Finance & Risk Analytics
Project Report

Contents

PART-A **Page**

PART-B

List of Figures

List of Tables

FRA PROJECT

Problem: PART-A

Businesses or companies can fall prey to default if they are not able to keep up their debt obligations. Defaults will lead to a lower credit rating for the company which in turn reduces its chances of getting credit in the future and may have to pay higher interest on existing debts as well as any new obligations. From an investor's point of view, he would want to invest in a company if it is capable of handling its financial obligations, can grow quickly, and is able to manage the growth scale.

A balance sheet is a financial statement of a company that provides a snapshot of what a company owns, owes, and the amount invested by the shareholders. Thus, it is an important tool that helps evaluate the performance of a business.

Data that is available includes information from the financial statement of the companies for the previous year.

1. Outlier Treatment

In order to treat the outlier, we need to first create the outlier identification.

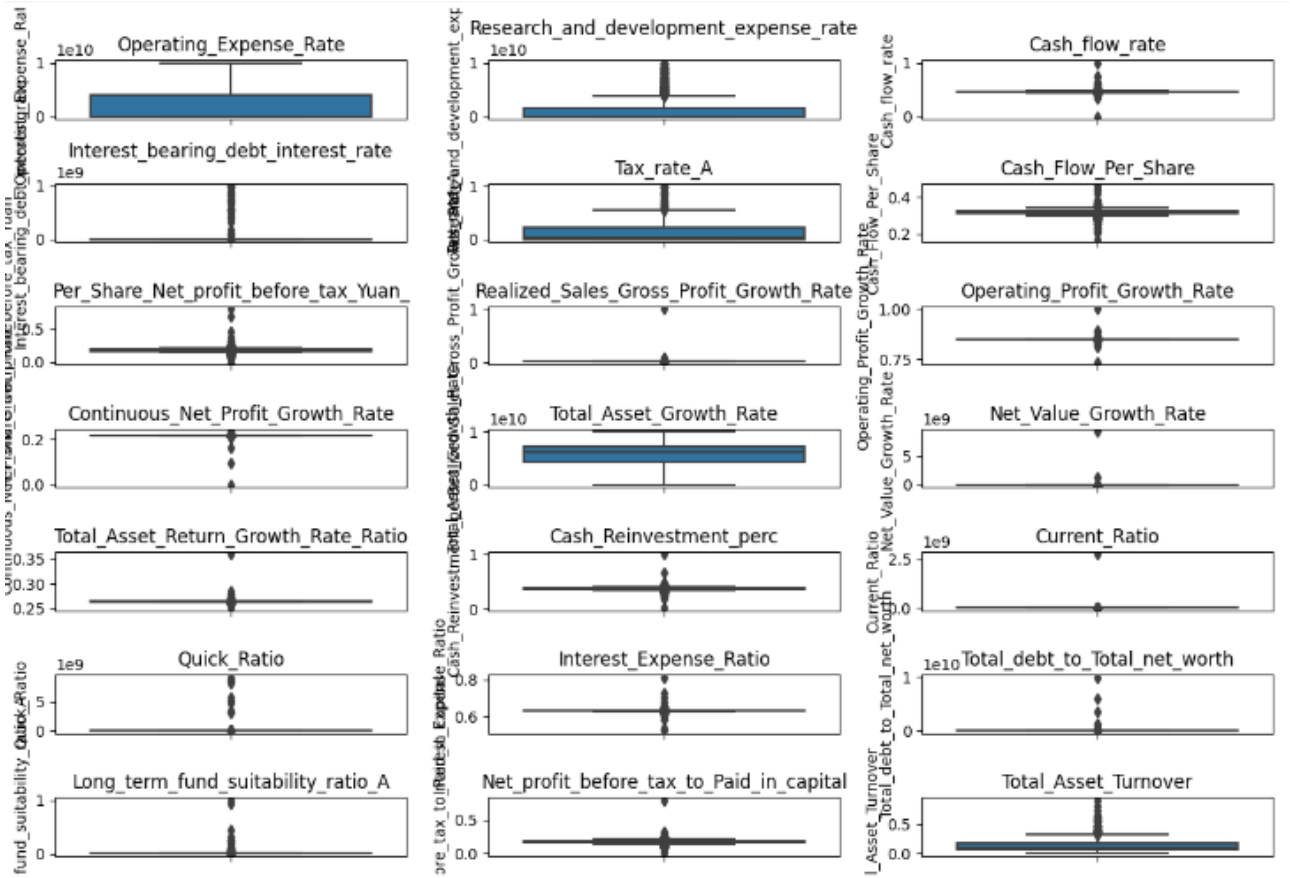

Figure 1: Box Plot of Numerical Data

We can see that, most the columns are having outliers. Only below columns aren't having outliers

- Fixed_Assets_to_Assets
- Liability_Assets_Flag
- Quick_Assets_to_Total_Assets
- Total_Asset_Growth_Rate
- Cash_Turnover_Rate
- Quick_Asset_Turnover_Rate
- Net_Income_Flag
- Operating_Expense_Rate

After Outlier Treatment:

```
0.000111127 0.00015787275 4110000000.0 8971499999.999998 <built-in function min> <built-in function max>
0.0 0.0 1550000000.0 6235999999.999994 <built-in function min> <built-in function max>
0.45200791775 0.460099142 0.46806908049999996 0.48237647095 <built-in function min> <built-in function max>
0.0 0.000276028 0.000663066 0.0010774079999999998 <built-in function min> <built-in function max>
0.0 0.0 0.21619090975 0.3416236463499999 <built-in function min> <built-in function max>
0.2999522528 0.315705242 0.325387282 0.33788816574999997 <built-in function min> <built-in function max>
0.14042328805 0.166603901 0.185885366 0.21451758669999998 <built-in function min> <built-in function max>
0.0219479388 0.022058314000000002 0.022151999 0.022579723299999997 <built-in function min> <built-in function max>
0.84770057975 0.84797396475 0.848114747 0.8485231827 <built-in function min> <built-in function max>
0.21737961015 0.217574132 0.21761982075 0.21774071545 <built-in function min> <built-in function max>
0.0001133856 4315000000.0 7220000000.0 8980000000.0 <built-in function min> <built-in function max>
0.00036385015 0.00043628325 0.00048837575 0.0006664002999999998 <built-in function min> <built-in function max>
0.2628939083 0.2637383345 0.26430966375 0.2653374748000003 <built-in function min> <built-in function max>
0.341920942 0.37072948099999997 0.38555754275 0.40518423324999997 <built-in function min> <built-in function max>
0.0034248259000000006 0.006567062 0.01350541775 0.034290064949999956 <built-in function min> <built-in function max>
0.0005525975 0.00294639875 0.00890298325 0.026455419749999994 <built-in function min> <built-in function max>
0.62819612065 0.630611567 0.63174365775 0.6348716631 <built-in function min> <built-in function max>
0.0012587656500000002 0.003939029 0.012987306 0.027580681649999994 <built-in function min> <built-in function max>
0.004971702 0.0051620305 0.00641530075 0.012746516849999993 <built-in function min> <built-in function max>
0.143312722 0.16586230275 0.18444498925 0.21076238605 <built-in function min> <built-in function max>
0.02226386835000001 0.061469265 0.167916042 0.326836582 <built-in function min> <built-in function max>
0.00046838365 0.000744626 0.001854463 0.011880129349999945 <built-in function min> <built-in function max>
0.00070908575 0.0035763845 0.008638997 0.014823774849999995 <built-in function min> <built-in function max>
0.00011044 0.00019092974999999999 3815000000.0 8712999999.999998 <built-in function min> <built-in function max>
0.00011877 0.00022789499999999999 0.008423224 8020000000.0 <built-in function min> <built-in function max>
0.012903226 0.020483871 0.044354839 0.09857258049999998 <built-in function min> <built-in function max>
0.37294271975000004 0.391386445 0.40089267525 0.46110219129999996 <built-in function min> <built-in function max>
0.0009401182500000001 0.00467161225 0.024574907 0.09450514144999984 <built-in function min> <built-in function max>
0.0569859891 0.17348271 0.4845435295 0.7437303149499999 <built-in function min> <built-in function max>
0.00550473845 0.021660374750000003 0.094028692 0.2709194153499999 <built-in function min> <built-in function max>
0.0008048039000000001 0.0036163042499999997 0.009608533499999999 0.027124043649999995 <built-in function min> <built-in funct
n max>
0.0002871999000000006 0.0010854764999999999 0.0075405355 0.04501159264999997 <built-in function min> <built-in function max>
0.31785974699999997 0.33770317675 0.354140153 0.38570003445 <built-in function min> <built-in function max>
0.2763681878 0.27700928225 0.2777110665 0.27972695015 <built-in function min> <built-in function max>
0.0 0.0028908425 0.01275115525 0.03137403239999999 <built-in function min> <built-in function max>
0.0 0.0 0.0104968395 0.0603855664999999 <built-in function min> <built-in function max>
0.8925789974 0.9278867909999999 0.9409370779999999 0.95276040155 <built-in function min> <built-in function max>
0.0019754772000000003 0.00218696425 0.00243314575 0.0028483944 <built-in function min> <built-in function max>
0.0057252165499999995 0.012704261 0.03530120050000005 0.09182074289999993 <built-in function min> <built-in function max>
0.00011147125000000001 0.00015046975 0.0012640050000000001 8670000000.0 <built-in function min> <built-in function max>
0.0001088617 0.00015117575000000001 5790000000.0 9160000000.0 <built-in function min> <built-in function max>
0.000120604 0.00173741775 4550000000.0 8415999999.999994 <built-in function min> <built-in function max>
0.021388934950000004 0.09650577175 0.41502871625 0.6940128751999998 <built-in function min> <built-in function max>
0.5879329429 0.633364456 0.654157684 0.7095933372 <built-in function min> <built-in function max>
0.44351537620000003 0.45748019325 0.46174331975 0.47901701225 <built-in function min> <built-in function max>
0.4789856817 0.5503789665 0.61232146275 0.6708100425 <built-in function min> <built-in function max>
0.30537515455 0.312783016 0.31654596525 0.32516559995 <built-in function min> <built-in function max>
0.0085560507 0.0218848805 0.04386544175 0.0810052462 <built-in function min> <built-in function max>
0.0 0.0 0.0 0.0 <built-in function min> <built-in function max>
0.00031085045 0.0009124052499999999 0.007004449 0.030854618349999946 <built-in function min> <built-in function max>
0.6201102675 0.6233273615 0.624045246 0.62533233115 <built-in function min> <built-in function max>
0.02620258805 0.026775576000000002 0.02702943125 0.02883343925 <built-in function min> <built-in function max>
0.56248505395 0.5651579852499999 0.56623235625 0.56944816205 <built-in function min> <built-in function max>
1.0 1.0 1.0 1.0 <built-in function min> <built-in function max>
0.015236007950000002 0.020407867 0.043432547 0.10967084234999999 <built-in function min> <built-in function max>
```

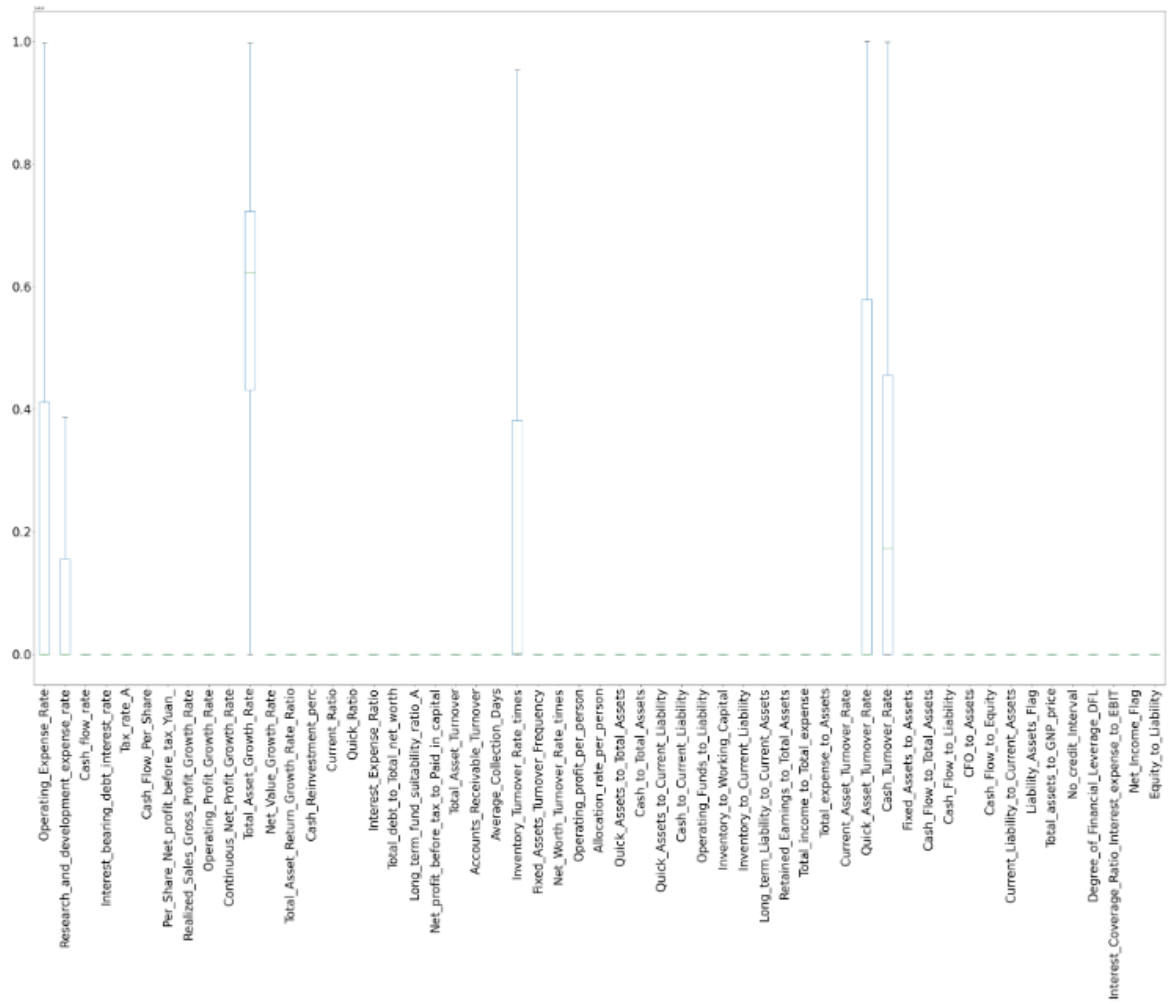

Figure 2: Box Plot after outlier treatment

2. Missing Value Treatment

We have found out the missing values as below:

