greatlearning

Learning for Life

FRA BUSINESS REPORT

PREPARED BY - PRATHEEK U

Table of Contents

List of Figures

List of Figures

List of Tables

List of Tables

Problem Statement : FRA

You are requested to create an Indian credit risk(default) model, using the data provided in the spreadsheet.

Hints:

- Dependent variable - We need to create a default variable that should take the value of 1 when net worth next year is negative & 0 when net worth next year is positive.
- Validation Dataset - We need to build the model on a train dataset and check the model performance measures on the validation dataset.

Executive Summary

We need to build Indian Credit Risk (Default) Model on the basis of given data in spreadsheet . The dataset consists of **4256 rows with 51 variables.** Based on the different attributes of the dataset we have to build a model, to predict which company is going to be Defaulter or Non-Defaulter on the basis of the given information. We will create a Machine Learning Model that will help in predicting overall status of company i.e. Defaulter and Non Defaulter.

Introduction

The purpose of this whole exercise is to explore the dataset. Do the exploratory data analysis , visualization & apply various supervised learning algorithms i.e. **Logistics Regression , Random Forest** to predict which company will be defaulter or Non defaulter on the basis of the given information, to create a robust machine learning model that will help in predicting the defaulter or Non defaulter status of companies. Explore the dataset using central tendency and other parameters. The data consists of 4256 entries with 51 unique attributes. Analyse the different attributes of the dataset which can help in predicting which is defaulter or Non defaulter company on the basis of the given information.This assignment should help us to **reduce credit risk** by **predicting defaulter or Non defaulter status of companies.**

Checking the Records of the Dataset :

Head of the Dataset - First 10 Records of the Dataset.
Tail of the Dataset - Last 10 Records of the Dataset.

Num	Networth Next Year	Total assets	Net worth	Total income	Change in stock	Total expenses	Profit after tax	PBDITA	PBT	Cash profit	PBDITA as % of total income	PBT as % of total income	PAT as % of total income	Cash profit as % of total income
1	395.3	827.6	336.5	534.1	13.5	508.7	38.9	124.4	64.6	95.2	23.29	12.10	7.28	17.82
2	36.2	67.7	24.3	137.9	-3.7	131.0	3.2	5.5	1.0	3.8	3.99	0.73	2.32	2.76
3	84.0	238.4	78.9	331.2	-18.1	309.2	3.9	25.8	10.5	9.4	7.79	3.17	1.18	2.84
4	2041.4	6883.5	1443.3	8448.5	212.2	8482.4	178.3	418.4	185.1	178.0	4.95	2.19	2.11	2.11
5	41.8	90.9	47.0	388.6	3.4	392.7	-0.7	7.2	-0.6	3.9	1.85	-0.15	-0.18	1.00
6	291.5	573.8	238.6	582.6	31.0	565.3	48.3	110.1	68.5	82.6	18.90	11.76	8.29	14.18
7	93.3	329.9	92.5	17.3	0.1	16.0	1.4	14.0	6.4	6.8	80.92	36.99	8.09	39.31
8	985.1	5435.2	1013.6	1921.2	76.6	2047.1	-49.3	248.1	-49.3	157.9	12.91	-2.57	-2.57	8.22
9	188.6	526.1	117.2	946.1	21.9	919.3	48.7	108.6	71.2	66.5	11.48	7.53	5.15	7.03
10	229.6	280.9	95.9	1272.0	15.7	1280.0	7.7	31.8	12.5	14.8	2.50	0.98	0.61	1.16

Num	Networth Next Year	Total assets	Net worth	Total income	Change in stock	Total expenses	Profit after tax	PBDITA	PBT	Cash profit	PBDITA as % of total income	PBT as % of total income	PAT as % of total income	Cash profit as % of total income
4247	135.5	651.2	118.4	961.2	6.2	939.6	27.8	78.0	28.7	60.2	8.11	2.99	2.89	6.26
4248	47.0	100.4	43.2	273.6	1.3	271.3	3.6	13.6	6.0	7.1	4.97	2.19	1.32	2.60
4249	81.4	225.8	70.8	435.9	23.5	449.5	9.9	25.9	15.3	18.9	5.94	3.51	2.27	4.34
4250	383.1	1591.9	375.6	3717.2	-29.7	3681.5	6.0	81.1	13.7	5.7	2.18	0.37	0.16	0.15
4251	336.5	455.2	197.8	199.2	NaN	193.3	5.9	59.1	6.7	35.9	29.67	3.36	2.96	18.02
4252	0.2	0.4	0.2	NaN	NaN	NaN	NaN	NaN	NaN	NaN	0.00	0.00	0.00	0.00
4253	93.3	159.6	86.7	172.9	0.1	169.7	3.3	18.4	3.7	12.6	10.64	2.14	1.91	7.29
4254	932.2	833.8	664.6	2314.7	32.1	2151.6	195.2	348.4	303.0	219.5	15.05	13.09	8.43	9.48
4255	64.6	95.0	48.5	110.5	4.6	113.5	1.6	9.7	2.6	6.7	8.78	2.35	1.45	6.06
4256	0.0	384.6	111.3	345.8	11.3	341.7	15.4	57.6	20.7	34.8	16.66	5.99	4.45	10.06

Tab:1 Records of the Dataset Head & Tail

Note:

We are going to drop the column Num as this column didn't contribute for analysis and model building excercise , because Num for each entry in the dataset is unique hence it is useless for the model.That's why we decided to drop this column.

Note:

As we saw in the data set we don't have the **target column (y)** , so we are advised to drive the target column from the existing independent variable i.e. **Net worth Next Year** in such a way that default variable should take the value of 1 when net worth next year is negative & 0 when net worth next year is positive.

After creating the target variable we are going to drop the Net worth Next Year attribute because dependent variable is drive from this variable and if we are not dropping this variable then it will make our model bias and other variables will not perform well in predicting the dependent variable.

Checking the Records of the Dataset After Creating Target Column and Dropping Num and Net worth Next Year column :

Head of the Dataset - First 5 Records of the Dataset.

