FRA PROJECT MILESTONE -1

COMPANY CREDIT RISK ANALYSIS

PROBLEM STATEMENT

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 interests 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 (2015). Also, information about the Networth of the company in the following year (2016) is provided which can be used to drive the labeled field.

1.1 Outlier Treatment

1.2 Missing Value Treatment

1.3 Transform Target variable into 0 and 1

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

1.5 Train Test Split

1.6 Build Logistic Regression Model (using statsmodel library) on most important variables on Train Dataset and choose the optimum cutoff. Also showcase your model building approach

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

The database is a balance sheet that consists of records of around 3500+ companies with around 60+ parameters or metrics describing its market positions, all the data being numeric in nature. So we have to analyze the data and predict if the company will default or not. So we do Logistic Regression to classify and predict the defaulters.

So, we start our analysis by importing the required libraries for data operations, visualizations , regression, evaluation of data/ validating the data like – **numpy, pandas, seaborn, matplotlib, statsmodels,etc**

We use read_excel from pandas to read the data: -

	Co_Code	Co_Name	Networth Next Year	Equity Paid Up	Networth	Capital Employed	Total Debt	Gross Block	Net Working Capital	Current Assets	...	PBIDTM (%) [Latest]	PBITM (%) [Latest]	PBDTM (%) [Latest]	CPM (%) [Latest]	APATM (%) [Latest]
0	16974	Hind.Cables	-8021.60	419.36	-7027.48	-1007.24	5936.03	474.30	-1076.34	40.50	...	0.00	0.00	0.00	0.00	0.00
1	21214	Tata Tele. Mah.	-3986.19	1954.93	-2968.08	4458.20	7410.18	9070.86	-1098.88	486.86	...	-10.30	-39.74	-57.74	-57.74	-87.18
2	14852	ABG Shipyard	-3192.58	53.84	506.86	7714.68	6944.54	1281.54	4496.25	9097.64	...	-5279.14	-5516.98	-7780.25	-7723.67	-7961.51
3	2439	GTL	-3054.51	157.30	-623.49	2353.88	2326.05	1033.69	-2612.42	1034.12	...	-3.33	-7.21	-48.13	-47.70	-51.58
4	23505	Bharati Defence	-2967.36	50.30	-1070.83	4675.33	5740.90	1084.20	1836.23	4685.81	...	-295.55	-400.55	-845.88	379.79	274.79

Now, we use replace function for replacing some elements of a string like (becomes _ and removing spaces and % from the column names.

PBDTM (%) [Latest]	CPM (%) [Latest]	APATM (%) [Latest]	Debtors Velocity (Days)	Creditors Velocity (Days)	Inventory Velocity (Days)	Value of Output/Total Assets

Becomes

perc_Latest	CPM_perc_Latest	APATM_perc_Latest	Debtors_Velocity_Days	Creditors_Velocity_Days	Inventory_Velocity_Days	Value_of_Output_to_Total_Assets

Now, we check the dimensions of the data i.e rows and colums using shape : -

```
The number of rows (observations) is 3586
The number of columns (variables) is 67
```

Info() returns the datatype and number of values present in the columns. Below are some observations : -

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 3586 entries, 0 to 3585
Data columns (total 67 columns):
 #   Column                             Non-Null Count  Dtype
---  ------                             --------------  -----
 0   Co_Code                            3586 non-null   int64
 1   Co_Name                            3586 non-null   object
 2   Networth_Next_Year                 3586 non-null   float64
 3   Equity_Paid_Up                     3586 non-null   float64
 4   Networth                           3586 non-null   float64
 5   Capital_Employed                   3586 non-null   float64
 6   Total_Debt                         3586 non-null   float64
 7   Gross_Block_                       3586 non-null   float64
 8   Net_Working_Capital_               3586 non-null   float64
 9   Current_Assets_                    3586 non-null   float64
 10  Current_Liabilities_and_Provisions_  3586 non-null   float64
 11  Total_Assets_to_Liabilities_       3586 non-null   float64
```

```
 27  Capital_expenses_in_forex              3586 non-null   float64
 28  Book_Value_Unit_Curr                   3586 non-null   float64
 29  Book_Value_Adj__Unit_Curr              3582 non-null   float64
 30  Market_Capitalisation                  3586 non-null   float64
 31  CEPS_annualised_Unit_Curr              3586 non-null   float64
 32  Cash_Flow_From_Operating_Activities    3586 non-null   float64
```

```
 59  PBDTM_perc_Latest                     3585 non-null   float64
 60  CPM_perc_Latest                       3585 non-null   float64
 61  APATM_perc_Latest                     3585 non-null   float64
 62  Debtors_Velocity_Days                 3586 non-null   int64
 63  Creditors_Velocity_Days               3586 non-null   int64
 64  Inventory_Velocity_Days               3483 non-null   float64
 65  Value_of_Output_to_Total_Assets       3586 non-null   float64
 66  Value_of_Output_to_Gross_Block        3586 non-null   float64
dtypes: float64(63), int64(3), object(1)
memory usage: 1.8+ MB
```

Now we want to see the 5 number summary of the data and hence we use the describe function : -

	count	mean	std	min	25%	50%	75%	max
Co_Code	3586.00	16065.39	19776.82	4.00	3029.25	6077.50	24269.50	72493.00
Networth_Next_Year	3586.00	725.05	4769.68	-8021.60	3.98	19.02	123.80	111729.10
Equity_Paid_Up	3586.00	62.97	778.76	0.00	3.75	8.29	19.52	42263.46
Networth	3586.00	649.75	4091.99	-7027.48	3.89	18.58	117.30	81657.35
Capital_Employed	3586.00	2799.61	26975.14	-1824.75	7.60	39.09	226.61	714001.25
Total_Debt	3586.00	1994.82	23652.84	-0.72	0.03	7.49	72.35	652823.81
Gross_Block_	3586.00	594.18	4871.55	-41.19	0.57	15.87	131.90	128477.59
Net_Working_Capital_	3586.00	410.81	6301.22	-13162.42	0.94	10.14	61.17	223257.56
Current_Assets_	3586.00	1960.35	22577.57	-0.91	4.00	24.54	135.28	721166.00
Current_Liabilities_and_Provisions_	3586.00	391.99	2675.00	-0.23	0.73	9.23	65.65	83232.98
Total_Assets_to_Liabilities_	3586.00	1778.45	11437.57	-4.51	10.55	52.01	310.54	254737.22
Gross_Sales	3586.00	1123.74	10603.70	-62.59	1.44	31.21	242.25	474182.94
Net_Sales	3586.00	1079.70	9996.57	-62.59	1.44	30.44	234.44	443775.16
Other_Income	3586.00	48.73	426.04	-448.72	0.02	0.45	3.63	14143.40
Value_Of_Output	3586.00	1077.19	9843.88	-119.10	1.41	30.89	235.84	435559.09
Cost_of_Production	3586.00	798.54	9076.70	-22.65	0.94	25.99	189.55	419913.50
Selling_Cost	3586.00	25.55	194.24	0.00	0.00	0.16	3.88	5283.91
PBIDT	3586.00	248.18	1949.59	-4655.14	0.04	2.04	23.52	42059.26
PBDT	3586.00	116.27	956.20	-5874.53	0.00	0.80	12.95	23215.00
PBIT	3586.00	217.66	1850.97	-4812.95	0.00	1.15	16.67	41402.96
PBT	3586.00	85.75	799.93	-6032.34	-0.06	0.31	7.42	16798.00
PAT	3586.00	61.22	620.30	-6032.34	-0.06	0.26	5.54	13383.39
Adjusted_PAT	3586.00	60.06	580.43	-4418.72	-0.09	0.21	5.34	13384.11
CP	3586.00	91.73	780.79	-5874.53	0.00	0.74	10.91	20760.20
Revenue_earnings_in_forex	3586.00	131.17	1150.73	0.00	0.00	0.00	7.20	46158.00
Revenue_expenses_in_forex	3586.00	256.33	4132.34	0.00	0.00	0.00	6.99	193979.73

ROG_Revenue_earnings_in_forex_perc	3586.00	37.23	658.67	-100.00	0.00	0.00	0.00	29084.77
ROG_Revenue_expenses_in_forex_perc	3586.00	364.86	15233.64	-100.00	0.00	0.00	0.00	894591.69
ROG_Market_Capitalisation_perc	3586.00	63.68	1047.93	-98.05	0.00	0.00	47.52	61865.26
Current_Ratio_Latest	3585.00	12.06	108.41	0.00	0.88	1.36	2.77	4813.00
Fixed_Assets_Ratio_Latest	3585.00	51.54	681.15	0.00	0.27	1.56	4.74	22172.00
Inventory_Ratio_Latest	3585.00	37.80	458.19	0.00	0.00	3.56	8.94	15472.00
Debtors_Ratio_Latest	3585.00	33.03	489.56	0.00	0.42	3.82	8.52	22992.67
Total_Asset_Turnover_Ratio_Latest	3585.00	1.24	2.67	0.00	0.07	0.60	1.55	57.75
Interest_Cover_Ratio_Latest	3585.00	16.39	351.74	-5450.00	0.00	1.08	3.71	18639.40
PBIDTM_perc_Latest	3585.00	-51.16	1795.13	-78870.45	0.00	8.07	18.99	19233.33
PBITM_perc_Latest	3585.00	-109.21	3057.64	-141600.00	0.00	5.23	14.29	19195.70
PBDTM_perc_Latest	3585.00	-311.57	10921.59	-590500.00	0.00	4.69	14.11	15640.00
CPM_perc_Latest	3585.00	-307.01	10676.15	-572000.00	0.00	3.89	11.39	15640.00
APATM_perc_Latest	3585.00	-365.06	12500.05	-688600.00	0.00	1.59	7.41	15266.67
Debtors_Velocity_Days	3586.00	603.89	10636.76	0.00	8.00	49.00	106.00	514721.00
Creditors_Velocity_Days	3586.00	2057.85	54169.48	0.00	8.00	39.00	89.00	2034145.00
Inventory_Velocity_Days	3483.00	79.64	137.85	-199.00	0.00	35.00	96.00	996.00
Value_of_Output_to_Total_Assets	3586.00	0.82	1.20	-0.33	0.07	0.48	1.16	17.63
Value_of_Output_to_Gross_Block	3586.00	61.88	976.82	-61.00	0.27	1.53	4.91	43404.00

MISSING VALUES AND ITS TREATMENT

Now we find the number of missing values using isnull().sum() : -