Co_Code	0
Co_Name	0
Operating_Expense_Rate	0
Research_and_development_expense_rate	0
Cash_flow_rate	0
Interest_bearing_debt_interest_rate	0
Tax_rate_A	0
Cash_Flow_Per_Share	167
Per_Share_Net_profit_before_tax_Yuan_	0
Realized_Sales_Gross_Profit_Growth_Rate	0
Operating_Profit_Growth_Rate	0
Continuous_Net_Profit_Growth_Rate	0
Total_Asset_Growth_Rate	0
Net_Value_Growth_Rate	0
Total_Asset_Return_Growth_Rate_Ratio	0
Cash_Reinvestment_perc	0
Current_Ratio	0
Quick_Ratio	0
Interest_Expense_Ratio	0
Total_debt_to_Total_net_worth	21
Long_term_fund_suitability_ratio_A	0
Net_profit_before_tax_to_Paid_in_capital	0
Total_Asset_Turnover	0
Accounts_Receivable_Turnover	0
Average_Collection_Days	0
Inventory_Turnover_Rate_times	0
Fixed_Assets_Turnover_Frequency	0
Net_Worth_Turnover_Rate_times	0
Operating_profit_per_person	0
Allocation_rate_per_person	0
Quick_Assets_to_Total_Assets	0
Cash_to_Total_Assets	96
Quick_Assets_to_Current_Liability	0
Cash_to_Current_Liability	0
Operating_Funds_to_Liability	0
Inventory_to_Working_Capital	0
Inventory_to_Current_Liability	0
Long_term_Liability_to_Current_Assets	0
Retained_Earnings_to_Total_Assets	0
Total_income_to_Total_expense	0
Total_expense_to_Assets	0
Current_Asset_Turnover_Rate	0
Quick_Asset_Turnover_Rate	0
Cash_Turnover_Rate	0
Fixed_Assets_to_Assets	0
Cash_Flow_to_Total_Assets	0
Cash_Flow_to_Liability	0
CFO_to_Assets	0
Cash_Flow_to_Equity	0
Current_Liability_to_Current_Assets	14
Liability_Assets_Flag	0
Total_assets_to_GNP_price	0
No_credit_Interval	0
Degree_of_Financial_Leverage_DFL	0
Interest_Coverage_Ratio_Interest_expense_to_EBIT	0
Net_Income_Flag	0
Equity_to_Liability	0
Default	0

dtype: int64

Figure 3: DataFrame missing value count list

The columns with missing values are as follows:

- Cash_Flow_Per_Share
- Total_debt_to_Total_net_worth
- Cash_to_Total_Assets
- Current_Liability_to_Current_Assets

I have treated missing values with median and replacement with median eliminates the impact of outliers.

3. **Univariate (4 marks) & Bivariate (6 marks) analysis with proper interpretation. (You may choose to include only those variables which were significant in the model building)**

Univariate Analysis:

Operating Expense Rate

Overall Sum	4223816224000.954
No. zeros	0
No. missings	0
No. NaN	0
No. +∞	0
No. -∞	0

Histogram

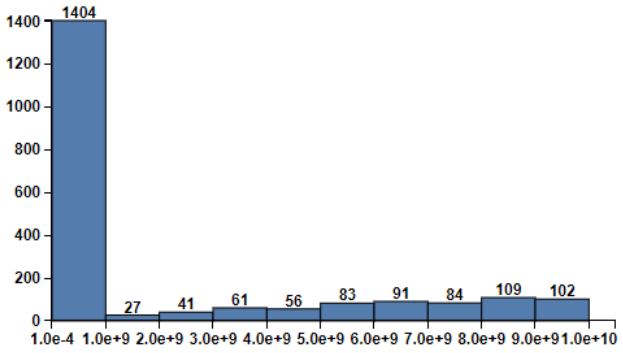

Figure 4: Histogram of Operating Expense Rate

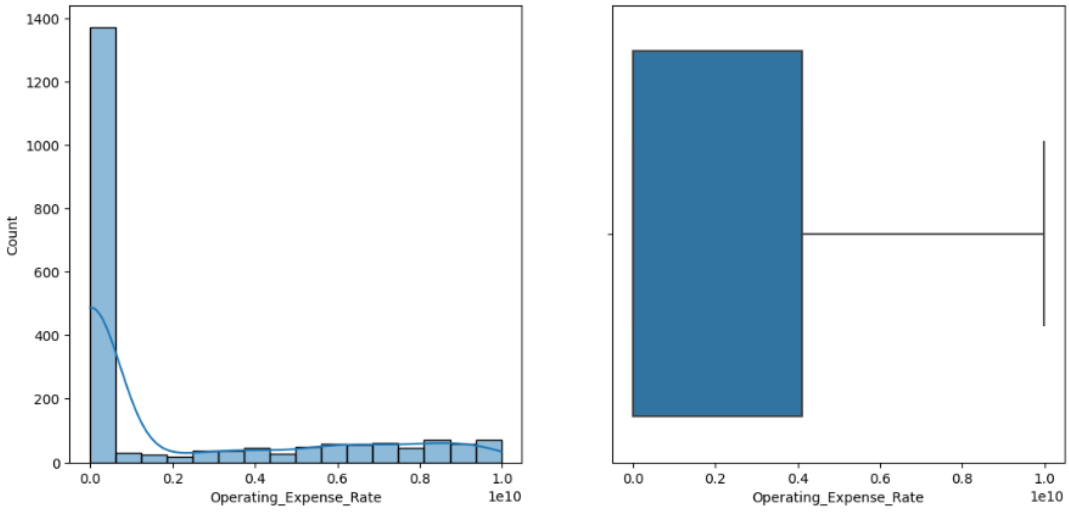

Figure 5: Box Plot of Operating Expense Rate

- The data appears to be skewed to the right (skewness 1.221), with a mean value of 2052388835.
- The minimum and maximum values are 0.0001 and 9980000000, respectively.

Research_and_development_expense_rate

Kurtosis	3.295
Overall Sum	2487369300000.154
No. zeros	910
No. missings	0
No. NaN	0
No. +∞	0
No. -∞	0

Histogram

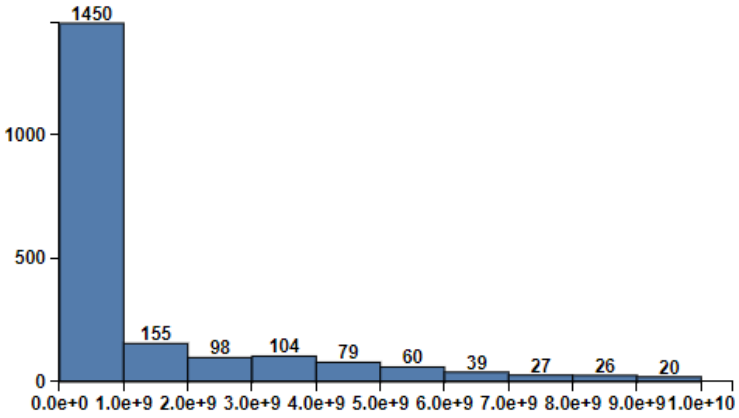

Figure 6: Histogram of Research_and_development_expense_rate

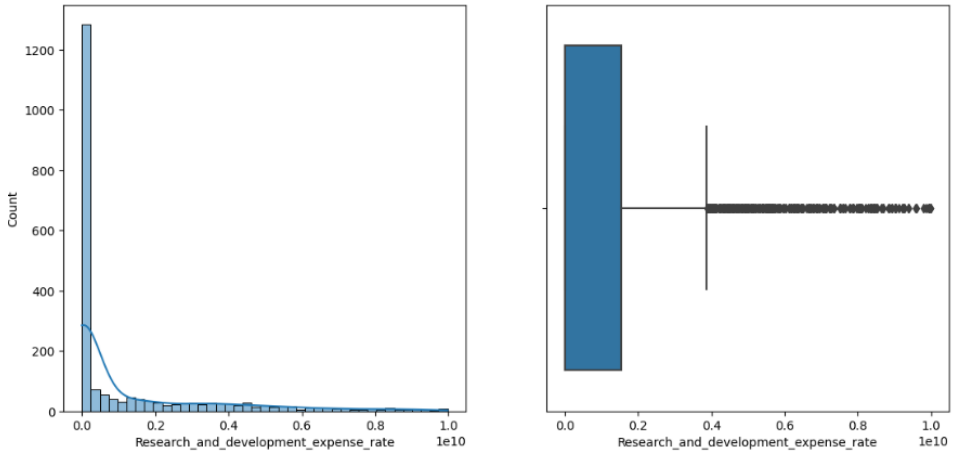

Figure 7: Box Plot of Research_and_development_expense_rate

- The data appears to be skewed to the right (skewness 1.986), with a mean value of 1208634256.560.
- The minimum and maximum values are 0 and 9980000000, respectively.
- There are a lot of outliers in Research_and_development_expense_rate.

Retained_Earnings_to_Total_Assets

Kurtosis	468.824
Overall Sum	1914.670
No. zeros	1
No. missings	0
No. NaN	0
No. +∞	0
No. -∞	0

Histogram

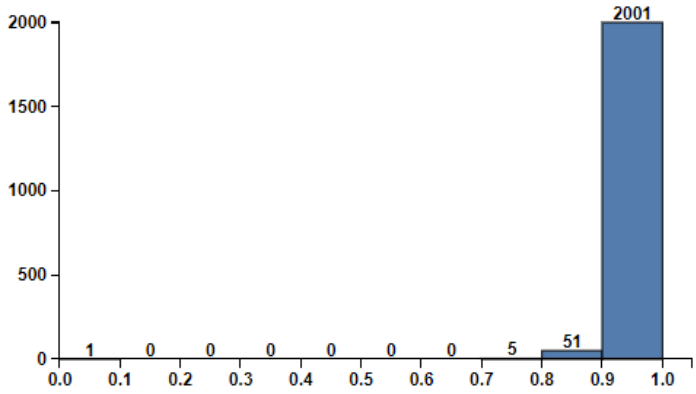

Figure 8: Histogram of Retained_Earnings_to_Total_Assets

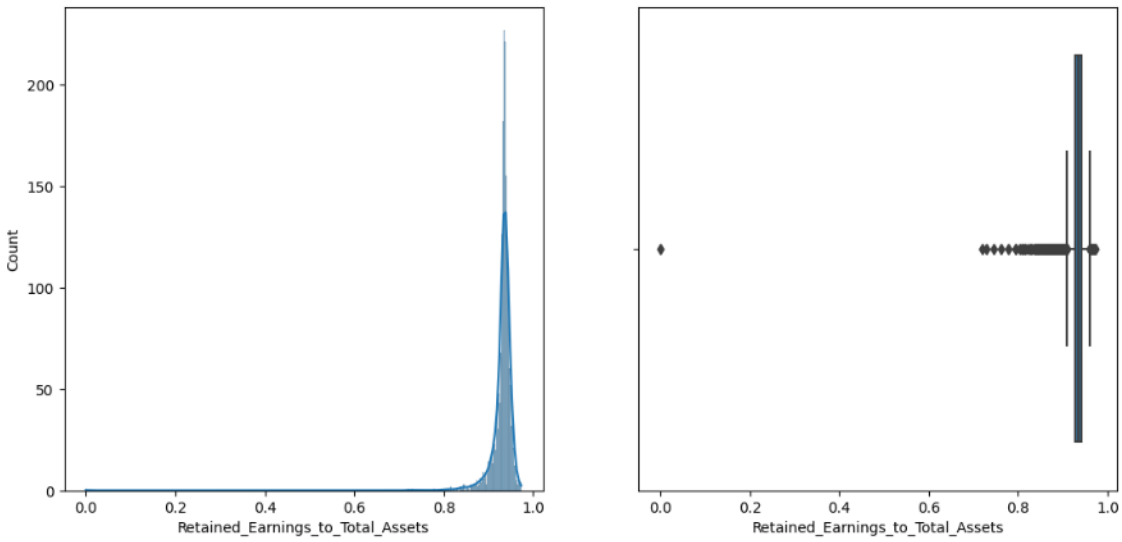

Figure 9: Histogram of Retained_Earnings_to_Total_Assets

- The data appears to be skewed to the left (skewness -16.145), with a mean value of 0.930.
- The minimum and maximum values are 0 and 0.973, respectively.
- There are a number of outliers in Retained_Earnings_to_Total_Assets.