	Total assets	Net worth	Total income	Change in stock	Total expenses	Profit after tax	PBDITA	PBT	Cash profit	PBDITA as % of total income	PBT as % of total income	PAT as % of total income	Cash profit as % of total income	PAT as % of net worth
0	827.6	336.5	534.1	13.5	508.7	38.9	124.4	64.6	95.2	23.29	12.10	7.28	17.82	12.27
1	67.7	24.3	137.9	-3.7	131.0	3.2	5.5	1.0	3.8	3.99	0.73	2.32	2.76	0.00
2	238.4	78.9	331.2	-18.1	309.2	3.9	25.8	10.5	9.4	7.79	3.17	1.18	2.84	5.07
3	6883.5	1443.3	8448.5	212.2	8482.4	178.3	418.4	185.1	178.0	4.95	2.19	2.11	2.11	13.17
4	90.9	47.0	388.6	3.4	392.7	-0.7	7.2	-0.6	3.9	1.85	-0.15	-0.18	1.00	-1.48

Cash to average cost of sales per day	Creditors turnover	Debtors turnover	Finished goods turnover	WIP turnover	Raw material turnover	Shares outstanding	Equity face value	EPS	Adjusted EPS	Total liabilities	PE on BSE	Default
5.41	11.60	5.65	3.99	3.37	14.87	8760056.0	10.0	4.44	4.44	827.6	NaN	0
1.62	NaN	NaN	NaN	NaN	NaN	NaN	NaN	0.00	0.00	67.7	NaN	0
26.42	2.24	2.51	17.67	8.76	8.35	NaN	NaN	0.00	0.00	238.4	NaN	0
15.93	3.48	1.91	18.14	18.62	11.11	10000000.0	10.0	17.60	17.60	6883.5	NaN	0
0.85	21.67	68.00	45.87	28.67	19.93	107315.0	100.0	-6.52	-6.52	90.9	NaN	0

Tab: 2 Records of the Dataset After Creating Target Column and Dropping Num and Net worth Next Year column

Data Dictionary of the Problem Statement

Variable Name	Description
Networth Next Year	Net worth of the customer in next year
Total assets	Total assets of customer
Net worth	Net worth of the customer of present year
Total income	Total income of the customer
Change in stock	difference between value of current stock and the value of stock in last trading day
Total expenses	Total expense done by customer
Profit after tax	Profit after tax deduction
PBDITA	Profit before depreciation, income tax and amortization
PBT	Profit before tax deduction
Cash profit	Total Cash profit
PBDITA as % of total income	PBDITA / Total income
PBT as % of total income	PBT / Total income
PAT as % of total income	PAT / Total income
Cash profit as % of total income	Cash Profit / Total income
PAT as % of net worth	PAT / Net worth
Sales	Sales done by customer
Income from financial services	Income from financial services
Other income	Income from other sources
Total capital	Total capital of the customer
Reserves and funds	Total reserves and funds of the customer

Deposits (accepted by commercial banks)	All blank values
Borrowings	Total amount borrowed by customer
Current liabilities & provisions	current liabilities of the customer
Deferred tax liability	Future income tax customer will pay because of the current transaction
Shareholders funds	Amount of equity in a company, which is belong to shareholder
Cumulative retained profits	Total cumulative profit retained by customer
Capital employed	Current asset minus current liabilities
TOL/TNW	Total liabilities of the customer divided by Total net worth
Total term liabilities / tangible net worth	Short + long term liabilities divided by tangible net worth
Contingent liabilities / Net worth (%)	Contingent liabilities / Net worth
Contingent liabilities	Liabilities because of uncertain events
Net fixed assets	purchase price of all fixed assets
Investments	Total invested amount
Current assets	Assets that are expected to be converted to cash within a year
Net working capital	Difference of current liabilities and current assets
Quick ratio (times)	Total cash divided by current liabilities
Current ratio (times)	Current assets divided by current liabilities
Debt to equity ratio (times)	Total liabilities divided by its shareholder equity
Cash to current liabilities (times)	Total liquid cash divided by current liabilities
Cash to average cost of sales per day	Total cash divided by average cost of the sales
Creditors turnover	Net credit purchase divided to average trade creditors
Debtors turnover	Net credit sales divided by average accounts receivable
Finished goods turnover	Annual sales divided by average inventory
WIP turnover	The cost of goods sold for a period divided by the average inventory for that period
Raw material turnover	Cost of goods sold is divided by the average inventory for the same period
Shares outstanding	Number of issued shares minus the number of share held in the company
Equity face value	cost of the equity at the time of issuing
EPS	Net income divided by total number of outstanding share
Adjusted EPS	Adjusted net earning divided by the weighted average number of common share outstanding on a diluted basis during the plan year
Total liabilities	Sum of all type of liabilities
PE on BSE	Company current stock price divided by its earning per share

Tab: 3 Data Dictionary of the Probelm Statement

Checking the Summary of the Dataset.

	count	mean	std	min	25%	50%	75%	max
Total_assets	4256.0	3.573617e+03	3.007444e+04	1.000000e-01	91.300	315.500	1.120800e+03	1.176509e+06
Net_worth	4256.0	1.351950e+03	1.296131e+04	0.000000e+00	31.475	104.800	3.898500e+02	6.131516e+05
Total_income	4025.0	4.688190e+03	5.391895e+04	0.000000e+00	107.100	455.100	1.485000e+03	2.442828e+06
Change_in_stock	3706.0	4.370248e+01	4.369150e+02	-3.029400e+03	-1.800	1.600	1.840000e+01	1.418550e+04
Total_expenses	4091.0	4.356301e+03	5.139809e+04	-1.000000e-01	96.800	426.800	1.395700e+03	2.366035e+06
Profit_after_tax	4102.0	2.950506e+02	3.079902e+03	-3.908300e+03	0.500	9.000	5.330000e+01	1.194391e+05
PBDITA	4102.0	6.059406e+02	5.646231e+03	-4.407000e+02	6.925	36.900	1.587000e+02	2.085765e+05
PBT	4102.0	4.102590e+02	4.217415e+03	-3.894800e+03	0.800	12.600	7.417500e+01	1.452926e+05
Cash_profit	4102.0	4.082675e+02	4.143926e+03	-2.245700e+03	2.900	19.400	9.625000e+01	1.769118e+05
'c_of_total_income	4177.0	3.179892e+00	1.722566e+02	-6.400000e+03	4.970	9.680	1.647000e+01	1.000000e+02
'c_of_total_income	4177.0	-1.819683e+01	4.199111e+02	-2.134000e+04	0.560	3.340	8.940000e+00	1.000000e+02
'c_of_total_income	4177.0	-2.003367e+01	4.235762e+02	-2.134000e+04	0.350	2.370	6.420000e+00	1.500000e+02
'c_of_total_income	4177.0	-9.021278e+00	2.999574e+02	-1.502000e+04	2.000	5.660	1.073000e+01	1.000000e+02
perc_of_net_worth	4256.0	1.016786e+01	6.153240e+01	-7.487200e+02	0.000	8.040	2.020250e+01	2.466670e+03
Sales	3951.0	4.645685e+03	5.308090e+04	1.000000e-01	113.350	468.600	1.481200e+03	2.384984e+06
m_fincial_services	3145.0	8.136006e+01	1.042759e+03	0.000000e+00	0.500	1.900	9.800000e+00	5.193820e+04
Other_income	2700.0	5.595289e+01	1.178415e+03	0.000000e+00	0.400	1.500	6.200000e+00	4.285670e+04

