&D may experience financial difficulties. While R&D and innovation are crucial for growth, businesses must balance these investments with their capacity to satisfy debt obligations and earn adequate profits. If a greater R&D spending rate is connected with a higher risk of default, then increasing this rate would have a positive effect on the default. It implies that organizations that spend a larger portion of their revenue on research and development may be more likely to default.
- Average Collection Days: A longer average collection period is related with a higher likelihood of default, increasing the average collection days would have a positive impact on the default parameter. It suggests that organizations who take longer to collect payments from their consumers may encounter liquidity issues and are more prone to default.
- Retained Earnings to Total Assets: The retained earnings/total assets ratio shows the amount of retained earnings or losses in a company. If a company reports low retained earnings to total assets ratio, it means that it is financing its expenditure using borrowed funds rather than funds from its retained earnings. It increases the probability of a company going bankrupt. On the other hand, high retained earnings to total assets ratio shows that a company uses its retained earnings to fund capital expenditure. It shows that the company achieved profitability over the years, and it does not need to rely on borrowings.
- Total_debt_to_Total_net_worth is a ratio of a Total Liability/Equity Ratio . A higher debt to net worth ratio indicates that the company has taken on more debt relative to its equity, which can increase the risk of default if the company experiences financial difficulties. Conversely, a lower the debt to net worth ratio suggests a lower financial risk and a more conservative financing strategy.

- Average Collection Days or Days Receivable Outstanding . The average collection period is the average number of days it takes a business to collect and convert its accounts receivable into cash. Collecting its receivables in a relatively short and reasonable period of time gives the company time to pay off its obligations .A shorter average collection period is generally preferable and means a business has higher liquidity and lesser chances of defaulting .
- Quick Ratio or Acid-test is a measure of a company's liquidity, Acid-test ratio (also known as quick ratio) is a measure of a company's liquidity, which is its ability to pay its short-term obligations using only its most liquid assets. It is calculated by dividing the sum of a company's cash, cash equivalents and marketable securities by its total current liabilities. The higher the acid-test ratio, the better. If the acid-test ratio is below 1, it means that a company does not have enough liquid assets (cash or equivalents) to cover its current liabilities. This suggests that the company is not in a financially sound position, as it cannot pay its short-term debts.
- Accounts_Receivable_Turnover for companies that are not defaulting is much higher than that for companies that are defaulting indicating that companies that default have are not managing the credit that they extend to their customers by evaluating how long it takes to collect the outstanding debt throughout the accounting period. A high accounts receivable turnover ratio can indicate that the company is conservative about extending credit to customers and is efficient or aggressive with its collection practices. It can also mean the company's customers are of high quality, and/or it runs on a cash basis
- Interest_bearing_debt_interest_rate measure is helpful in understanding the overall rate being paid by a company to use debt financing. The measure can also give banks an idea of the company's risk level compared to others because riskier companies generally have a higher Interest_bearing_debt_interest_rate . The degree of the Interest_bearing_debt_interest_rate depends entirely on the borrower's creditworthiness, so higher costs mean the borrower is considered risky and has a potential to default.