Cash_Turnover_Rate

Kurtosis	-0.337
Overall Sum	5461305430001.089
No. zeros	0
No. missings	0
No. NaN	0
No. +∞	0
No. -∞	0

Histogram

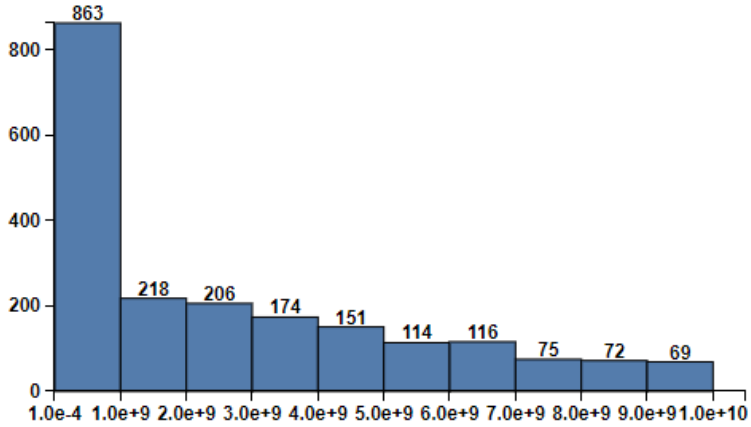

Figure 10: Histogram of Cash_Turnover_Rate

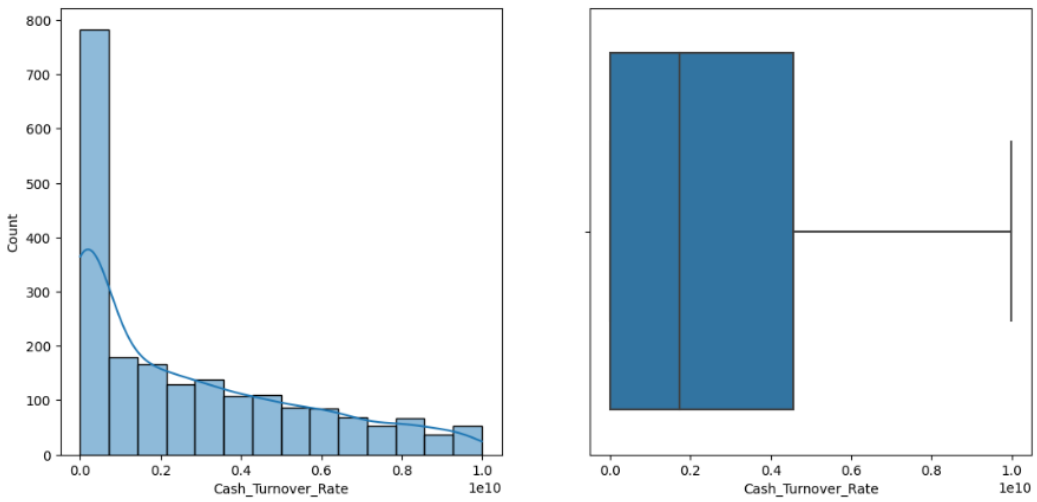

Figure 11: Histogram of Cash_Turnover_Rate

- The data appears to be skewed to the right (skewness 0.892), with a mean value of 2653695544.218.
- The minimum and maximum values are 0.0001 and 9990000000, respectively.
- There are a number of outliers in Retained_Earnings_to_Total_Assets.

Cash_Flow_to_Total_Assets

Kurtosis	35.770
Overall Sum	1325.830
No. zeros	1
No. missings	0
No. NaN	0
No. +∞	0
No. -∞	0

Histogram

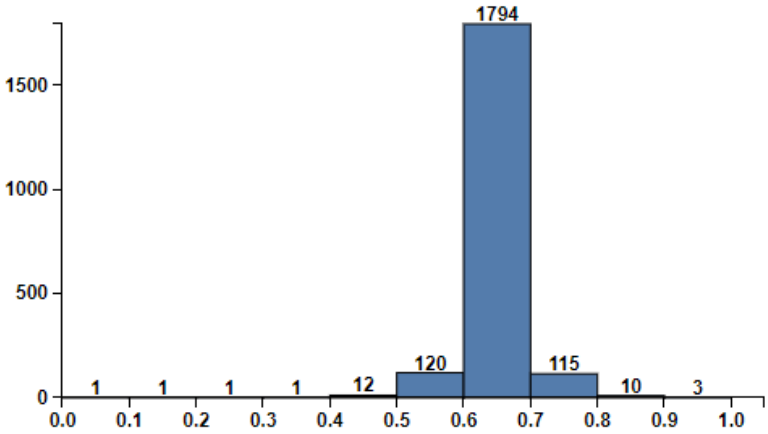

Figure 12: Histogram of Cash_Flow_to_Total_Assets

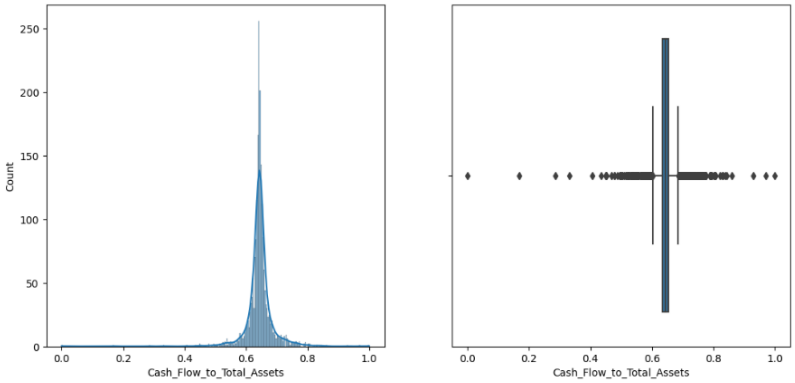

Figure 13: BoxPlot of Cash_Flow_to_Total_Assets

- The data appears to be skewed to the left (skewness -1.760), with a mean value of 0.644.
- The minimum and maximum values are 0 and 1, respectively.
- There are numerous anomalies in Cash_Flow_to_Total_Assets.

No_credit_Interval

Kurtosis	451.438
Overall Sum	1283.545
No. zeros	0
No. missings	0
No. NaN	0
No. +∞	0
No. -∞	0

Histogram

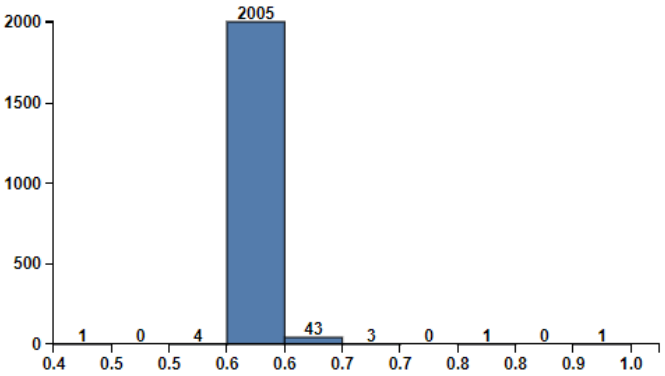

Figure 14: Histogram of No_credit_Interval

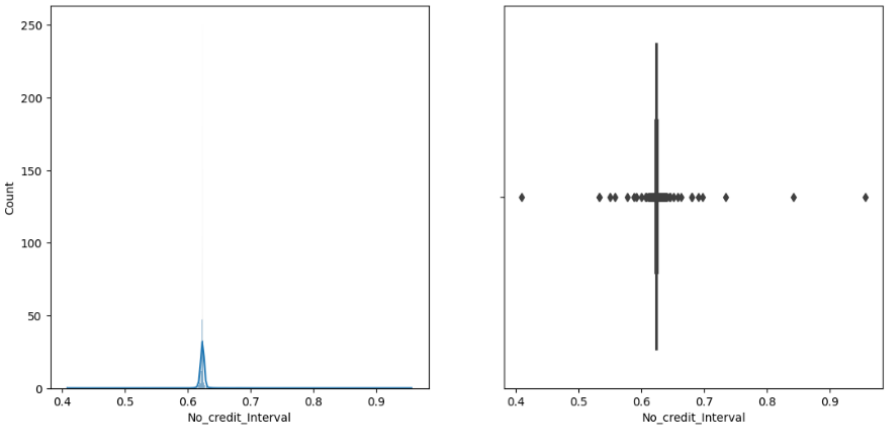

Figure 15: Box Plot of No_credit_Interval

- The data appears to be skewed to the right (skewness 11.531), with a mean value of 0.624.
- The minimum and highest values are 0.409 and 0.956, respectively.
- There are outliers in No_credit_Interval.

Interest_Coverage_Ratio_Interest_expense_to_EBIT

Kurtosis	746.014
Overall Sum	1163.666
No. zeros	0
No. missings	0
No. NaN	0
No. +∞	0
No. -∞	0

Histogram

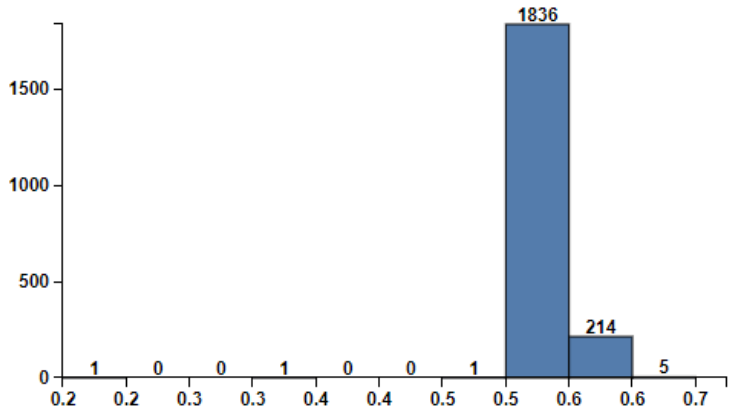

Figure 16: Histogram of Interest_Coverage_Ratio_Interest_expense_to_EBIT

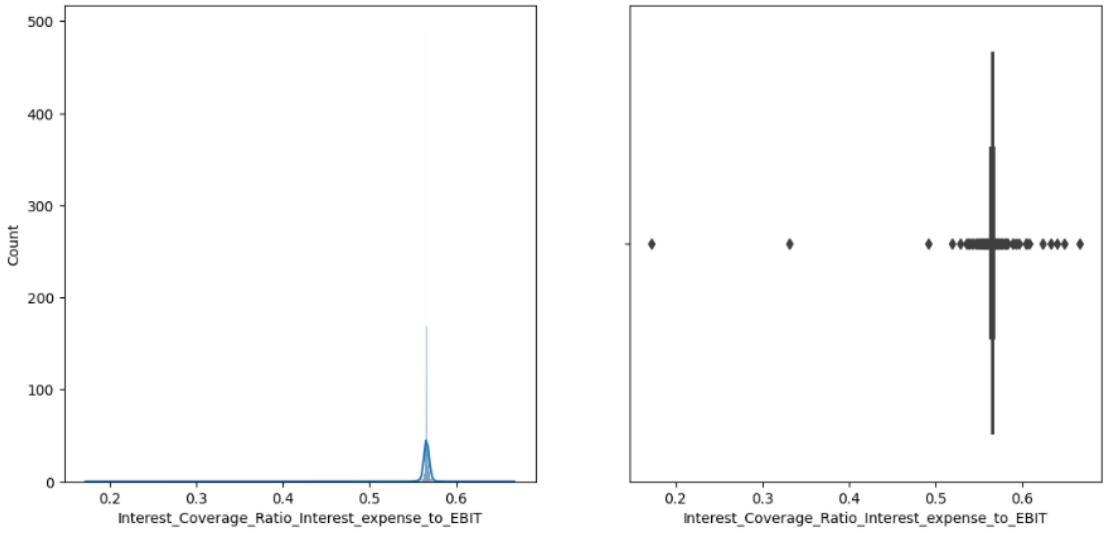

Figure 17: Box Plot of Interest_Coverage_Ratio_Interest_expense_to_EBIT

- The data appears to be skewed to the left (skewness -22.667), with a mean value of 0.565.
- The minimum and highest values are 0.172 and 0.667, respectively.
- There are outliers in Interest_Coverage_Ratio_Interest_expense_to_EBIT.