	count	mean	std	min	25%	50%	75%	max
Total_capital	4251.0	2.245577e+02	1.684951e+03	1.000000e-01	13.200	42.600	1.031500e+02	7.827320e+04
serves_and_funds	4158.0	1.210562e+03	1.281623e+04	-6.525900e+03	5.300	55.150	2.825250e+02	6.251378e+05
Borrowings	3825.0	1.176248e+03	8.581249e+03	1.000000e-01	24.400	99.800	3.583000e+02	2.782573e+05
es_and_provisions	4146.0	9.606314e+02	9.140536e+03	1.000000e-01	17.500	70.300	2.659250e+02	3.522403e+05
ferred_tax_liability	2887.0	2.344951e+02	2.106253e+03	1.000000e-01	3.200	13.500	5.130000e+01	7.279660e+04
hareholders_funds	4256.0	1.376487e+03	1.301069e+04	0.000000e+00	32.300	107.600	4.089000e+02	6.131516e+05
ve_retained_profits	4211.0	9.371820e+02	9.853096e+03	-6.534300e+03	1.100	37.400	2.062000e+02	3.901338e+05
Capital_employed	4256.0	2.433618e+03	2.049640e+04	0.000000e+00	61.300	221.200	7.903000e+02	8.914089e+05
TOL_to_TNW	4256.0	4.025343e+00	2.087909e+01	-3.504800e+02	0.600	1.420	2.830000e+00	4.730000e+02
angible_net_worth	4256.0	1.854288e+00	1.587506e+01	-3.256000e+02	0.050	0.345	1.000000e+00	4.560000e+02
__Net_worth_perc	4256.0	5.570750e+01	3.691657e+02	0.000000e+00	0.000	5.360	3.101250e+01	1.470427e+04
ntingent_liabilities	2854.0	9.485522e+02	1.205674e+04	1.000000e-01	6.000	37.850	1.953250e+02	5.595068e+05
Net_fixed_assets	4124.0	1.209487e+03	1.250240e+04	0.000000e+00	26.200	93.850	3.528250e+02	6.366046e+05
Investments	2541.0	7.218659e+02	6.793860e+03	0.000000e+00	1.000	8.200	6.380000e+01	1.999786e+05
Current_assets	4176.0	1.350360e+03	1.015557e+04	1.000000e-01	36.600	148.350	5.150000e+02	3.548152e+05
et_working_capital	4219.0	1.628742e+02	3.182030e+03	-6.383900e+04	-1.100	16.700	8.650000e+01	8.578280e+04
Quick_ratio_times	4151.0	1.497355e+00	9.327519e+00	0.000000e+00	0.410	0.670	1.030000e+00	3.410000e+02
urrent_ratio_times	4151.0	2.257398e+00	1.247829e+01	0.000000e+00	0.930	1.230	1.720000e+00	5.050000e+02
equity_ratio_times	4256.0	2.871563e+00	1.559997e+01	0.000000e+00	0.220	0.790	1.750000e+00	4.560000e+02
nt_liabilities_times	4151.0	5.284197e-01	4.796342e+00	0.000000e+00	0.020	0.070	1.900000e-01	1.650000e+02
_of_sales_per_day	4156.0	1.451579e+02	2.521992e+03	0.000000e+00	2.880	8.040	2.197000e+01	1.280408e+05
Creditors_turnover	3865.0	1.681226e+01	7.567492e+01	0.000000e+00	3.720	6.170	1.169000e+01	2.401000e+03
Debtors_turnover	3871.0	1.792903e+01	9.016443e+01	0.000000e+00	3.810	6.470	1.185000e+01	3.135200e+03
d_goods_turnover	3382.0	8.436999e+01	5.626374e+02	-9.000000e-02	8.190	17.320	4.001250e+01	1.794760e+04
WIP_turnover	3492.0	2.868451e+01	1.696509e+02	-1.800000e-01	5.100	9.860	2.024000e+01	5.651400e+03
_material_turnover	3828.0	1.773393e+01	3.431259e+02	-2.000000e+00	3.020	6.410	1.182250e+01	2.109200e+04
hares_outstanding	3446.0	2.376491e+07	1.709790e+08	-2.147484e+09	1308382.500	4750000.000	1.090602e+07	4.130401e+09
Equity_face_value	3446.0	-1.094829e+03	3.410136e+04	-9.999989e+05	10.000	10.000	1.000000e+01	1.000000e+05
EPS	4256.0	-1.962175e+02	1.306195e+04	-8.431818e+05	0.000	1.490	1.000000e+01	3.452253e+04
Adjusted_EPS	4256.0	-1.975276e+02	1.306193e+04	-8.431818e+05	0.000	1.240	7.615000e+00	3.452253e+04
Total_liabilities	4256.0	3.573617e+03	3.007444e+04	1.000000e-01	91.300	315.500	1.120800e+03	1.176509e+06
PE_on_BSE	1629.0	5.546229e+01	1.304445e+03	-1.116640e+03	2.970	8.690	1.700000e+01	5.100274e+04
Default	4256.0	5.498120e-02	2.279704e-01	0.000000e+00	0.000	0.000	0.000000e+00	1.000000e+00

Tab: 4 Summary of the Dataset

Insights-

- From the above table we can infer the count,mean, std , 25% , 50% ,75% and min & max values of the all variables present in the dataset.
- There is no bad value present in the dataset.

Checking the Shape of the Dataframe :

Shape attribute tells us number of observations and variables we have in the data set. It is used to check the dimension of data. The Company(FRA) .csv data set has 4256 observations (rows) and 50 variables (columns) in the dataset.

No. of Rows	No. of Columns
4256	50

Tab: 5 Shape of the Dataset

Checking the Appropriateness of Datatypes & Information of the Dataframe :

The info() function is used to print a concise summary of a DataFrame. This method prints information about a DataFrame including the index d-type and column d-types, non-null values and memory usage.