```
Inventory_Velocity_Days              103
Book_Value_Adj__Unit_Curr              4
Interest_Cover_Ratio_Latest            1
PBITM_perc_Latest                      1
Fixed_Assets_Ratio_Latest              1
Inventory_Ratio_Latest                 1
Debtors_Ratio_Latest                   1
Total_Asset_Turnover_Ratio_Latest      1
PBIDTM_perc_Latest                     1
PBDTM_perc_Latest                      1
CPM_perc_Latest                        1
APATM_perc_Latest                      1
Current_Ratio_Latest                   1
```

These are the columns that have missing values. So there are total of 118 missing values.

Since there are very few missing values , we can remove them by dropping them and set inplace=True and then check the number of null values in the dataset.

```
0
```

Now, we also drop Co_Code and Co_Name since they don't add any value to the data and prediction.

CREATING TARGET VARIABLE – DEFAULT (values 1 and 0)

Now, we create a target variable "Default" based on "Networth_Next_Year" where if is >0, the default will be 0, and if it is <0, the default will be 1

	default	Networth_Next_Year
723	0	3.00
1152	0	6.80
3402	0	6211.96
1613	0	14.97
2929	0	338.43
238	1	-6.86
612	0	1.92
134	1	-38.60
2775	0	194.12
2497	0	91.53

```
df['default'].value_counts(normalize=True)
```

```
0    0.89
1    0.11
Name: default, dtype: float64
```

```
df['default'].value_counts()
```

```
0    3101
1     377
Name: default, dtype: int64
```

Now, we use the StandardScaler to scale the variables and then drop the index column.

	Networth_Next_Year	Equity_Paid_Up	Networth	Capital_Employed	Total_Debt	Gross_Block_	Net_Working_Capital_	Current_Assets_
130	-0.16	-0.05	-0.15	-0.10	-0.08	-0.11	-0.06	-0.08
1595	-0.15	-0.06	-0.15	-0.10	-0.08	-0.10	-0.06	-0.08
1880	-0.15	-0.08	-0.15	-0.10	-0.08	-0.12	-0.06	-0.08
224	-0.15	-0.07	-0.16	-0.11	-0.08	-0.12	-0.06	-0.09
1803	-0.15	-0.06	-0.15	-0.10	-0.08	-0.06	-0.07	-0.08

IDENTIFYING OUTLIERS and TREATING OUTLIERS

Now, we define limits (upper and lower limits) as UL= Q3+(1.5*IQR) and LL= Q1-(1.5*IQR), anything above UL and below LL is identified as outlier.

Below are some of the columns with number of outliers : -

Networth_Next_Year	652
Equity_Paid_Up	431
Networth	631
Capital_Employed	575
Total_Debt	560
Gross_Block_	525
Net_Working_Capital_	607
Current_Assets_	558
Current_Liabilities_and_Provisions_	565
Total_Assets_to_Liabilities_	555
Gross_Sales	537
Net_Sales	541
Other_Income	576
Value_Of_Output	543
Cost_of_Production	539
Selling_Cost	586
PBIDT	663
PBDT	792
PBIT	706
PBT	927
PAT	927
Adjusted_PAT	938
CP	795
Revenue_earnings_in_forex	703
Revenue_expenses_in_forex	675
Capital_expenses_in_forex	676
Book_Value_Unit_Curr	473

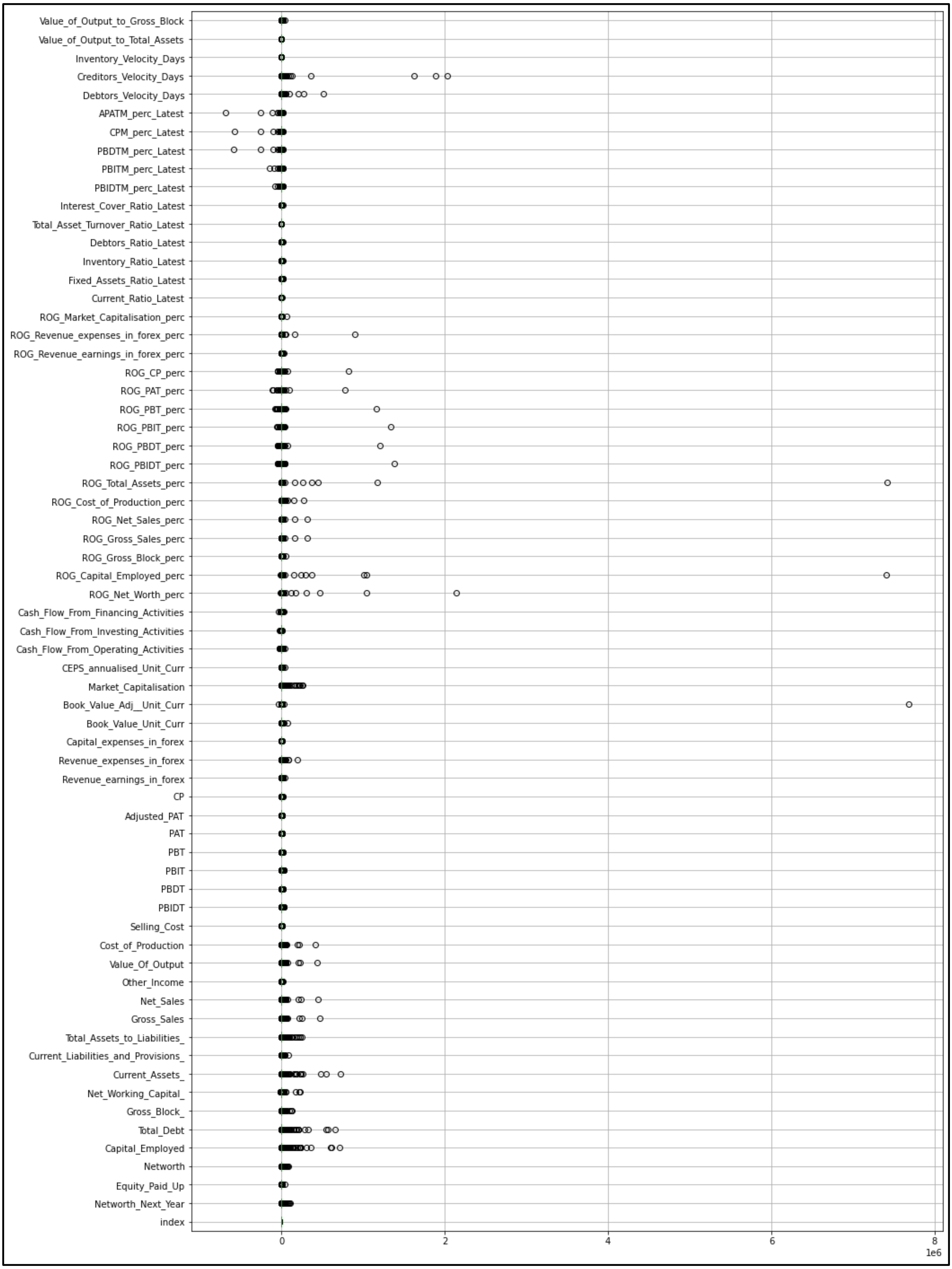
Now we calculate the proportion of outliers in the total data