Default

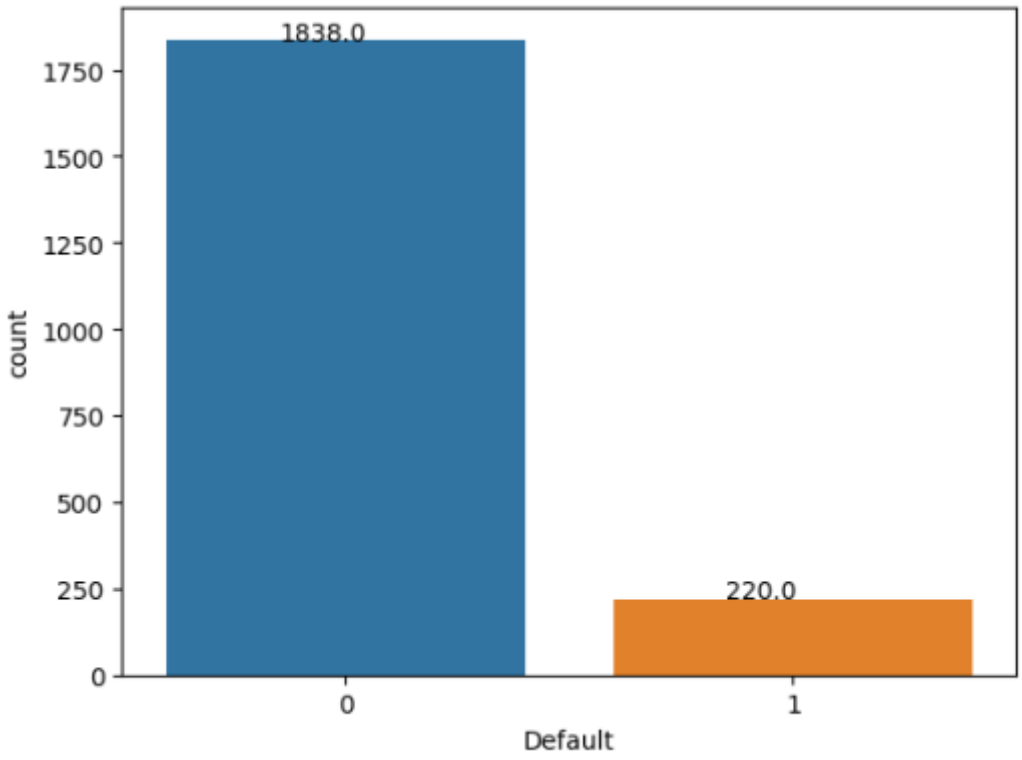

Figure 18: Count Plot of Default

- There are 1838 non-defaulters (89%) out of a total of 2058.
- There are 220 defaults (11%) out of a total of 2058.

Bivariate analysis

Operating_Expense_Rate vs Inventory_to_Working_Captial

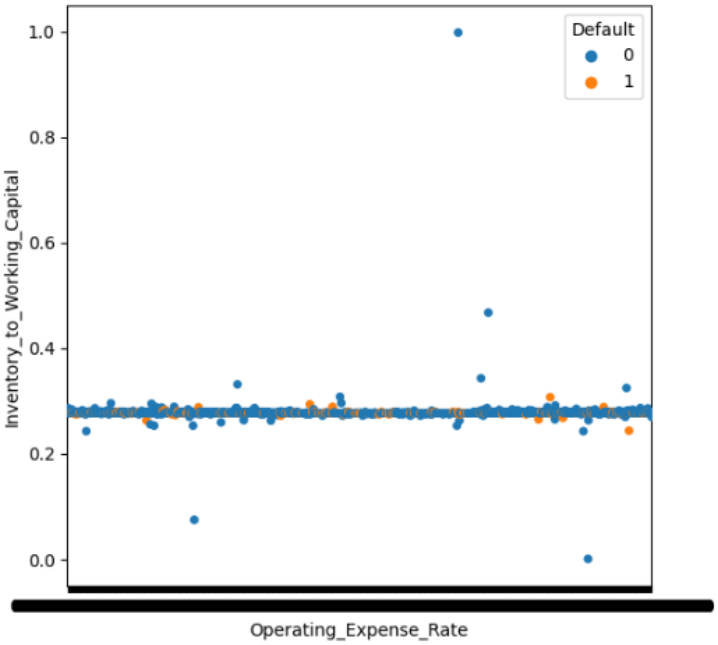

Figure 19: Operating_Expense_Rate vs Inventory_to_Working_Captial

- Inventory_to_working_capital is constant (0.27) when compare to Operating_Expense_Rate. So, there is corelation between above params.

Interest_Expense_Ratio vs Interest_bearing_debt_Interest_rate

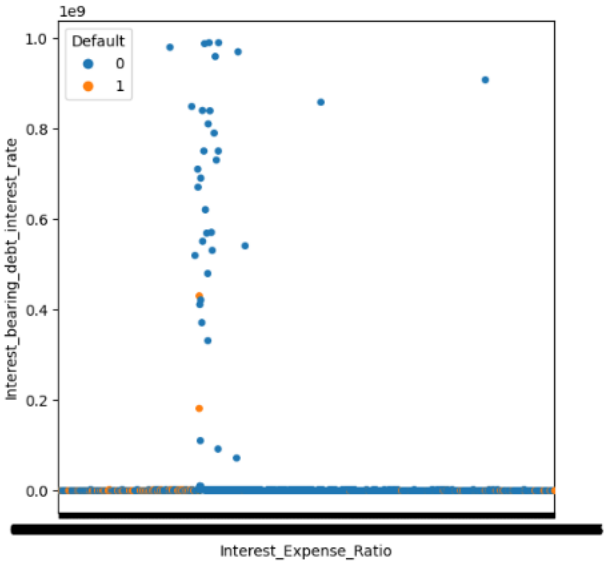

Figure 20: Interest_Expense_Ratio vs Interest_bearing_debt_Interest_rate

- As per the map, there is corelation between Interest_Expense_Ratio and Interest_bearing_debt_Interest_rate

Correlation Plot

Figure 21: Correlation map of RFE-selected characteristics

- We've plotted independent variables which is identified using RFM model. So, No correlation will be high in this correlation plot.
- We can see the minimum correlation (0.32) is identified between Cash_Flow_Total_Asset and Retained_earnings_to_Total_Assets

Scatter Plot

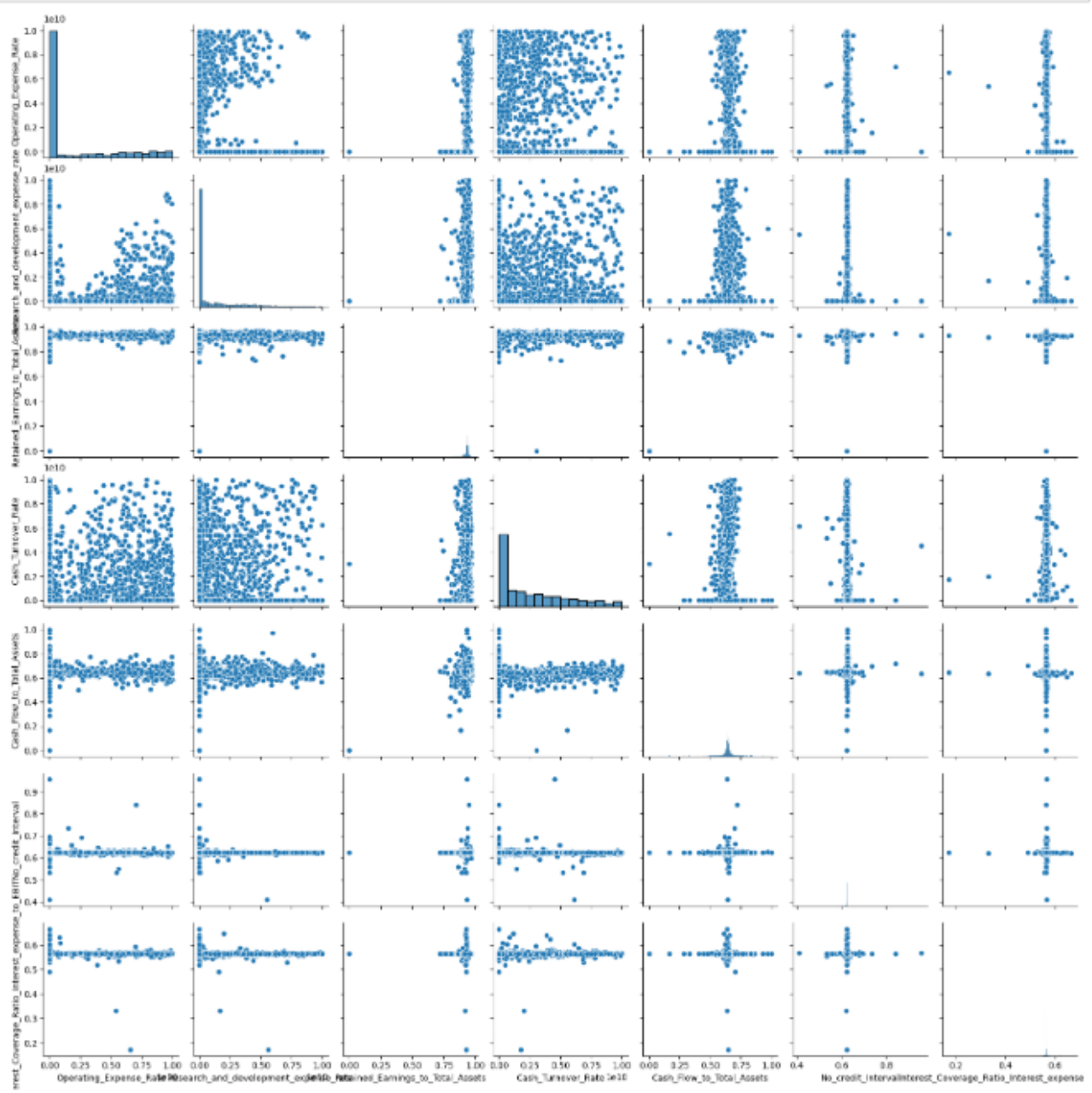

Figure 22: Scatter Plot of RFE-selected characteristics

4. Train Test Split

The original dataframe except the variables Co_Code and Co_Name is divided into dependent and independent variable type dataframe. Then both independent and dependent variable dataframe is splitted into 67:33 (train:test) ratio. One requirement for Statsmodel is that dependent and independent variables should be contained in same dataframe. So, concatenation was performed to combine dependent and independent variables arrays.

```
The number of rows (observations) in TRAIN set is 1378
The number of columns (variables) in TRAIN set is 56
```

```
The number of rows (observations) in TEST set is 680
The number of columns (variables) in TEST set is 56
```

5. 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.

We have selected top 14 features using RFE.