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 4256 entries, 0 to 4255
Data columns (total 50 columns):
 #   Column                                Non-Null Count  Dtype
---  ------                                --------------  -----
 0   Total_assets                          4256 non-null   float64
 1   Net_worth                             4256 non-null   float64
 2   Total_income                          4025 non-null   float64
 3   Change_in_stock                       3706 non-null   float64
 4   Total_expenses                        4091 non-null   float64
 5   Profit_after_tax                      4102 non-null   float64
 6   PBDITA                                4102 non-null   float64
 7   PBT                                   4102 non-null   float64
 8   Cash_profit                           4102 non-null   float64
 9   PBDITA_as_perc_of_total_income        4177 non-null   float64
 10  PBT_as_perc_of_total_income           4177 non-null   float64
 11  PAT_as_perc_of_total_income           4177 non-null   float64
 12  Cash_profit_as_perc_of_total_income   4177 non-null   float64
 13  PAT_as_perc_of_net_worth              4256 non-null   float64
 14  Sales                                 3951 non-null   float64
 15  Income_from_fincial_services         3145 non-null   float64
 16  Other_income                          2700 non-null   float64
 17  Total_capital                         4251 non-null   float64
 18  Reserves_and_funds                    4158 non-null   float64
 19  Borrowings                            3825 non-null   float64
 20  Current_liabilities_and_provisions    4146 non-null   float64
 21  Deferred_tax_liability                2887 non-null   float64
 22  Shareholders_funds                    4256 non-null   float64
 23  Cumulative_retained_profits           4211 non-null   float64
 24  Capital_employed                      4256 non-null   float64
 25  TOL_to_TNW                            4256 non-null   float64
```

```
RangeIndex: 4256 entries, 0 to 4255
Data columns (total 50 columns):
 #   Column                                    Non-Null Count  Dtype
26  Total_term_liabilities__to__tangible_net_worth  4256 non-null   float64
27  Contingent_liabilities__to__Net_worth_perc      4256 non-null   float64
28  Contingent_liabilities                          2854 non-null   float64
29  Net_fixed_assets                                4124 non-null   float64
30  Investments                                     2541 non-null   float64
31  Current_assets                                  4176 non-null   float64
32  Net_working_capital                             4219 non-null   float64
33  Quick_ratio_times                               4151 non-null   float64
34  Current_ratio_times                             4151 non-null   float64
35  Debt_to_equity_ratio_times                      4256 non-null   float64
36  Cash_to_current_liabilities_times               4151 non-null   float64
37  Cash_to_average_cost_of_sales_per_day           4156 non-null   float64
38  Creditors_turnover                              3865 non-null   float64
39  Debtors_turnover                                3871 non-null   float64
40  Finished_goods_turnover                         3382 non-null   float64
41  WIP_turnover                                    3492 non-null   float64
42  Raw_material_turnover                           3828 non-null   float64
43  Shares_outstanding                              3446 non-null   float64
44  Equity_face_value                               3446 non-null   float64
45  EPS                                             4256 non-null   float64
46  Adjusted_EPS                                    4256 non-null   float64
47  Total_liabilities                               4256 non-null   float64
48  PE_on_BSE                                       1629 non-null   float64
49  Default                                         4256 non-null   int64
dtypes: float64(49), int64(1)
memory usage: 1.6 MB
```

Tab: 6 Appropriateness of Datatypes & Information of the Dataframe

Insights :

- From the above results we can see that there are null values present in most of the columns.
- There are total 4256 rows & 50 columns given in this dataset,indexed from 0 to 4255.
- Out of 50 variables 49 are float64 and 1 variable is of int64 d-type. Memory used by the dataset: 1.6 MB.

Checking for Null Values and % of Null Values Present in Each Variable.

	Null	Null %
PE_on_BSE	2627	61.72
Investments	1715	40.30
Other_income	1556	36.56
Contingent_liabilities	1402	32.94
Deferred_tax_liability	1369	32.17
Income_from_fincial_services	1111	26.10
Finished_goods_turnover	874	20.54
Equity_face_value	810	19.03
Shares_outstanding	810	19.03
WIP_turnover	764	17.95
Change_in_stock	550	12.92
Borrowings	431	10.13
Raw_material_turnover	428	10.06
Creditors_turnover	391	9.19
Debtors_turnover	385	9.05
Sales	305	7.17
Total_income	231	5.43
Total_expenses	165	3.88
Profit_after_tax	154	3.62

	Null	Null %
Cash_profit	154	3.62
PBT	154	3.62
PBDITA	154	3.62
Net_fixed_assets	132	3.10
Current_liabilities_and_provisions	110	2.58
Quick_ratio_times	105	2.47
Current_ratio_times	105	2.47
Cash_to_current_liabilities_times	105	2.47
Cash_to_average_cost_of_sales_per_day	100	2.35
Reserves_and_funds	98	2.30
Current_assets	80	1.88
Cash_profit_as_perc_of_total_income	79	1.86
PAT_as_perc_of_total_income	79	1.86
PBT_as_perc_of_total_income	79	1.86
PBDITA_as_perc_of_total_income	79	1.86
Cumulative_retained_profits	45	1.06
Net_working_capital	37	0.87
Total_capital	5	0.12

Tab: 7 Checking for Null Values and % of Null Values Present in Each Variable.

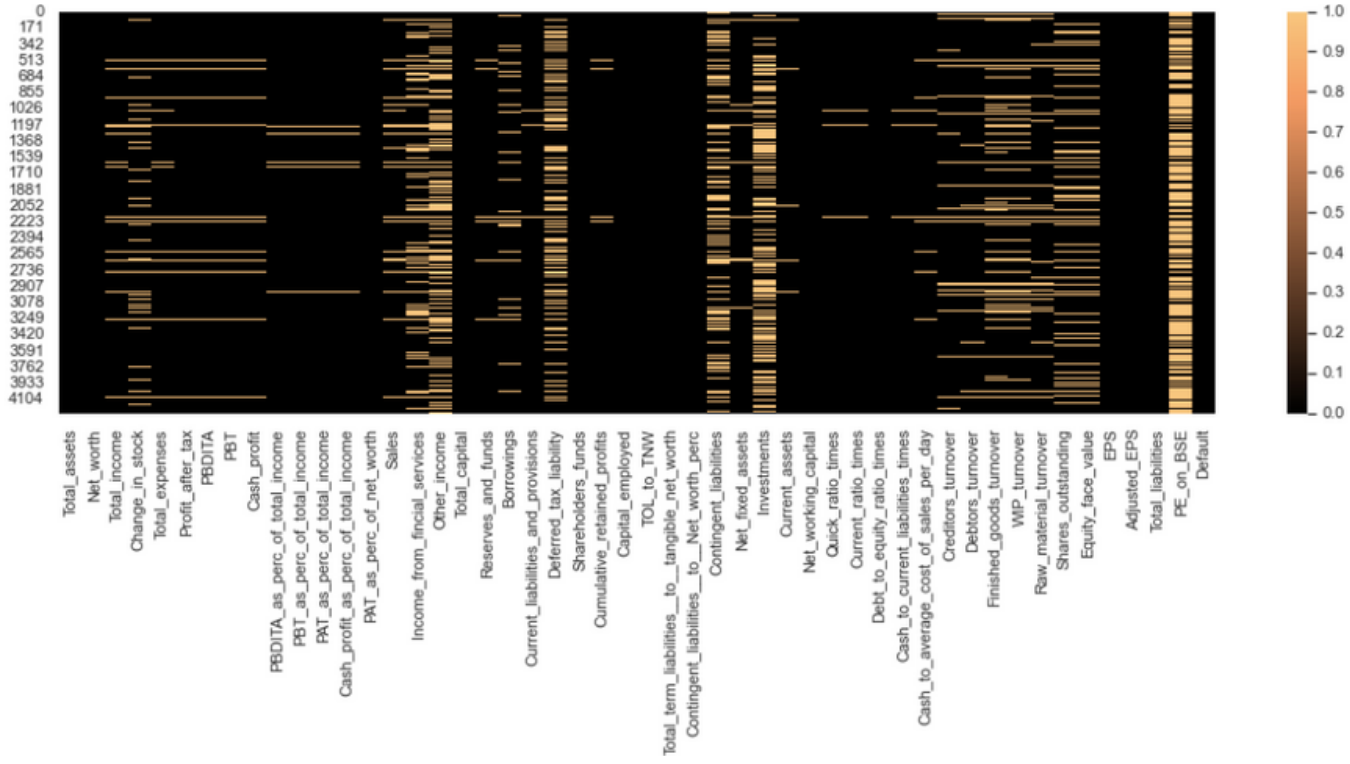


Fig: 1 Null Values Present in Each Variable

Note -

We will decide to drop the columns which have null % more than 30 % , because as these columns have too much null values and these can't help us in our analysis.

Checking for Null Values and % of Null Values Present in Each Variable After Dropping Variables which have Null % more than 30 %.

	Null	Null %
Income_from_fincial_services	1111	26.10
Finished_goods_turnover	874	20.54
Equity_face_value	810	19.03
Shares_outstanding	810	19.03
WIP_turnover	764	17.95
Change_in_stock	550	12.92
Borrowings	431	10.13
Raw_material_turnover	428	10.06
Creditors_turnover	391	9.19
Debtors_turnover	385	9.05
Sales	305	7.17
Total_income	231	5.43
Total_expenses	165	3.88
Profit_after_tax	154	3.62
PBDITA	154	3.62
PBT	154	3.62
Cash_profit	154	3.62
Net_fixed_assets	132	3.10
Current_liabilities_and_provisions	110	2.58

	Null	Null %
Quick_ratio_times	105	2.47
Cash_to_current_liabilities_times	105	2.47
Current_ratio_times	105	2.47
Cash_to_average_cost_of_sales_per_day	100	2.35
Reserves_and_funds	98	2.30
Current_assets	80	1.88
PBDITA_as_perc_of_total_income	79	1.86
PBT_as_perc_of_total_income	79	1.86
PAT_as_perc_of_total_income	79	1.86
Cash_profit_as_perc_of_total_income	79	1.86
Cumulative_retained_profits	45	1.06
Net_working_capital	37	0.87
Total_capital	5	0.12

Tab: 8 Checking for Null Values and % of Null Values Present in Each Variable After Dropping Variables which have Null % more than 30 %.

Insights-

Now we have columns which have null % less than 30 %. We will impute these null values by using median i.e. as outliers is present in the data so median will be effective choice for null value imputation.

Now , we have 4256 rows and 45 columns.

Checking for Anomalies in the Dataset.