```
Total outliers- 40779
Total cells- 226070
% of outliers in data- 0.1803821825098421
```

So we do the outlier treatment by converting the outliers to the values of upper and lower limits or whiskers.

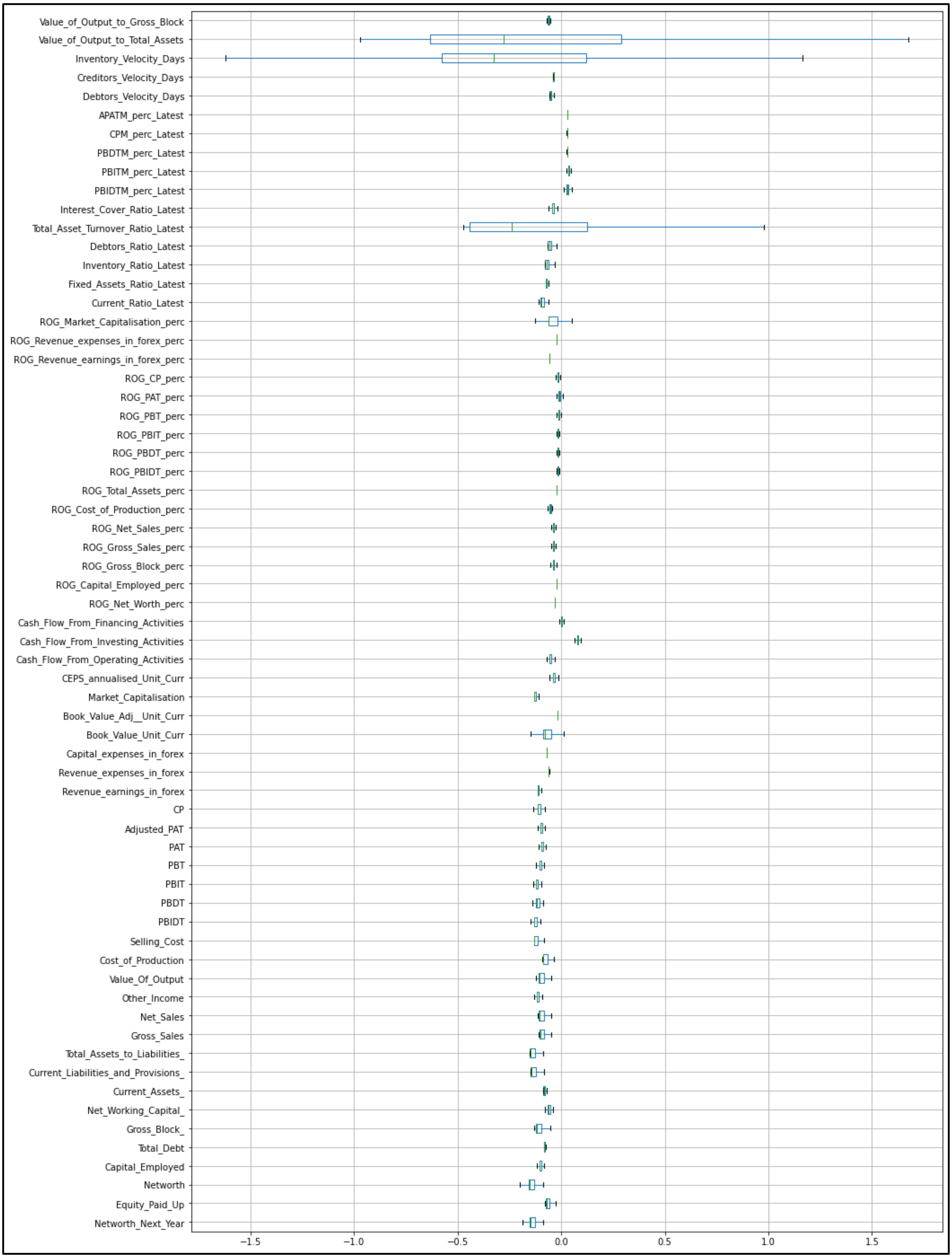
EDA

Outliers before Treatment : -

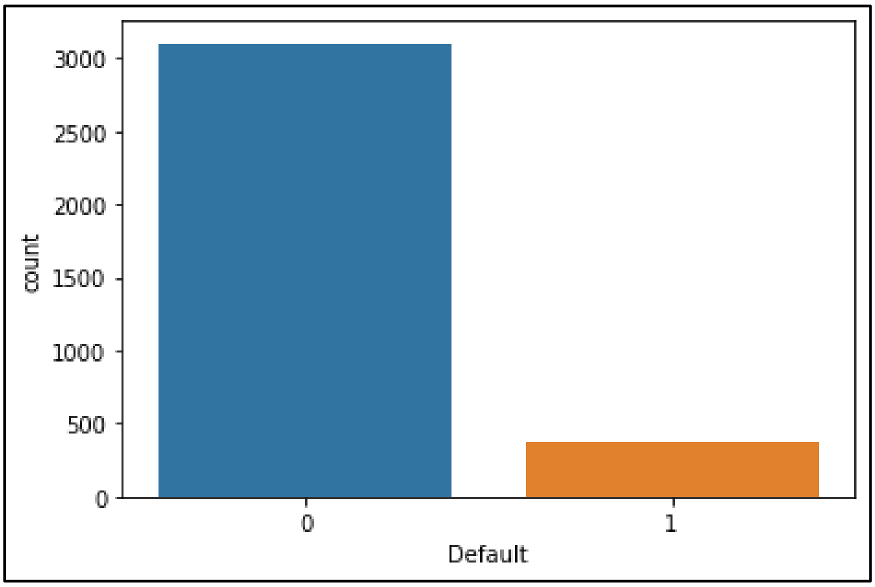


We can see that almost all the variables have outliers.

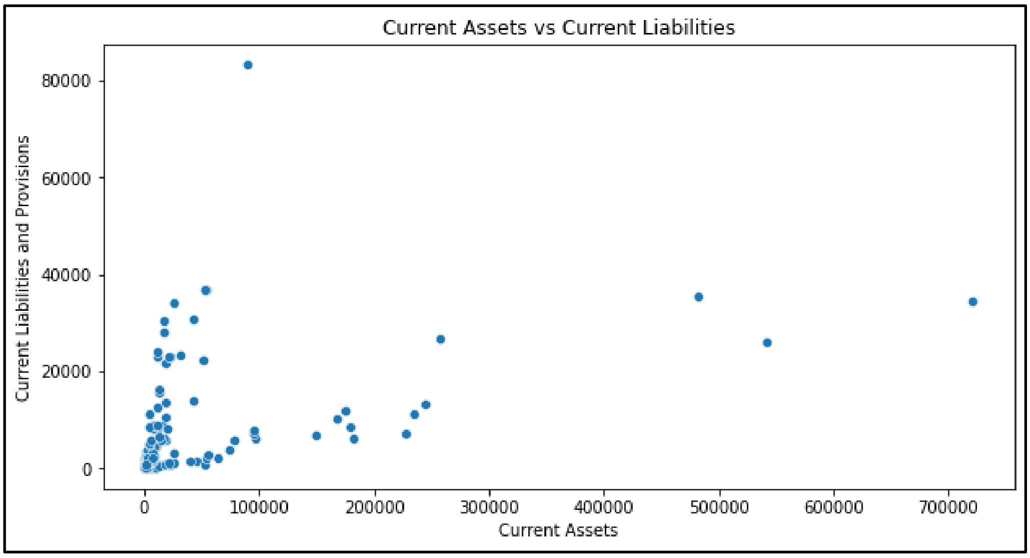
Post Outlier Treatment-



Now, we have removed the outliers by setting those values as upper or lower whiskers. We could also use another method i.e we could set all the outliers as NaN and impute those values using KNNImputer which used KNN clustering method.

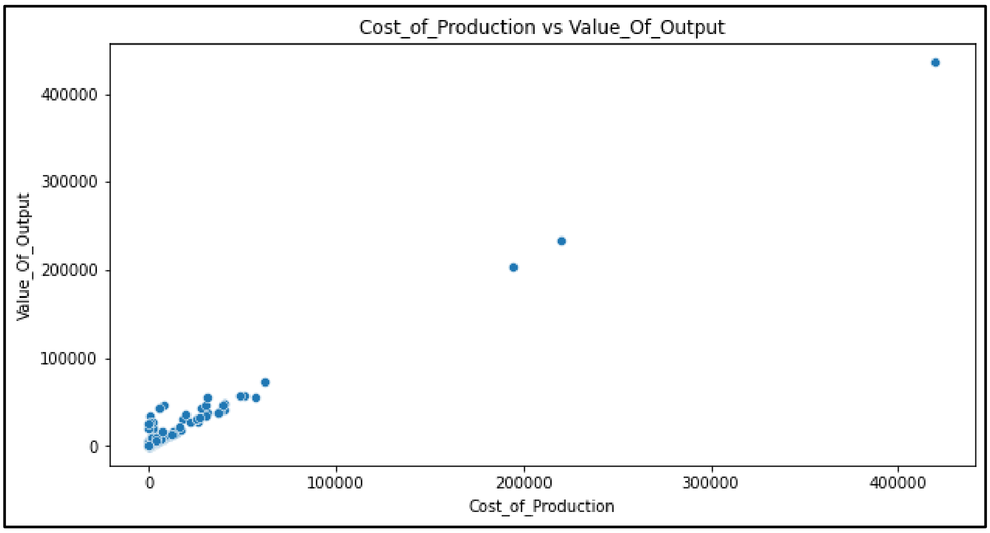


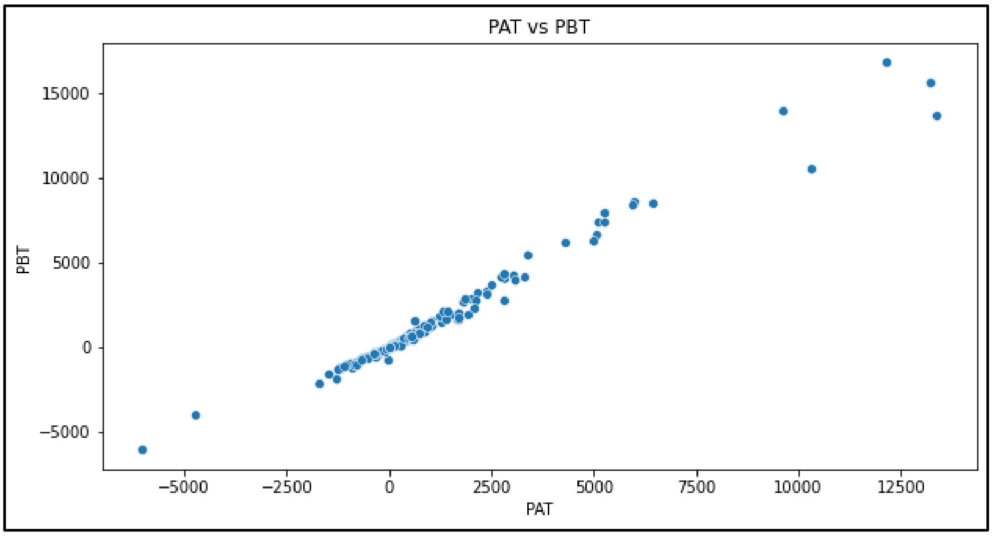
As we can see here, the target variable isn't balanced, most of them are non-defaulters.

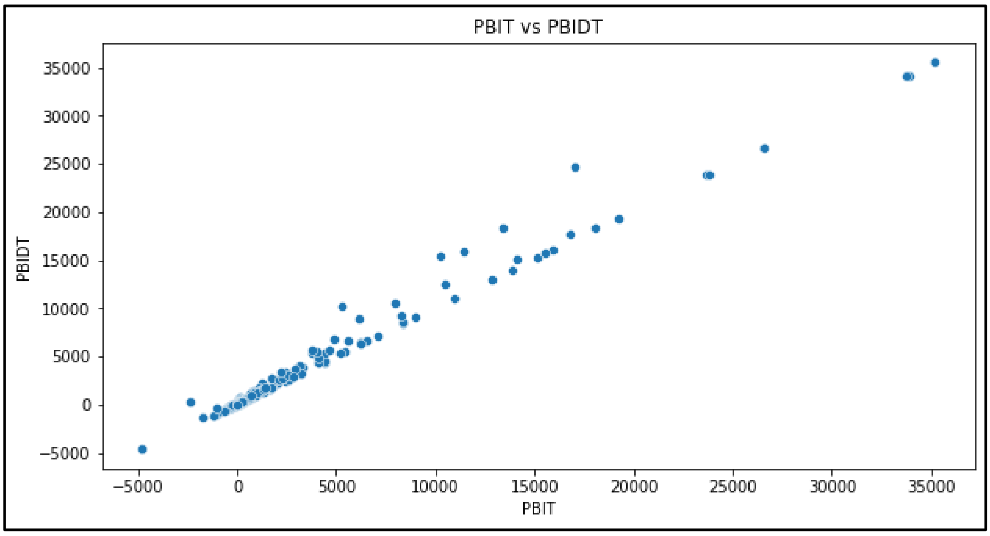