	Feature	Rank
0	Operating_Expense_Rate	1
1	Research_and_development_expense_rate	1
8	Operating_Profit_Growth_Rate	1
10	Total_Asset_Growth_Rate	1
16	Interest_Expense_Ratio	1
23	Inventory_Turnover_Rate_times	1
36	Retained_Earnings_to_Total_Assets	1
40	Quick_Asset_Turnover_Rate	1
41	Cash_Turnover_Rate	1
43	Cash_Flow_to_Total_Assets	1
45	CFO_to_Assets	1
50	No_credit_Interval	1
52	Interest_Coverage_Ratio_Interest_expense_to_EBIT	1
53	Net_Income_Flag	1

Figure 23: Top Features using RFE

Model Summary:

.ogit Regression Results

Dep. Variable:	Default	**No. Observations:**	1378
Model:	Logit	**Df Residuals:**	1364
Method:	MLE	**Df Model:**	13
Date:	Sun, 05 Nov 2023	**Pseudo R-squ.:**	0.3024
Time:	12:01:43	**Log-Likelihood:**	-326.37
converged:	True	**LL-Null:**	-467.84
Covariance Type:	nonrobust	**LLR p-value:**	8.829e-53

	coef	std err	z	P>|z|	[0.025	0.975]
Intercept	1495.1419	600.961	2.488	0.013	317.279	2673.005
Operating_Expense_Rate	6.917e-11	3.35e-11	2.066	0.039	3.56e-12	1.35e-10
Research_and_development_expense_rate	2.632e-10	6.77e-11	3.887	0.000	1.3e-10	3.96e-10
Operating_Profit_Growth_Rate	-1240.5302	710.343	-1.746	0.081	-2632.777	151.717
Total_Asset_Growth_Rate	-3.076e-12	4.05e-11	-0.076	0.939	-8.25e-11	7.63e-11
Interest_Expense_Ratio	55.0748	170.999	0.322	0.747	-280.078	390.227
Inventory_Turnover_Rate_times	-1.357e-11	3.54e-11	-0.383	0.702	-8.3e-11	5.59e-11
Retained_Earnings_to_Total_Assets	-110.4750	10.116	-10.921	0.000	-130.302	-90.648
Quick_Asset_Turnover_Rate	-3.027e-12	3.09e-11	-0.098	0.922	-6.35e-11	5.74e-11
Cash_Turnover_Rate	-8.349e-11	4.12e-11	-2.027	0.043	-1.64e-10	-2.77e-12
Cash_Flow_to_Total_Assets	-9.5654	4.769	-2.006	0.045	-18.913	-0.218
CFO_to_Assets	0.1467	2.001	0.073	0.942	-3.775	4.068
No_credit_Interval	-482.8066	133.956	-3.604	0.000	-745.356	-220.257
Interest_Coverage_Ratio_Interest_expense_to_EBIT	-125.1135	170.949	-0.732	0.464	-460.168	209.941

Figure 24: Logistic Model Summary

We checked the probability values for each independent variable and some of them are found to be > 0.05. So, at 95% confidence level, if p < 0.05, we can say that there is a relation between dependent and other independent variable. Alternately we can say that variables whose p > 0.05 donot have influence on the dependent variable. Therefore, a new model is prepared by discarding the variables whose p > 0.05.

Selected columns:

- Operating_Expense_Rate
- Research_and_development_expense_rate
- Retained_Earnings_to_Total_Assets
- Cash_Turnover_Rate
- Cash_Flow_to_Total_Assets
- No_credit_Interval

New Model Summary:

Logit Regression Results

Dep. Variable:	Default	**No. Observations:**	1378
Model:	Logit	**Df Residuals:**	1371
Method:	MLE	**Df Model:**	6
Date:	Sun, 05 Nov 2023	**Pseudo R-squ.:**	0.2968
Time:	14:20:43	**Log-Likelihood:**	-328.99
converged:	True	**LL-Null:**	-467.84
Covariance Type:	nonrobust	**LLR p-value:**	4.883e-57

	coef	std err	z	P>|z|	[0.025	0.975]
Intercept	402.2719	83.438	4.821	0.000	238.736	565.808
Operating_Expense_Rate	6.862e-11	3.27e-11	2.096	0.036	4.44e-12	1.33e-10
Research_and_development_expense_rate	2.676e-10	6.63e-11	4.035	0.000	1.38e-10	3.98e-10
Retained_Earnings_to_Total_Assets	-117.2409	9.131	-12.840	0.000	-135.137	-99.345
Cash_Turnover_Rate	-8.588e-11	4.09e-11	-2.099	0.036	-1.66e-10	-5.68e-12
Cash_Flow_to_Total_Assets	-10.5746	4.637	-2.280	0.023	-19.663	-1.486
No_credit_Interval	-463.8164	132.203	-3.508	0.000	-722.930	-204.702

Figure 25: Model Summary after removing P values greater 0.05

The new model having all the variables with $p < 0.05$. This model will be considered for Test set prediction and performance evaluation.

Model Evaluation on the Training Data:

Performance of 0.5 probability cut-off

	precision	recall	f1-score	support
0	0.920	0.976	0.947	1231
1	0.592	0.286	0.385	147
accuracy			0.903	1378
macro avg	0.756	0.631	0.666	1378
weighted avg	0.885	0.903	0.887	1378

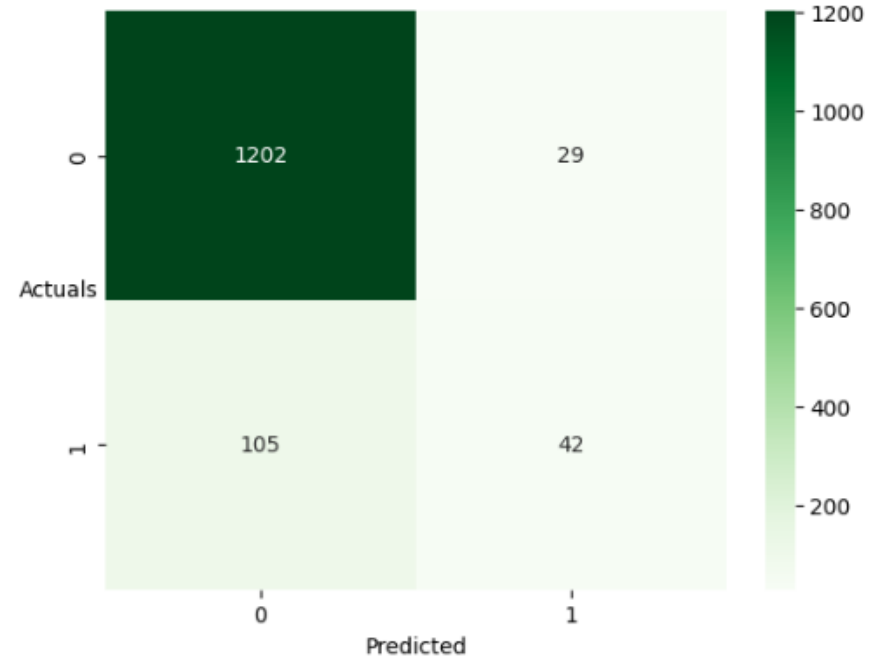

Figure 26: Classification Report of Logistic Model on Training Data using HeatMap

Overall 90% of correct predictions to total predictions were made by the model. 29% of those defaulted were correctly identified as defaulters by the model, which is not so good number.

So, we will change the probability cut-off to 0.07 as from the boxplot it is clear that “Default” status 0 has very low probability median.

Performance of 0.07 probability cut-off:

	precision	recall	f1-score	support
0	0.979	0.733	0.838	1231
1	0.280	0.871	0.424	147
accuracy			0.747	1378
macro avg	0.630	0.802	0.631	1378
weighted avg	0.905	0.747	0.794	1378

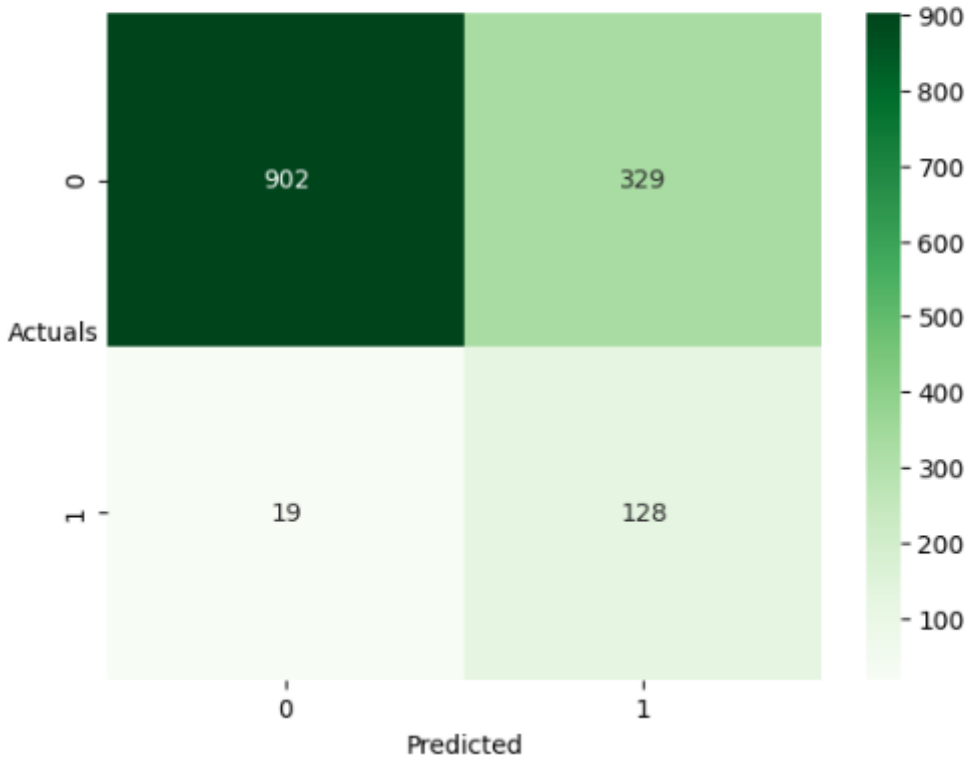

Figure 27: Classification Report of Logistic Model on Training Data using HeatMap

Accuracy of the model i.e. %overall correct predictions has decreased from 90% to 75% but sensitivity of the model has increased from 29% to 87%, which is good for our prediction. But we will try with some more probability cut-off values.

Performance of 0.09 probability cut-off:

	precision	recall	f1-score	support
0	0.972	0.791	0.872	1231
1	0.316	0.810	0.455	147
accuracy			0.793	1378
macro avg	0.644	0.800	0.664	1378
weighted avg	0.902	0.793	0.828	1378

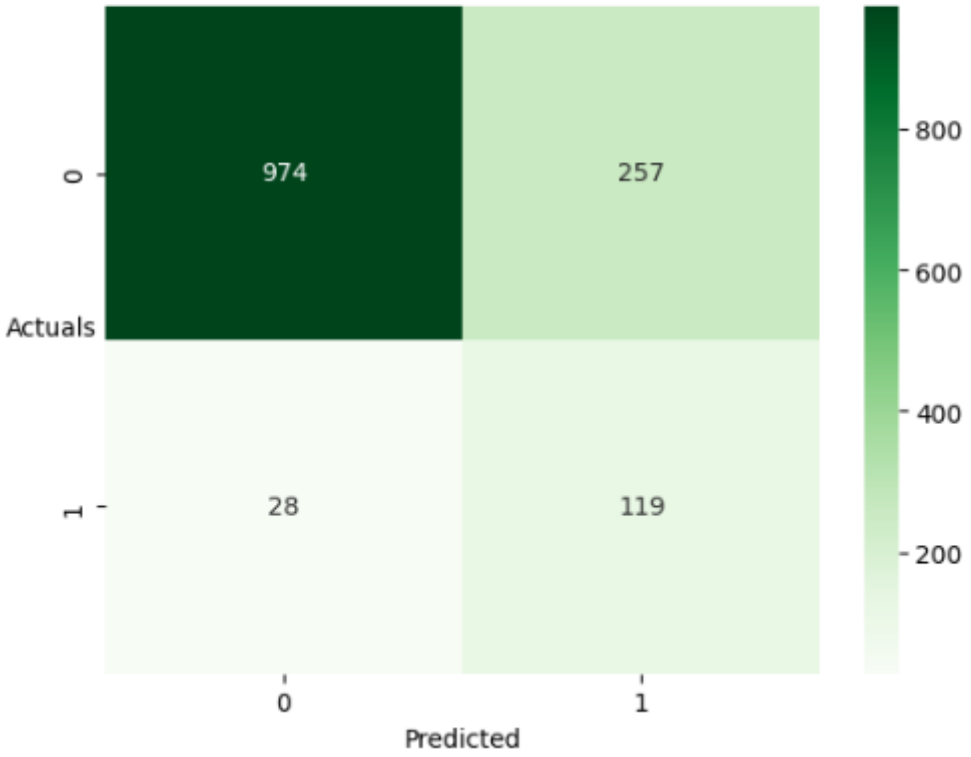

Figure 28: Classification Report of Logistic Model on Training Data using HeatMap

Accuracy of the model i.e. %overall correct predictions has increased from 74% to 79% but sensitivity of the model has decreased from 87% to 80%

Performance of 0.1 probability cut-off:

	precision	recall	f1-score	support
0	0.970	0.810	0.883	1231
1	0.331	0.789	0.467	147
accuracy			0.808	1378
macro avg	0.651	0.800	0.675	1378
weighted avg	0.902	0.808	0.838	1378