```
TOTAL_ASSETS :  2961
[ 827.6   67.7  238.4 ... 1591.9  159.6  833.8]

**************************************************

NET_WORTH :  2376
[336.5  24.3  78.9 ... 123.8  35.3 664.6]

**************************************************

TOTAL_INCOME :  2870
[ 534.1  137.9  331.2 ... 3717.2  172.9 2314.7]

**************************************************

CHANGE_IN_STOCK :  1164
[ 13.5  -3.7 -18.1 ...  41.4 306.5 321.6]

**************************************************

TOTAL_EXPENSES :  2898
[ 508.7  131.   309.2 ... 3681.5  169.7 2151.6]

**************************************************

PROFIT_AFTER_TAX :  1467
[ 38.9   3.2   3.9 ... 458.6  27.8 195.2]

**************************************************

PBDITA :  1826
[124.4   5.5  25.8 ...  81.1  18.4 348.4]

**************************************************

PBT :  1568
[ 64.6   1.   10.5 ... 467.9  75.4 303. ]

**************************************************

CASH_PROFIT :  1655
[ 95.2   3.8   9.4 ... 718.4 820.3 219.5]

**************************************************

PBDITA_AS_PERC_OF_TOTAL_INCOME :  2032
[23.29  3.99  7.79 ...  3.43  6.55 32.  ]

**************************************************

PBT_AS_PERC_OF_TOTAL_INCOME :  1878
[12.1   0.73  3.17 ...  9.3   2.73 13.09]

**************************************************

PAT_AS_PERC_OF_TOTAL_INCOME :  1692
[ 7.28  2.32  1.18 ... -6.17  4.47 13.67]

**************************************************

CASH_PROFIT_AS_PERC_OF_TOTAL_INCOME :  1867
[17.82  2.76  2.84 ... 32.    0.15  9.48]

**************************************************

PAT_AS_PERC_OF_NET_WORTH :  2385
[12.27  0.    5.07 ... 32.42  8.65 33.55]

**************************************************

SALES :  2847
[ 533.5  135.5  330.6 ... 3669.8  172.1 2309.4]

**************************************************

INCOME_FROM_FINCIAL_SERVICES :  561
[6.00000e-01         nan 2.00000e+00 2.00000e-01 7.30000e+00 5.71000e+01
 1.40000e+00 1.00000e-01 2.28400e+02 6.20000e+01 5.00000e-01 1.60000e+00

**************************************************

TOTAL_CAPITAL :  1525
[  87.6   11.9   25.  ... 2906.1  165.    74.9]

**************************************************

RESERVES_AND_FUNDS :  2361
[249.    4.3  56.7 ...  86.1 333.1 589.7]

**************************************************

BORROWINGS :  2135
[390.7  16.6  44.7 ... 160.2  58.6 512.5]

**************************************************

CURRENT_LIABILITIES_AND_PROVISIONS :  2095
[ 43.9  23.7 102.2 ... 278.3 262.3 675.9]

**************************************************

SHAREHOLDERS_FUNDS :  2413
[336.5  24.3  78.9 ... 123.8  35.3 664.6]

**************************************************

CUMULATIVE_RETAINED_PROFITS :  2265
[248.9  -8.2  53.1 ...  44.4 373.6 477.2]

**************************************************

CAPITAL_EMPLOYED :  2783
[727.2  40.9 123.6 ...  60.9 888.1 704. ]

**************************************************

TOL_TO_TNW :  841
[ 1.2800e+00  1.5300e+00  1.7000e+00  3.6900e+00  8.1000e-01  1.0500e+00
  6.6000e-01  2.9800e+00  1.8900e+00  7.2700e+00  1.2000e+00  3.3700e+00