We plot Current Assets vs Current Liabilities and see that most of the companies are small companies and have low assets and lower liabilities and are clustered together.

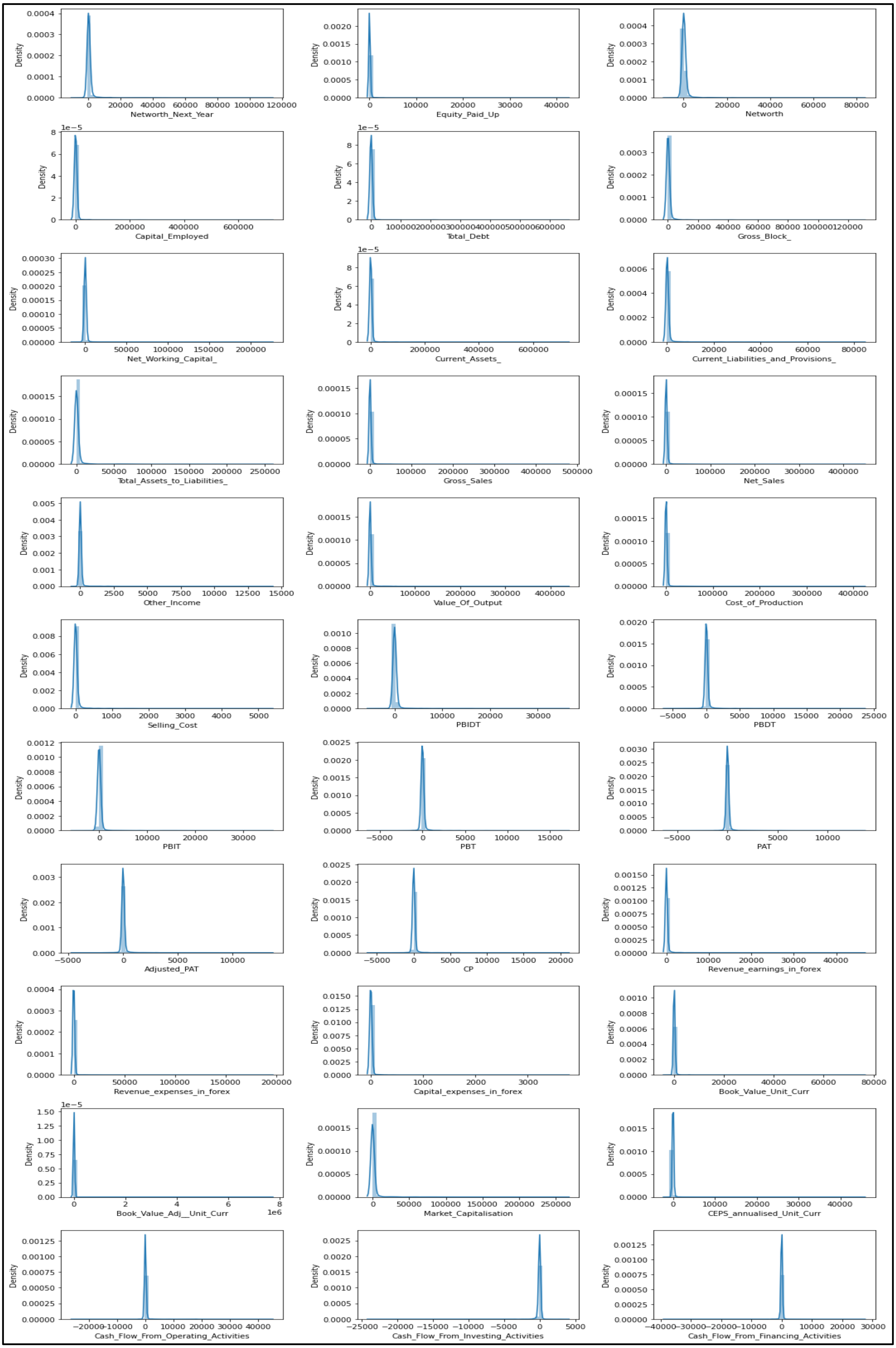
Below, we can see that value of output and cost of production are linearly related and we can see that with increase in Cost of production there is increase in Value of Output.







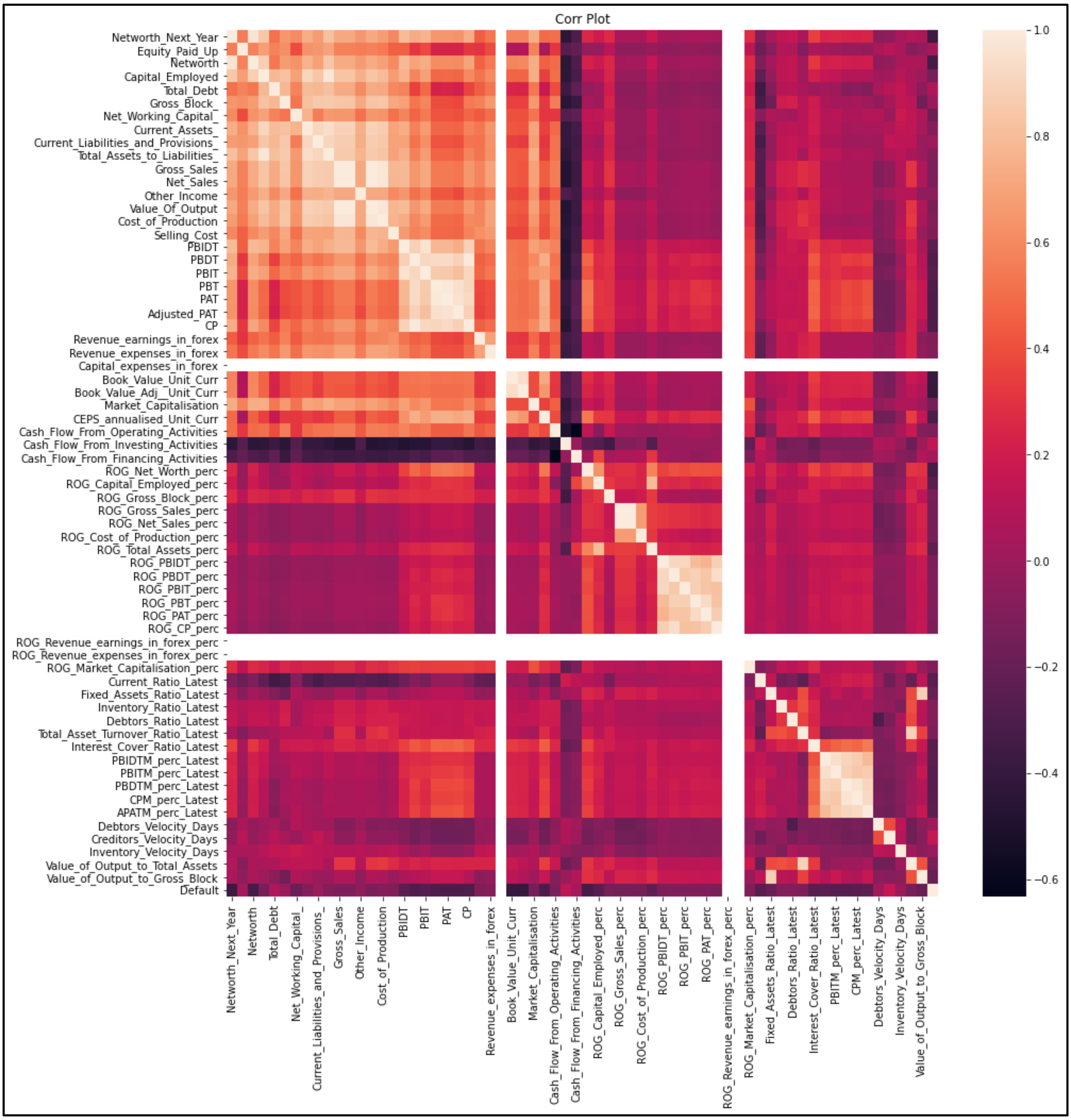
Similarly, we can see that PAT and PBT & PBIT and PBIDT are highly linearly correlated which we can also observe in the correlation plot with annotations in the heatmap.


Density
Networth_Next_Year
Equity_Paid_Up
Networth
Capital_Employed
Total_Debt
Gross_Block_
Net_Working_Capital_
Current_Assets_
Current_Liabilities_and_Provisions_
Total_Assets_to_Liabilities_
Gross_Sales
Net_Sales
Other_Income
Value_Of_Output
Cost_of_Production
Selling_Cost
PBIDT
PBDT
PBIT
PBT
PAT
Adjusted_PAT