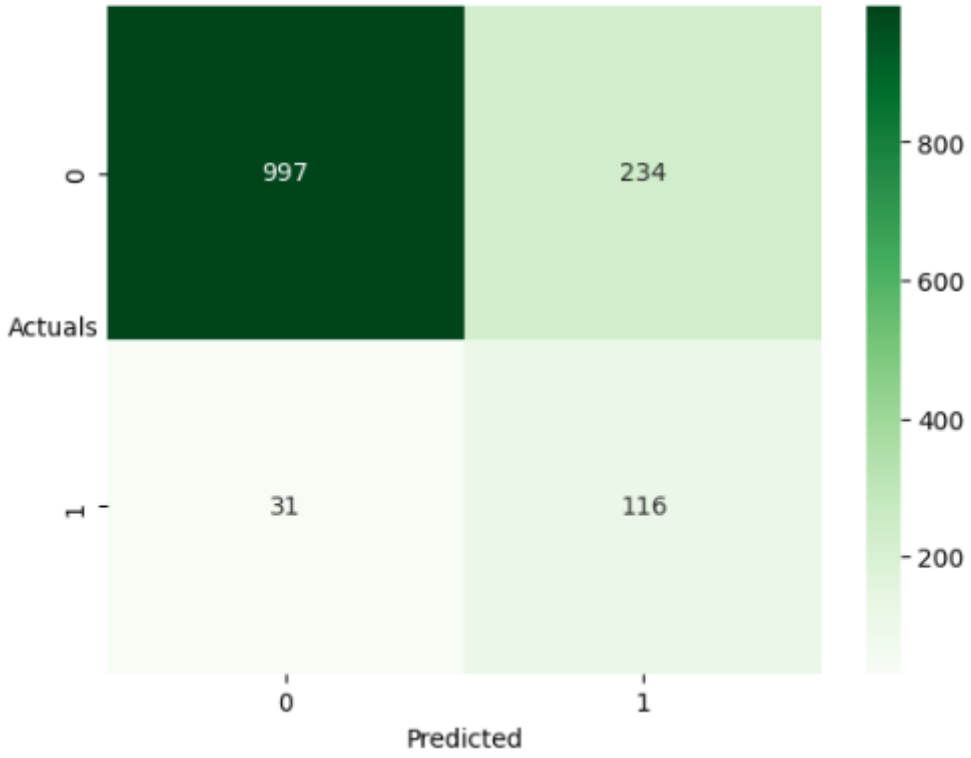

Figure 29: Classification Report of Logistic Model on Training Data using HeatMap

Accuracy of the model i.e. %overall correct predictions has increased from 79% to 80% but sensitivity of the model has not decreased (79%). But we will try with some more probability cut off values.

Performance of 0.11 probability cut-off:

	precision	recall	f1-score	support
0	0.970	0.820	0.889	1231
1	0.344	0.789	0.479	147
accuracy			0.817	1378
macro avg	0.657	0.805	0.684	1378
weighted avg	0.903	0.817	0.845	1378

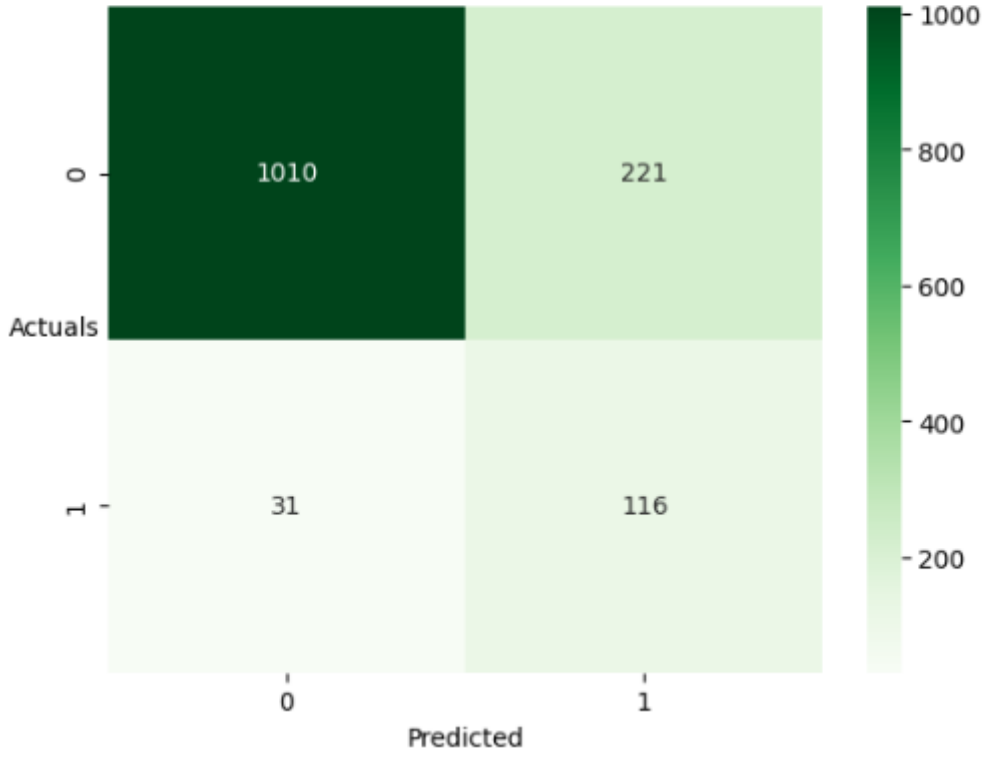

Figure 30: Classification Report of Logistic Model on Testing Data using HeatMap

Accuracy of the model i.e. %overall correct predictions has increased from 80% to 82% and sensitivity remains the same 79. We will keep this model (with p= 0.11 as cut-off) for further analysis as we are trying to maintain a balance between Accuracy and Recall.

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

Model Evaluation on the Testing Dataset

	precision	recall	f1-score	support
0	0.966	0.843	0.901	607
1	0.367	0.753	0.493	73
accuracy			0.834	680
macro avg	0.666	0.798	0.697	680
weighted avg	0.902	0.834	0.857	680

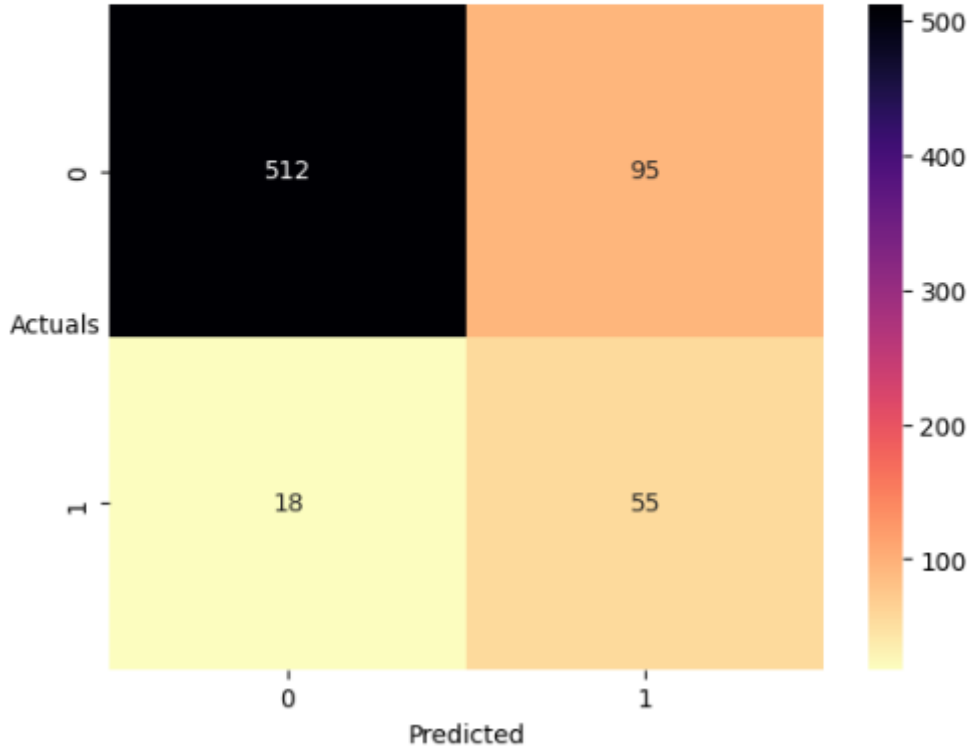

Figure 31: Classification Report of Logistic Model on Testing Data using HeatMap

Accuracy of the model i.e. % overall correct prediction is 83% and sensitivity of the model is 75%. The model performs well on the test set also.

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

Model: Random Forest

I used grid search to get the correct parameters to achieve high accuracy

min_samples_split ➔ [30,50,70,100]

min_samples_leaf ➔ [15,25,35,50]

max_depth ➔ [5,10,15,20]

random_state ➔ [42]

Best Random classifier as

```
RandomForestClassifier(max_depth=5, min_samples_leaf=15, min_samples_split=50,
                       random_state=42)
```

Figure 14: Naïve Approach Plot of Sparkling

	precision	recall	f1-score	support
0	0.94	0.99	0.96	1231
1	0.81	0.45	0.58	147
accuracy			0.93	1378
macro avg	0.88	0.72	0.77	1378
weighted avg	0.92	0.93	0.92	1378

Figure 32: Classification Report of Random Forest Model on Training Data

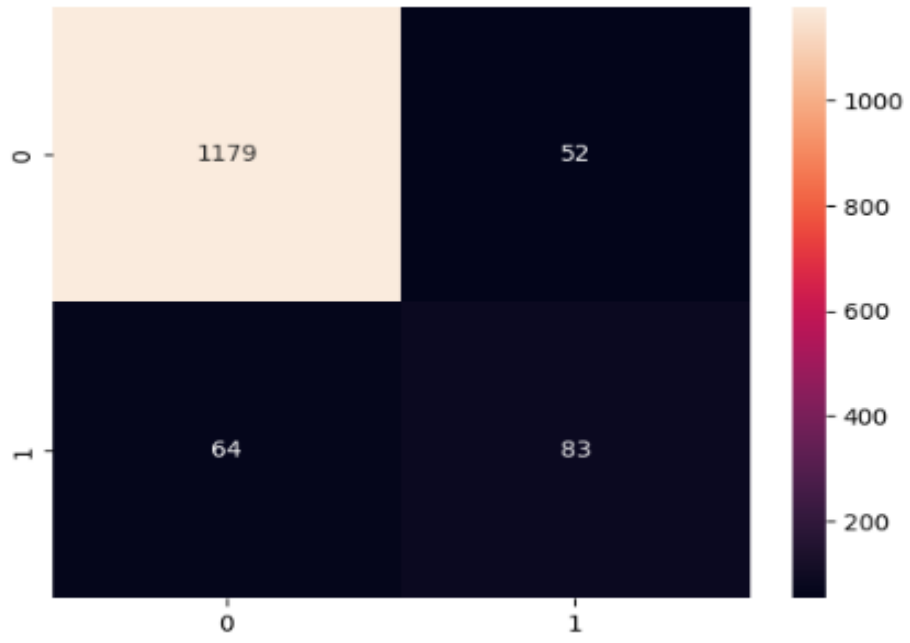

Figure 33: Classification Report of Random Forest Model on Testing Data using HeatMap

- Accuracy of the model i.e. % overall correct prediction is 93% and sensitivity of the model is 45%.
- This model is good to predict non-defaulters but not for defaulters.
- We can increase the sensitivity of the model by changing the parameters and cleaning the data as well.

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

Model Evaluation using test Data set

	precision	recall	f1-score	support
0	0.93	0.98	0.95	607
1	0.70	0.38	0.50	73
accuracy			0.92	680
macro avg	0.81	0.68	0.72	680
weighted avg	0.91	0.92	0.91	680

Figure 34: Classification Report of Random Forest Model on Testing Data

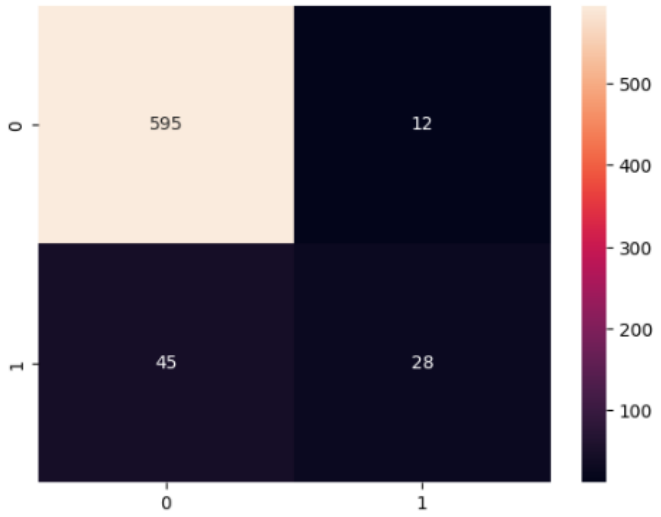

Figure 35: Classification Report of Random Forest Model on Testing Data using HeatMap

- Accuracy of the model i.e. % overall correct prediction is 92% and sensitivity of the model is 38%.
- This model is good to predict non-defaulters but not for defaulters.
- We can increase the sensitivity of the model by changing the parameters and cleaning the data as well.