**************************************************

TOTAL_TERM_LIABILITIES__TO__TANGIBLE_NET_WORTH :  508
[ 9.9000e-01  2.1000e-01  3.3000e-01  2.2000e-01  4.4000e-01  3.0000e-01
  2.0000e-02  1.3900e+00  4.1000e-01  5.6000e-01  4.0500e+00  2.7000e-01

**************************************************

CONTINGENT_LIABILITIES__TO__NET_WORTH_PERC :  1926
[186.21  47.74  30.42 ...  52.62   8.42  15.72]

**************************************************

NET_FIXED_ASSETS :  2234
[461.1  18.5  56.8 ... 402.5 307.5 308.5]

**************************************************

QUICK_RATIO_TIMES :  409
[9.900e-01 6.700e-01 1.110e+00 3.500e-01 5.900e-01 1.200e-01 8.000e-01
 7.100e-01 7.500e-01 3.200e-01 3.000e-01 7.000e-01 5.200e-01 5.500e-01

**************************************************

CURRENT_RATIO_TIMES :  517
[2.520e+00 1.110e+00 1.310e+00 1.280e+00 2.090e+00 1.580e+00 1.200e-01
 6.800e-01 1.120e+00 1.360e+00 8.800e-01 7.000e-01 1.170e+00 8.600e-01

**************************************************

DEBT_TO_EQUITY_RATIO_TIMES :  642
[1.1600e+00 6.8000e-01 5.7000e-01 1.9300e+00 5.4000e-01 9.4000e-01
 4.8000e-01 3.4000e+00 1.1000e+00 1.1100e+00 6.3100e+00 6.3000e-01
```

```
**********************************************

CASH_TO_CURRENT_LIABILITIES_TIMES :  249
[6.000e-02 2.000e-02 1.900e-01 7.000e-02 5.000e-02 9.000e-02 4.000e-02
 1.100e-01 3.000e-02 1.300e-01 8.000e-02 1.000e-02 4.800e+00 1.339e+01

**********************************************

CASH_TO_AVERAGE_COST_OF_SALES_PER_DAY :  2051
[  5.41   1.62  26.42 ...  18.86  25.78 342.65]

**********************************************

CREDITORS_TURNOVER :  1608
[11.6    nan  2.24 ... 24.49  4.3  14.86]

**********************************************

DEBTORS_TURNOVER :  1640
[ 5.65   nan  2.51 ...  6.65 18.57 13.42]

**********************************************

FINISHED_GOODS_TURNOVER :  2201
[  3.99    nan  17.67 ... 162.7   11.    59.28]

**********************************************

WIP_TURNOVER :  1941
[ 3.37   nan  8.76 ... 11.9  31.27  8.28]

**********************************************

RAW_MATERIAL_TURNOVER :  1601
[14.87   nan  8.35 ... 40.85  1.38 13.76]

**********************************************

SHARES_OUTSTANDING :  2370
[ 8760056.       nan 10000000. ...  8508479.  8162700.  7479762.]

**********************************************

EQUITY_FACE_VALUE :  18
[ 1.000000e+01           nan  1.000000e+02  5.000000e+00  8.000000e+00
  4.000000e+00  1.000000e+00  2.000000e+00  1.000000e+03  2.500000e+01

**********************************************

EPS :  1815
[ 4.44  0.   17.6  ...  7.03 -3.14 26.58]

**********************************************

ADJUSTED_EPS :  1730
[ 4.44  0.   17.6  ...  7.03 -3.14 26.58]

**********************************************

TOTAL_LIABILITIES :  2961
[ 827.6   67.7  238.4 ... 1591.9  159.6  833.8]