CP
Revenue_earnings_in_forex
Revenue_expenses_in_forex
Capital_expenses_in_forex
Book_Value_Unit_Curr
Book_Value_Adj__Unit_Curr
Market_Capitalisation
CEPS_annualised_Unit_Curr
Cash_Flow_From_Operating_Activities
Cash_Flow_From_Investing_Activities
Cash_Flow_From_Financing_Activities

The above plotted are distributions for various variables using distplot from seaborn library. We can see almost all the variables are right skewed.

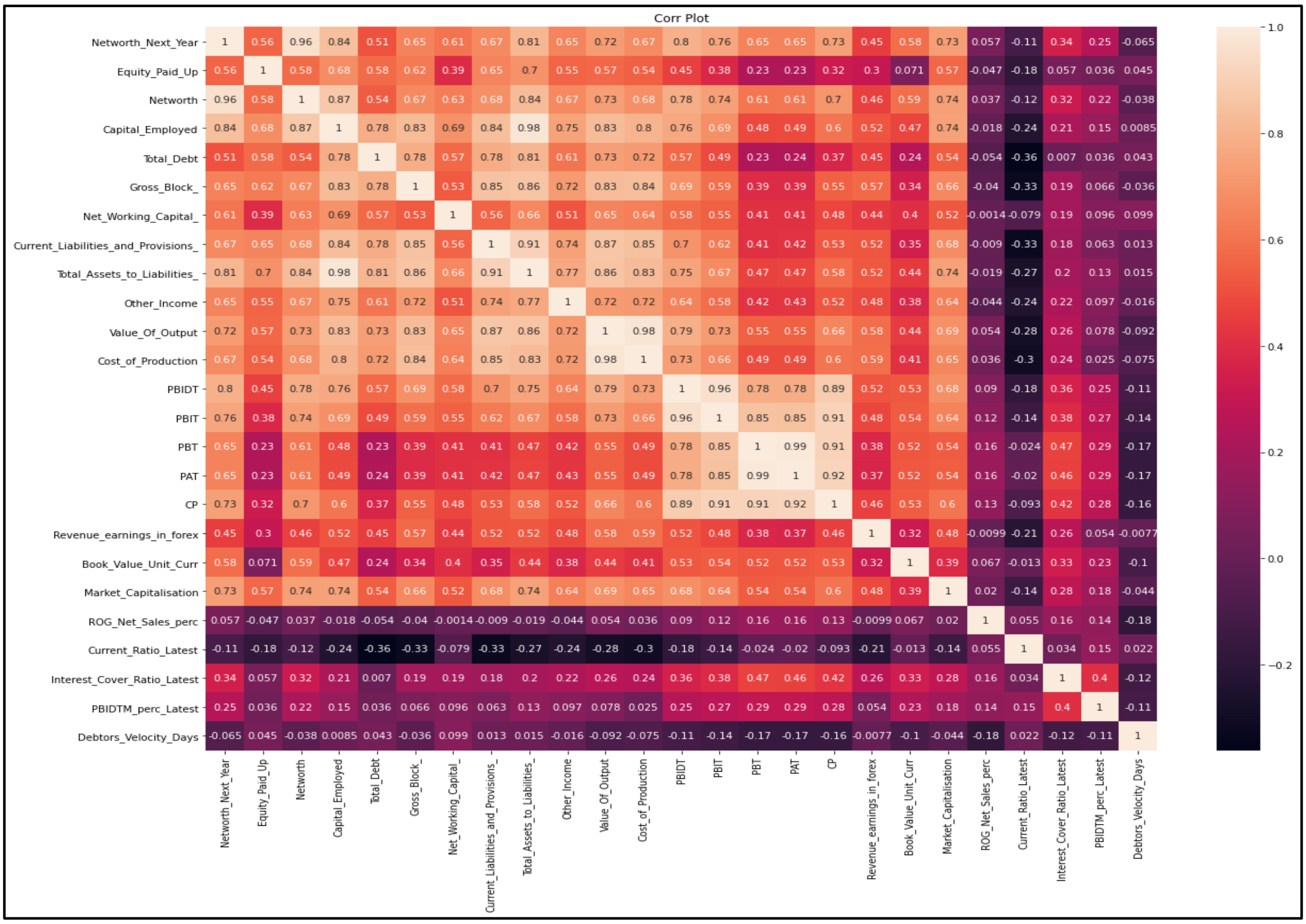
Correlation-

We find the correlation among variables and plot them using heatmap

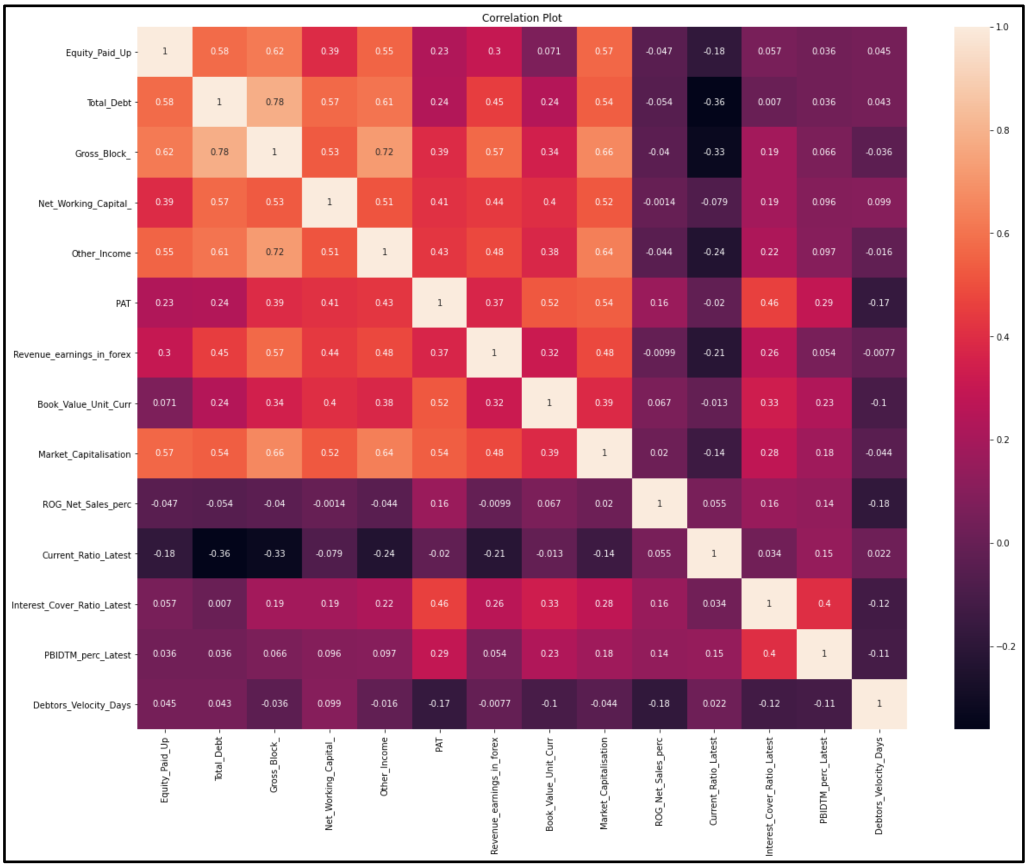


Since some variables are highly correlated, we drop them else it will make the model less credible and give importance to variables that shouldn't have much impact.

So we do the feature selection using the RFE from sklearn.feature_selection and then rank the features, we do this since there are 60 odd features in the dataset, hence we select 25 features that contribute the most to the model and then do the ranking on those features and select the features that Rank 1 and plot a correlation and heatmap on it.



Now, we can see that there are some features that are highly correlated like - **Capital_Employed,Value_Of_Output,Cost_of_Production,Networth,Networth_Next_Year,PBIDT,CP,PBT,PBIT,Current_Liabilities_and_Provisions_,Total_Assets_to_Liabilities_.**So we drop them from the data and again plot the correlation and check the heatmap



TRAIN TEST SPLIT

We split the data into X and Y as dependant and independent variables and then 33:67 as test:train and use random state as 42 as specified in the problem statement and then use this train and test data to build our model. We do this using the train_test_split from sklearn.model_selection

We then check the distribution of default in train and test sets-