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

LDA Model:

	precision	recall	f1-score	support
0	0.95	0.96	0.95	1231
1	0.61	0.56	0.59	147
accuracy			0.92	1378
macro avg	0.78	0.76	0.77	1378
weighted avg	0.91	0.92	0.91	1378

Figure 36: Classification Report of LDA Model on Traning Data

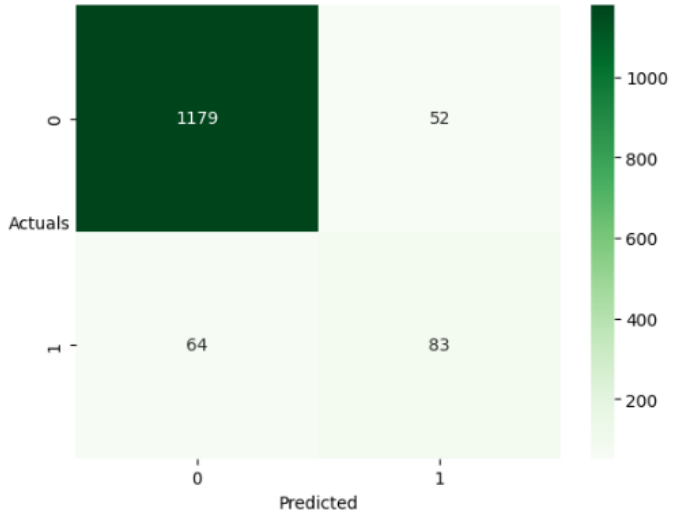

Figure 37: Classification Report of LDA Model on Training Data using HeatMap

- Accuracy of the model i.e. % overall correct prediction is 92% and sensitivity of the model is 56%.
- This model is good to predict non-defaulters but not for defaulters.
- We can increase the sensitivity of the model by changing the parameters and cleaning the data as well.

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

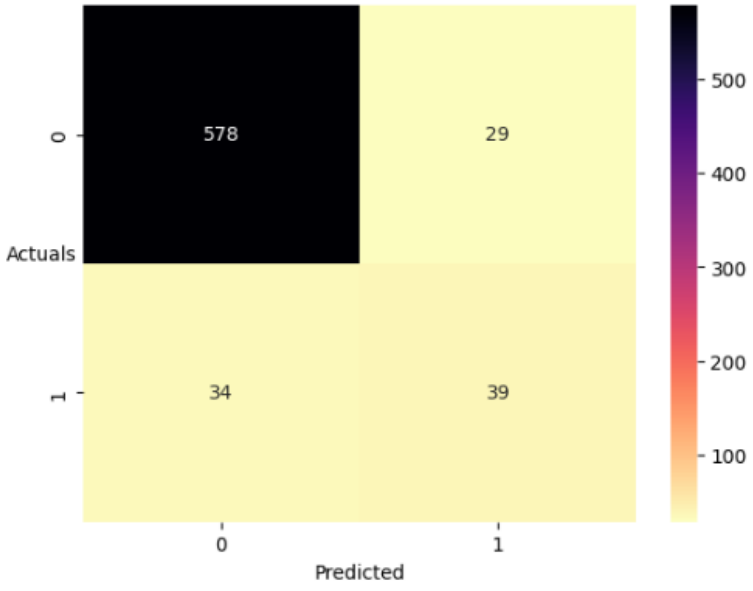

Figure 38: Classification Report of LDA Model on testing Data using HeatMap

	precision	recall	f1-score	support
0	0.94	0.95	0.95	607
1	0.57	0.53	0.55	73
accuracy			0.91	680
macro avg	0.76	0.74	0.75	680
weighted avg	0.90	0.91	0.91	680

Figure 39: Classification Report of LDA Model on Testing Data

- Accuracy of the model i.e. % overall correct prediction is 91% and sensitivity of the model is 53%.
- This model is good to predict non-defaulters but not for defaulters.
- We can increase the sensitivity of the model by changing the parameters and cleaning the data as well.

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

Logistic Model:

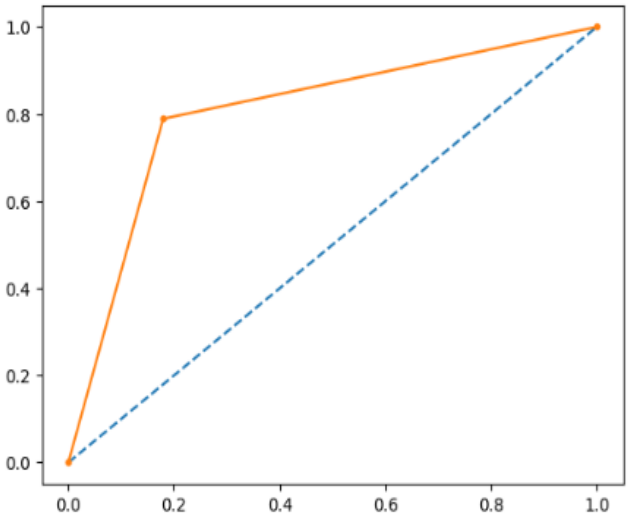

Figure 40: ROC Curve of Logistic Training Data

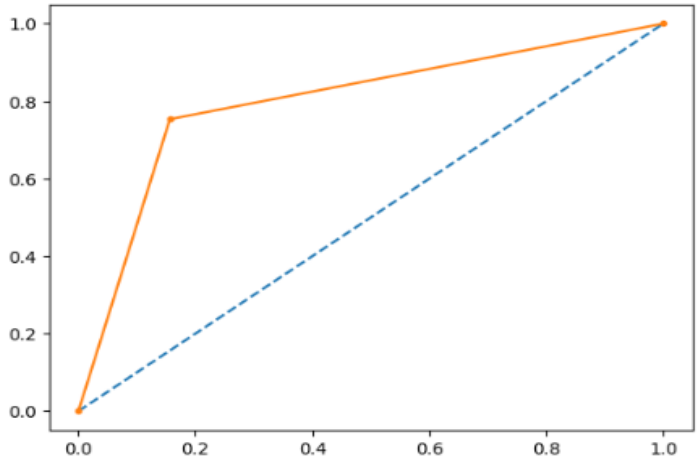

Figure 41: ROC Curve of Logistic Testing Data

Random Forest Model:

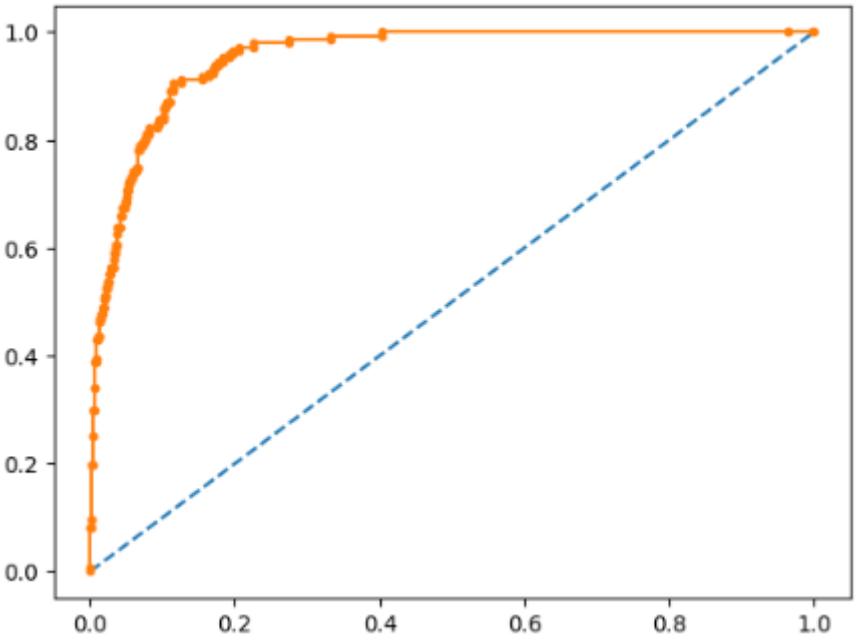

Figure 42: ROC Curve of Random Forest Training Data

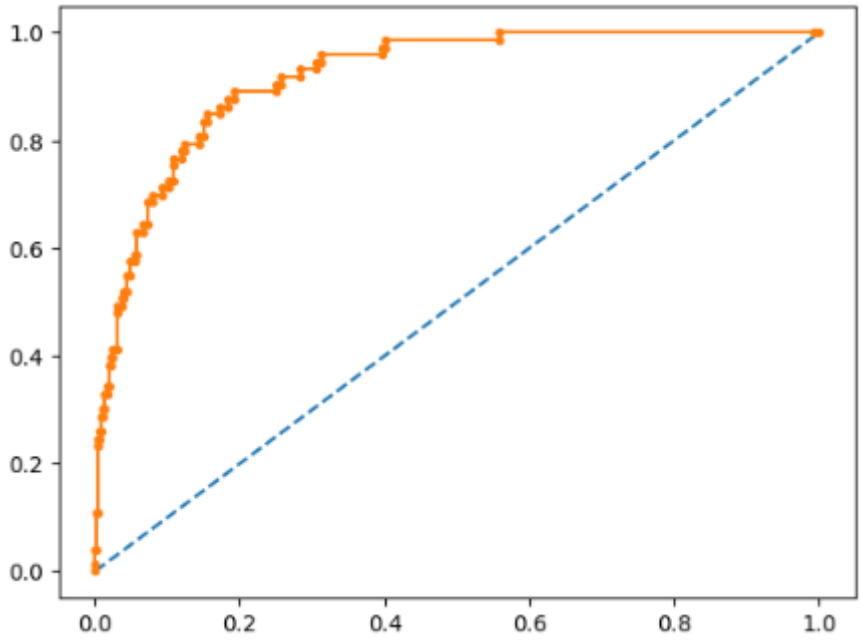

Figure 43: ROC Curve of Random Forest Testing Data

LDA Model:

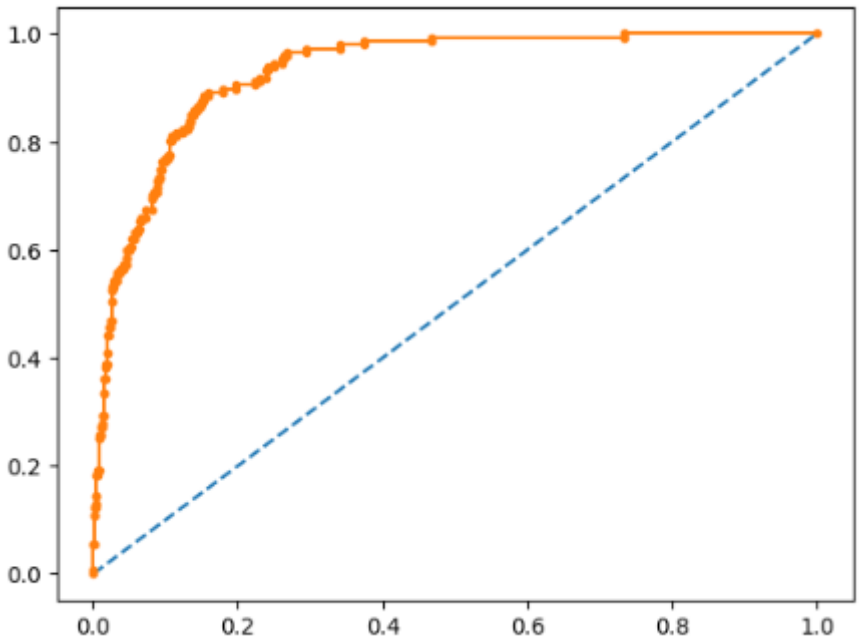

Figure 44: ROC Curve of LDA Training Data

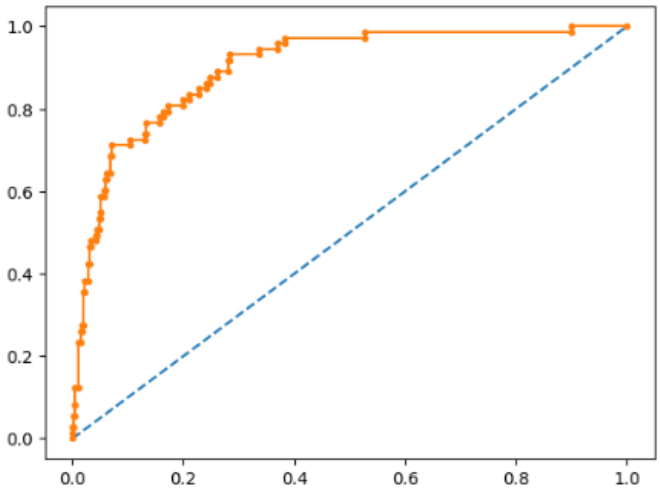

Figure 45: ROC Curve of LDA Testing Data

Sno	Model	AUC Score	
		Training	Testing
1	Logistic	0.805	0.798
2	Random Forest	0.954	0.919
3	LDA	0.930	0.901

Table 1: AUC score of various Models

- Random Forest model got the more AUC score (95.4%) among Logistic and LDA model.
- Logistic model got the low AUC score 80.5%.