**********************************************

DEFAULT :  2
[0 1]
```

Tab: 9 Checking for Anomalies for variables in the Dataset

Observation-

There is no Anomalies present in the dataset , but have nan values in most of columns.

Checking Duplicate Values

Here we found the number of duplicated rows in data set i.e. 665, as we know that duplicated rows are not useful we decided drop them by using .drop( ) function.

After using the .drop function ( ) , we drop all the duplicate rows of the data ,check the shape of the data once again.

Number of Rows : 3591

Number of Columns : 45

Observation :

Number of duplicate rows = 0

Checking the Class Proportion of Dependent Variable ('Default').

Proportion of 1s and 0s	Ratio
0	93.5
1	6.5

Tab: 10 Checking the Class Proportion of Dependent Variable ('Default')

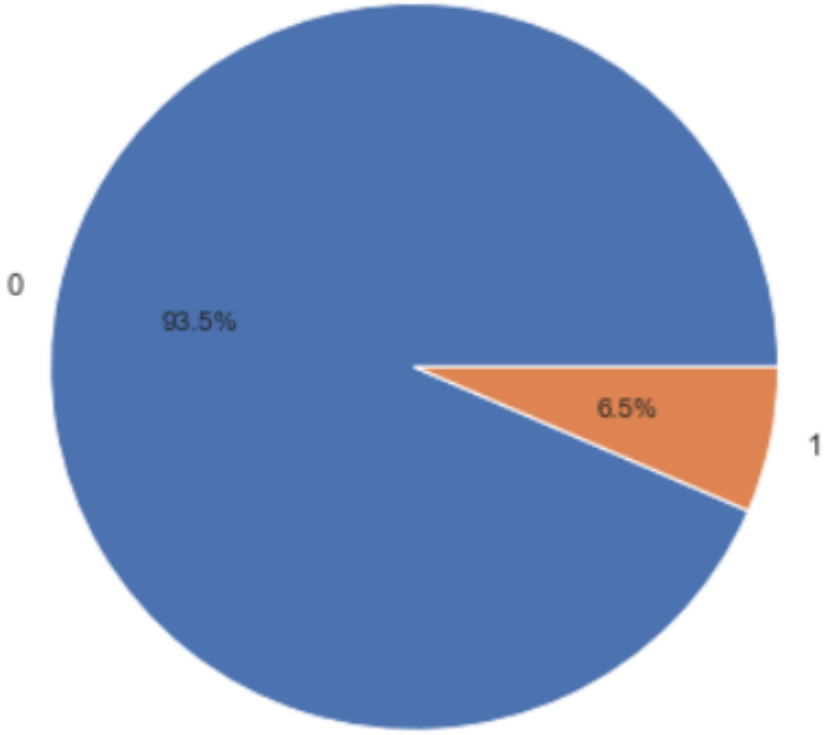


Fig: 2 Pie-Plot of Class Proportion of Target Column

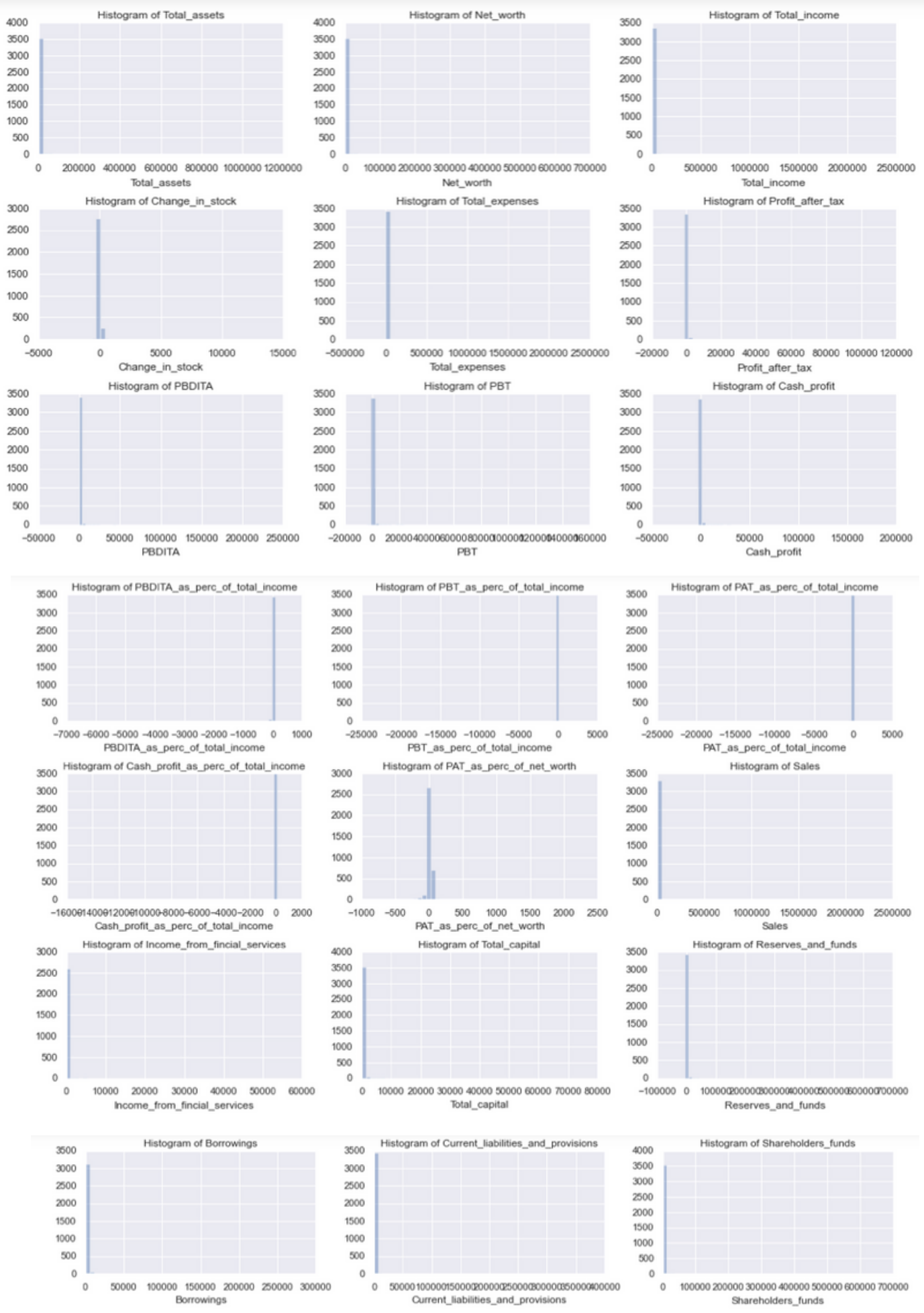
Insights -

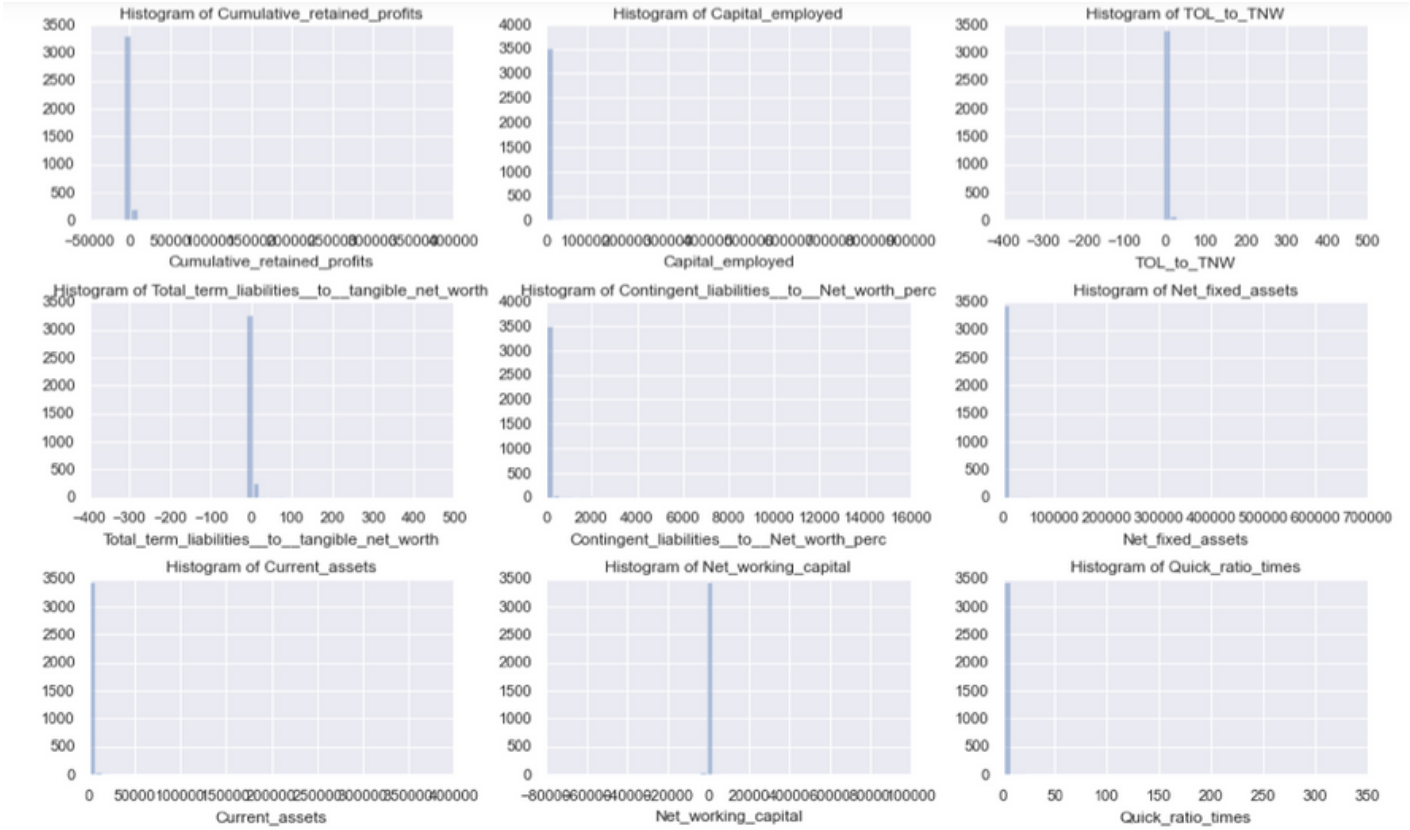
- 93.5% of the data-points belongs to class 0.
- Only 6.5% of the data-points belongs to class 1.
- As we saw there might be class imbalance problem , we will solve this issue at the time of model bulidng by using SMOTE.

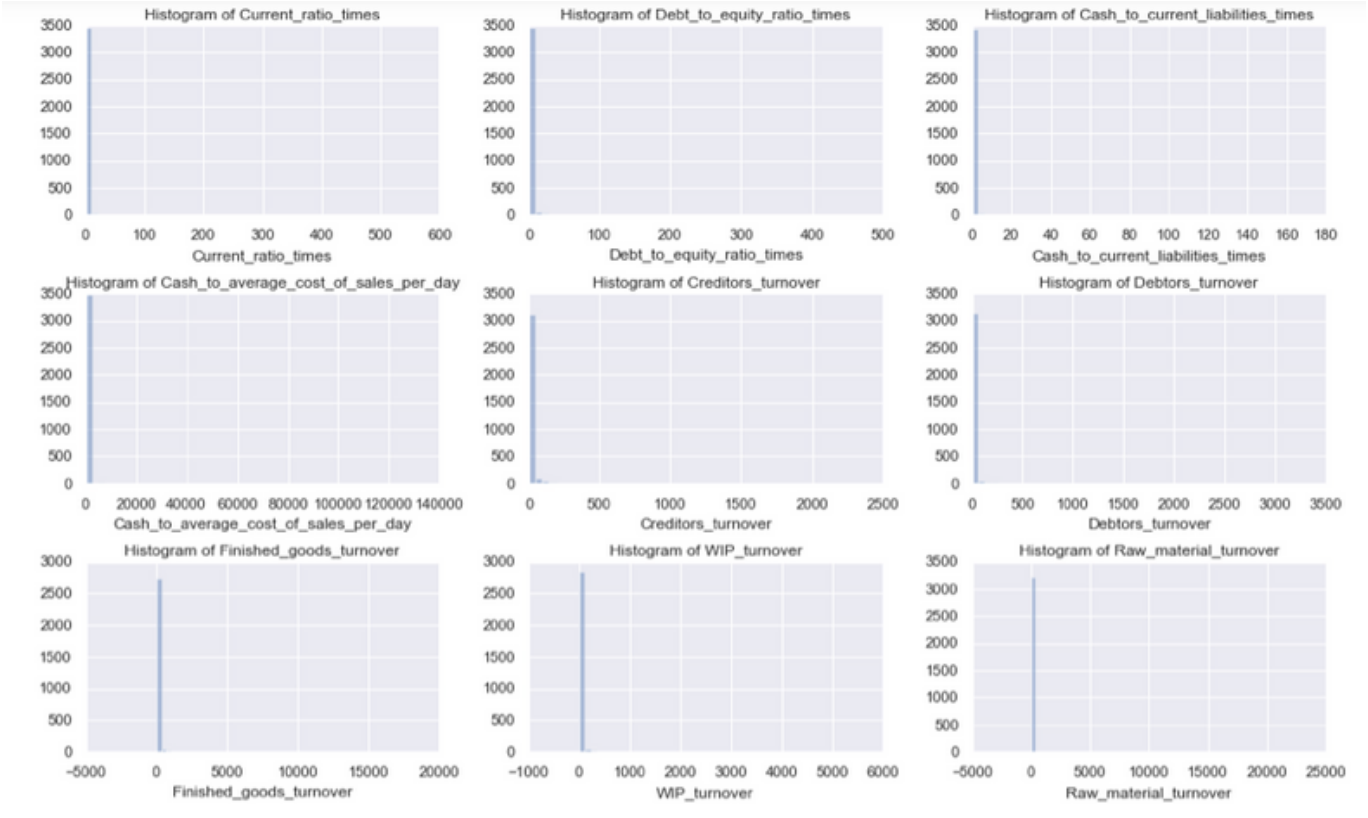
Univariate Analysis of continuous Numerical Variables

A **histogram** takes as input a numeric variable only. The variable is cut into several bins, and the number of observation per bin is represented by the height of the bar. It is possible to represent the distribution of several variable on the same axis using this technique.