```
0    2092
1     238
Name: Default, dtype: int64
```

Train set

Test set

```
0    1009
1     139
Name: Default, dtype: int64
```

MODEL BUILDING

Model 1

Now we use the statsmodels.formula.api to do logistic regression. Logistic Regression is used for classification problems.

For this we need to make formula based on which the model will be built.

So the parameters or columns that we have selected are after removing the unwanted or non-meaningful columns, columns with high correlations and keep the top 25 columns that have the highest rank.

```
'Default ~ Equity_Paid_Up + Total_Debt + Gross_Block_ + Net_Working_Capital_ + Other_Income + PAT + Revenue_earnings_in_fore
x + Book_Value_Unit_Curr + Market_Capitalisation + ROG_Net_Sales_perc + Current_Ratio_Latest + Interest_Cover_Ratio_Latest +
PBIDTM_perc_Latest + Debtors_Velocity_Days'
```

We use the logit() on the formula and train data and then fit it into the model using bfgs method

```
Optimization terminated successfully.
         Current function value: 0.119151
         Iterations: 149
         Function evaluations: 150
         Gradient evaluations: 150
```

So, after 149 iterations, a model is built that seems to be the most optimized model based on the given inputs.

MODEL SUMMARY

Model 1

Logit Regression Results

Dep. Variable:	Default	**No. Observations:**	2330
Model:	Logit	**Df Residuals:**	2315
Method:	MLE	**Df Model:**	14
Date:	Thu, 22 Jul 2021	**Pseudo R-squ.:**	0.6387
Time:	17:47:18	**Log-Likelihood:**	-277.62
converged:	True	**LL-Null:**	-768.37
Covariance Type:	nonrobust	**LLR p-value:**	1.458e-200

	coef	**std err**	**z**	**P>|z|**	**[0.025**	**0.975]**
Intercept	-33.8911	8.039	-4.216	0.000	-49.648	-18.134
Equity_Paid_Up	-19.7044	9.378	-2.101	0.036	-38.085	-1.324
Total_Debt	44.3735	96.238	0.461	0.645	-144.249	232.996
Gross_Block_	16.6206	13.429	1.238	0.216	-9.701	42.942
Net_Working_Capital_	32.5319	22.791	1.427	0.153	-12.138	77.201
Other_Income	-4.6053	21.855	-0.211	0.833	-47.440	38.230
PAT	-61.9970	20.993	-2.953	0.003	-103.142	-20.852
Revenue_earnings_in_forex	-11.9821	30.658	-0.391	0.696	-72.071	48.107
Book_Value_Unit_Curr	-177.7893	16.572	-10.728	0.000	-210.270	-145.309
Market_Capitalisation	-41.5417	29.003	-1.432	0.152	-98.387	15.304
ROG_Net_Sales_perc	-43.4812	21.728	-2.001	0.045	-86.067	-0.896
Current_Ratio_Latest	-61.0861	10.769	-5.672	0.000	-82.194	-39.978
Interest_Cover_Ratio_Latest	-40.3152	17.424	-2.314	0.021	-74.466	-6.164
PBIDTM_perc_Latest	-52.0479	13.637	-3.817	0.000	-78.775	-25.320
Debtors_Velocity_Days	-25.4680	14.014	-1.817	0.069	-52.935	1.999

Possibly complete quasi-separation: A fraction 0.32 of observations can be perfectly predicted. This might indicate that there is complete quasi-separation. In this case some parameters will not be identified.

MODEL VALIDATION

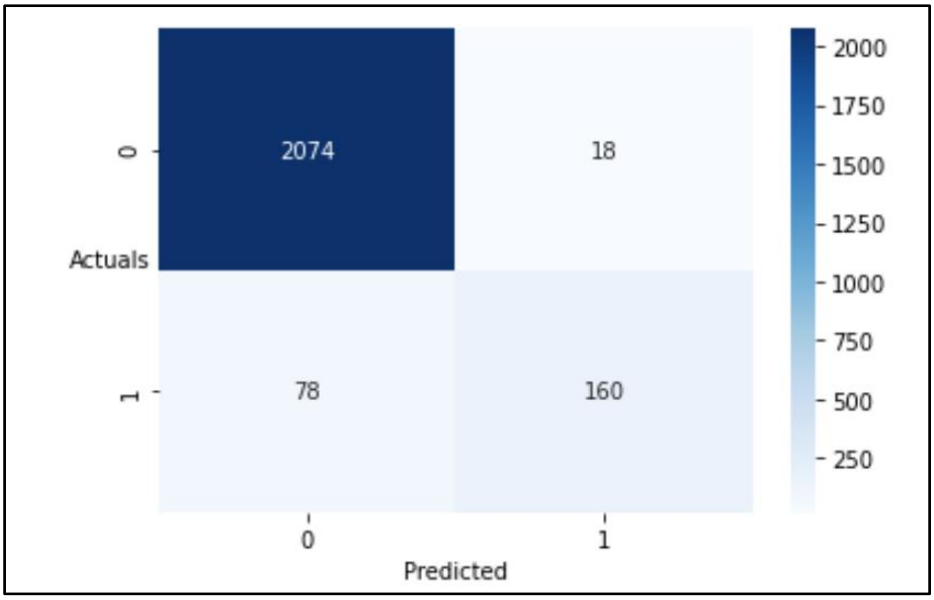
Model 1

Train Set-

```
0    0.74
1    0.00
2    0.00
3    0.01
4    0.00
dtype: float64
```

The above is the prediction for the first 5 rows of train data.

Now we make a function where we have set the threshold as 0.5 and the values above 0.5 will be considered as 1 and the values less than 0.5 as 0 as the prediction output.



The above image is the confusion matrix representation on the train data where 2074+160 values are predicted correctly and 18 FP and 78 TN values are predicted incorrectly.

	precision	recall	f1-score	support
0	0.96	0.99	0.98	2092
1	0.90	0.67	0.77	238
accuracy			0.96	2330
macro avg	0.93	0.83	0.87	2330
weighted avg	0.96	0.96	0.96	2330

This image is the classification report on the train data and shows that we have a reasonably good model with accuracy of around 96% and acceptable precision and recall for the defaulters.

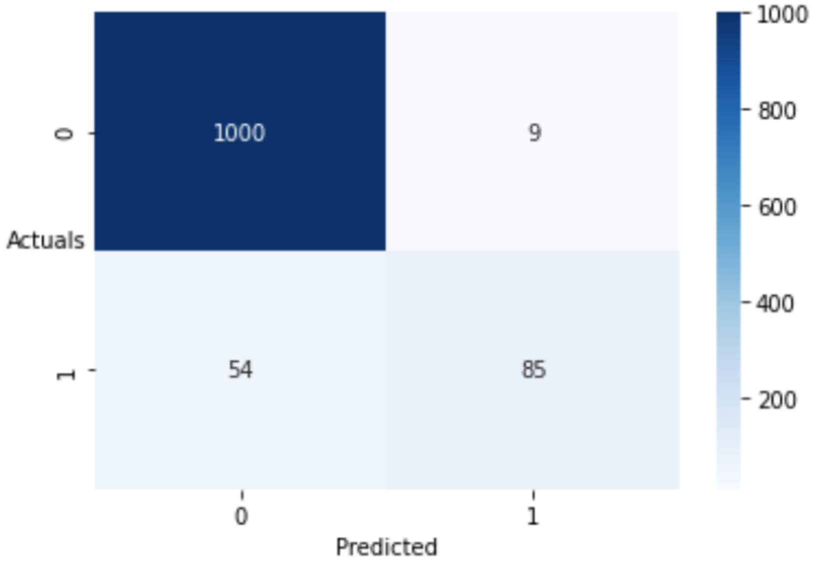
Model 1

Test Set-

```
0    0.00
1    0.06
2    0.67
3    0.00
4    0.00
dtype: float64
```

The above is the prediction for the first 5 rows of test data.

Now we make a function where we have set the threshold as 0.5 and the values above 0.5 will be considered as 1 and the values less than 0.5 as 0 as the prediction output.



The above image is the confusion matrix representation on the train data where 1000+85 values are predicted correctly and 9 FP and 54 TN values are predicted incorrectly.