12. Conclusions and Recommendations

- As per the model results, Logistic model performed well to identify the defaulters. So, we can use logistic model to predict the defaulters.
- However, it achieves only 80% on training data and 79% on testing data. So, more tuning is needed to increase the accuracy as well as predicting the defaulters.
- To identify the non-defaulters alone, we can Random Forest model. To increase the accuracy to predict defaulters, we can change the RandomForestClassifer parameters like max_depth, min_samples_leaf and min_samples_split.
- LDA model also best to identify non-defaulters alone.
- Also, we can change method for the outlier treatment (like KNN), so that we can increase the accuracy here.
- We can also VIF method to identify top independent features in logistic regression instead of RFA

Problem: PART-B

The dataset contains 6 years of information (weekly stock information) on the stock prices of 10 different Indian Stocks. Calculate the mean and standard deviation on the stock returns and share insights. You are expected to do the Market Risk Analysis using Python.

1. Draw Stock Price Graph (Stock Price vs Time) for any 2 given stocks with inference

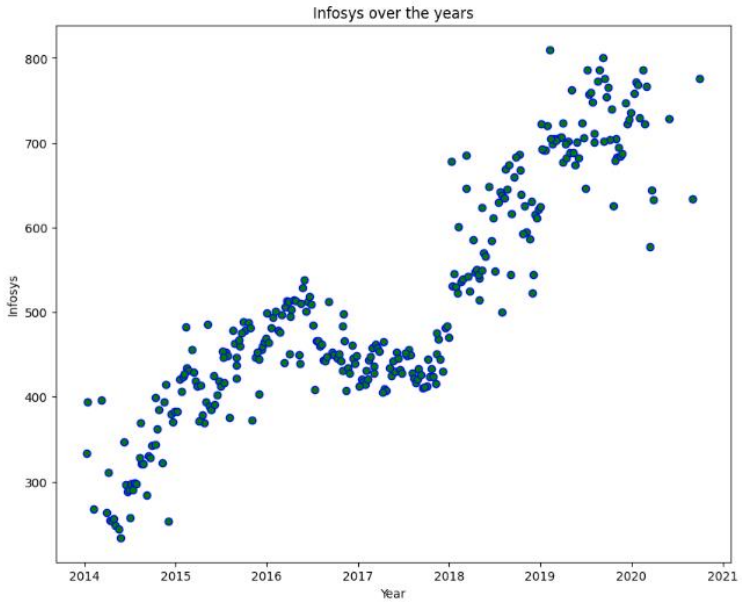

Figure 46: Stock Price Graph of Infosys

- According to the graph, Infosys stocks have given good returns throughout the years.
- There is a little drop in stock price after 2016, and it has progressively increased since 2017.

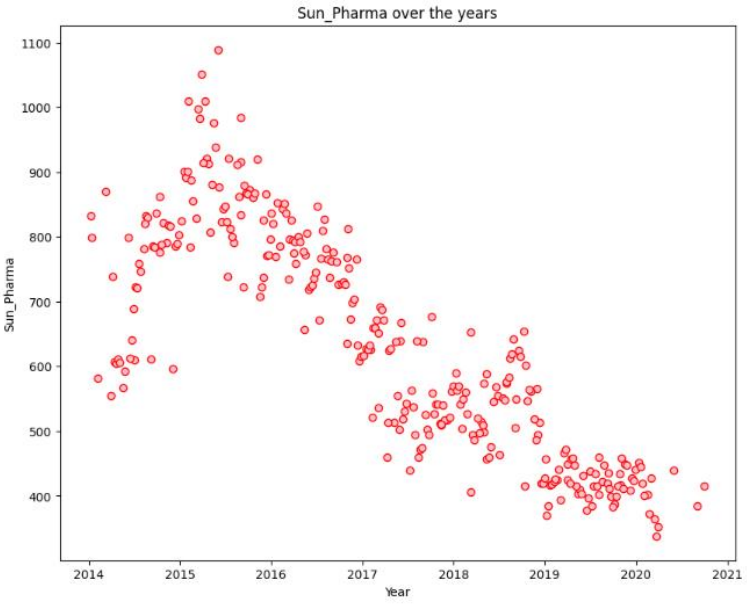

Figure 47: Stock Price Graph of Sun_Pharma

- According to the graph, Sunpharma stocks fell after 2015.
- From 2014 to 2015, there was an upward trend.

2. Calculate Returns for all stocks with inference

	Infosys	Indian_Hotel	Mahindra_&_Mahindra	Axis_Bank	SAIL	Shree_Cement	Sun_Pharma	Jindal_Steel	Idea_Vodafone	Jet_Airways
0	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
1	-0.026873	-0.014599	0.006572	0.048247	0.028988	0.032831	0.094491	-0.065882	0.011976	0.086112
2	-0.011742	0.000000	-0.008772	-0.021979	-0.028988	-0.013888	-0.004930	0.000000	-0.011976	-0.078943
3	-0.003945	0.000000	0.072218	0.047025	0.000000	0.007583	-0.004955	-0.018084	0.000000	0.007117
4	0.011788	-0.045120	-0.012371	-0.003540	-0.076373	-0.019515	0.011523	-0.140857	-0.049393	-0.148846

Figure 48: Returns of Stock(Top 5 Data)

```
Infosys                0.874521
Indian_Hotel           0.083382
Mahindra_&_Mahindra   -0.471323
Axis_Bank              0.365382
SAIL                  -1.084013
Shree_Cement           1.152290
Sun_Pharma            -0.455337
Jindal_Steel          -1.290374
Idea_Vodafone         -3.320228
Jet_Airways           -2.988564
dtype: float64
```

Figure 49: Sum of Stock Returns

- Shree_Cement, Infosys, and Indian_Hotel were all profitable.
- Among all stocks, Idea_Vodafond and Jet_Airways provided the lowest returns.

3. Calculate Stock Means and Standard Deviation for all stocks with inference

	Average	Volatility
Infosys	0.002794	0.035070
Indian_Hotel	0.000266	0.047131
Mahindra_&_Mahindra	-0.001506	0.040169
Axis_Bank	0.001167	0.045828
SAIL	-0.003463	0.062188
Shree_Cement	0.003681	0.039917
Sun_Pharma	-0.001455	0.045033
Jindal_Steel	-0.004123	0.075108
Idea_Vodafone	-0.010608	0.104315
Jet_Airways	-0.009548	0.097972

Figure 50: Average and Volatility of Stocks

- Shree Cement has a strong average (mean) and volatility (standard deviation).
- Indian_Hotel and Infosys have high volatility.
- Idea_Vodafone has the lowest returns, while Shree_Cements has the best.
- Idea_Vodafone has the highest danger factor, whereas Infosys has the lowest risk factor.

4. Draw a plot of Stock Means vs Standard Deviation and state your inference

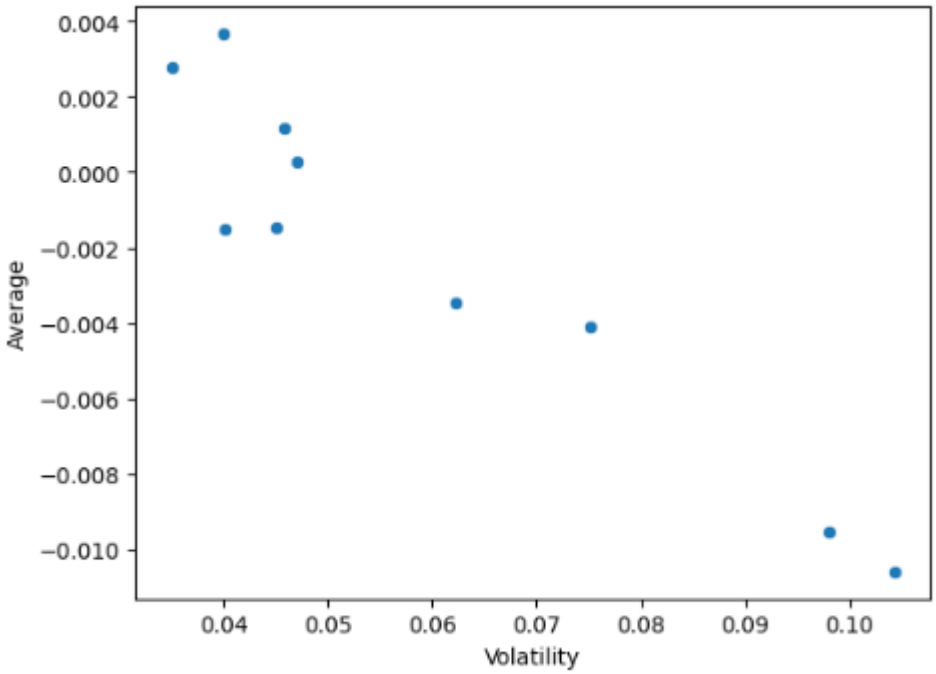

Figure 51: Volatility vs Average

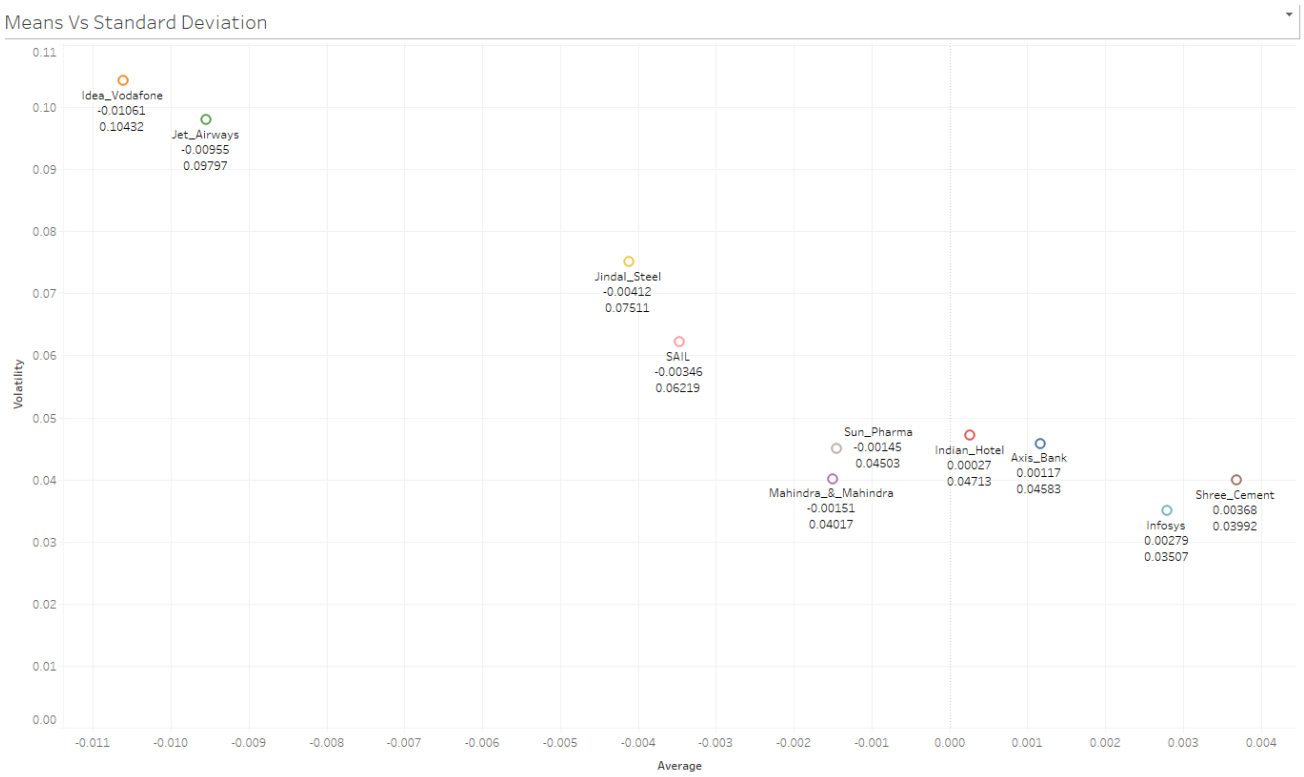

Figure 52: Volatility vs Average

- Stocks higher up but on the far left suggest high volatility and low returns, whereas stocks lower down but on the far right indicate low volatility and high returns..

5. Conclusions and Recommendations

- In this dataset, Shree Cements is the best, followed by Infosys and Axis_Bank.
- From, pure risk perspective Infosys followed by Shree cement and Mahindra_&_Mahindra looks good in this dataset.
- More volatile stocks may provide short-term gains but may not be a suitable investment in the short term but may provide long-term returns.
- Investing in Infosys and Shree_Cement will thus provide good profits.
- If a stock falls by more than 20%, averaging the stocks will lessen the loss.