```
              precision    recall  f1-score   support

           0       0.95      0.99      0.97      1009
           1       0.90      0.61      0.73       139

    accuracy                           0.95      1148
   macro avg       0.93      0.80      0.85      1148
weighted avg       0.94      0.95      0.94      1148
```

This image is the classification report on the train data and shows that we have a reasonably good model with accuracy of around 95% and acceptable precision and recall for the defaulters.

Model 2

Formula used for building model –

Here, we further remove the columns that have a higher p-value that we observed while doing model summary for Model 1 and get even optimized results.

```
'Default ~ Equity_Paid_Up + PAT + Book_Value_Unit_Curr + ROG_Net_Sales_perc + Current_Ratio_Latest + Interest_Cover_Ratio_La
test + PBIDTM_perc_Latest + Debtors_Velocity_Days '
```

```
Optimization terminated successfully.
         Current function value: 0.121764
         Iterations: 105
         Function evaluations: 107
         Gradient evaluations: 107
```

Logit Regression Results

Dep. Variable:	Default	**No. Observations:**	2330
Model:	Logit	**Df Residuals:**	2321
Method:	MLE	**Df Model:**	8
Date:	Thu, 22 Jul 2021	**Pseudo R-squ.:**	0.6308
Time:	23:23:08	**Log-Likelihood:**	-283.71
converged:	True	**LL-Null:**	-768.37
Covariance Type:	nonrobust	**LLR p-value:**	6.245e-204

	coef	std err	z	P>|z|	[0.025	0.975]
Intercept	-34.4460	3.065	-11.240	0.000	-40.452	-28.440
Equity_Paid_Up	-11.2457	7.408	-1.518	0.129	-25.765	3.273
PAT	-73.9977	20.350	-3.636	0.000	-113.883	-34.112
Book_Value_Unit_Curr	-172.2630	15.538	-11.087	0.000	-202.716	-141.810
ROG_Net_Sales_perc	-39.4143	21.039	-1.873	0.061	-80.651	1.822
Current_Ratio_Latest	-59.7016	10.041	-5.946	0.000	-79.381	-40.022
Interest_Cover_Ratio_Latest	-40.5006	16.920	-2.394	0.017	-73.664	-7.338
PBIDTM_perc_Latest	-47.6611	13.079	-3.644	0.000	-73.296	-22.026
Debtors_Velocity_Days	-30.2354	13.765	-2.196	0.028	-57.215	-3.256

Possibly complete quasi-separation: A fraction 0.32 of observations can be perfectly predicted. This might indicate that there is complete quasi-separation. In this case some parameters will not be identified.

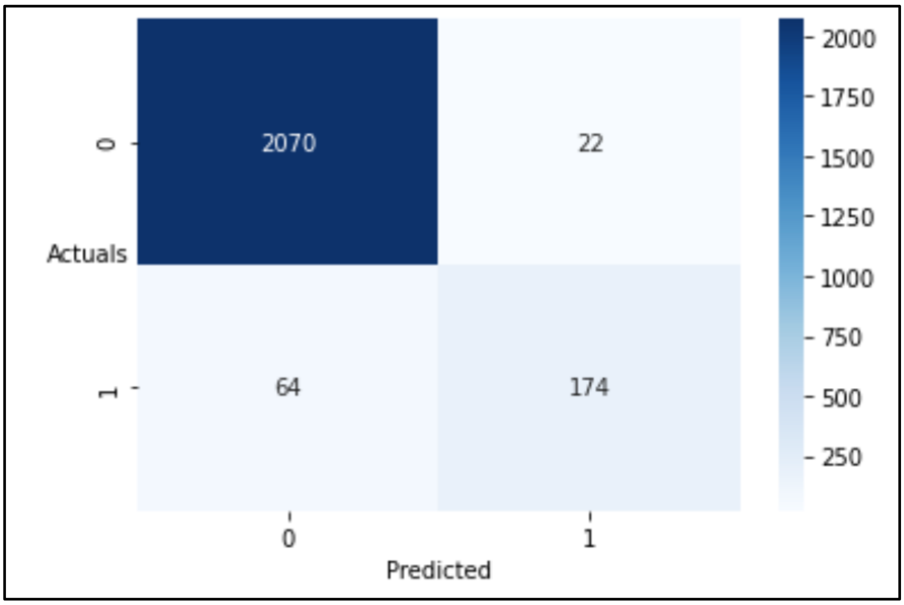
Model 2

Train Set-

```
0    0.74
1    0.00
2    0.00
3    0.01
4    0.00
dtype: float64
```

The above is the prediction for the first 5 rows of train data.

Now we make a function where we have set the threshold as 0.45 and the values above 0.45 will be considered as 1 and the values less than 0.45 as 0 as the prediction output.



The above image is the confusion matrix representation on the train data where 2070+174 values are predicted correctly and 22 FP and 64 TN values are predicted incorrectly.

	precision	recall	f1-score	support
0	0.97	0.99	0.98	2092
1	0.89	0.73	0.80	238
accuracy			0.96	2330
macro avg	0.93	0.86	0.89	2330
weighted avg	0.96	0.96	0.96	2330

This image is the classification report on the train data and shows that we have a reasonably good model with accuracy of around 96% and acceptable precision and recall for the defaulters. But in this model, our recall rate has increased than the previous model.

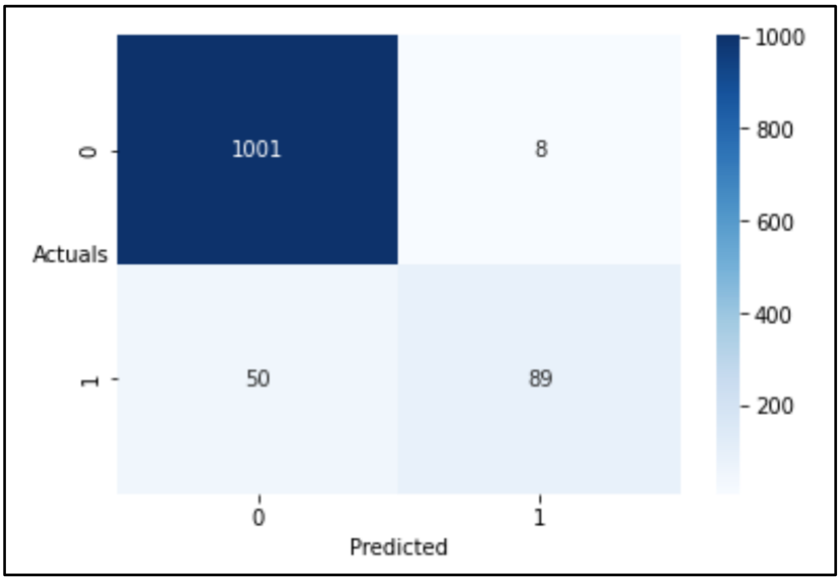
Model 2

Test Set-

```
0    0.00
1    0.06
2    0.67
3    0.00
4    0.00
dtype: float64
```

The above is the prediction for the first 5 rows of test data.

Now we make a function where we have set the threshold as 0.45 and the values above 0.45 will be considered as 1 and the values less than 0.45 as 0 as the prediction output.



The above image is the confusion matrix representation on the train data where 1001+89 values are predicted correctly and 8 FP and 50 TN values are predicted incorrectly.

```
              precision    recall  f1-score   support

           0       0.95      0.99      0.97      1009
           1       0.92      0.64      0.75       139

    accuracy                           0.95      1148
   macro avg       0.93      0.82      0.86      1148
weighted avg       0.95      0.95      0.95      1148
```

This image is the classification report on the train data and shows that we have a reasonably good model with accuracy of around 95% and acceptable precision and recall for the defaulters. Our recall has increased compared to model 1