A **box-plot** gives a nice summary of one or several numeric variables. The line that divides the box into 2 parts represents the median of the data. The end of the box shows the upper and lower quartiles. The extreme lines show the highest and lowest value excluding outliers.


Histogram of Total_assets
Total_assets
Histogram of Net_worth
Net_worth
Histogram of Total_income
Total_income
Histogram of Change_in_stock
Change_in_stock
Histogram of Total_expenses
Total_expenses
Histogram of Profit_after_tax
Profit_after_tax
Histogram of PBDITA
PBDITA
Histogram of PBT
PBT
Histogram of Cash_profit
Cash_profit
Histogram of PBDITA_as_perc_of_total_income
PBDITA_as_perc_of_total_income
Histogram of PBT_as_perc_of_total_income
PBT_as_perc_of_total_income
Histogram of PAT_as_perc_of_total_income
PAT_as_perc_of_total_income
Histogram of Cash_profit_as_perc_of_total_income
Cash_profit_as_perc_of_total_income
Histogram of PAT_as_perc_of_net_worth
PAT_as_perc_of_net_worth
Histogram of Sales
Sales
Histogram of Income_from_fincial_services
Income_from_fincial_services
Histogram of Total_capital
Total_capital
Histogram of Reserves_and_funds
Reserves_and_funds
Histogram of Borrowings
Borrowings
Histogram of Current_liabilities_and_provisions
Current_liabilities_and_provisions
Histogram of Shareholders_funds
Shareholders_funds


Histogram of Cumulative_retained_profits
Cumulative_retained_profits
Histogram of Capital_employed
Capital_employed
Histogram of TOL_to_TNW
TOL_to_TNW
Histogram of Total_term_liabilities__to__tangible_net_worth
Total_term_liabilities__to__tangible_net_worth
Histogram of Contingent_liabilities__to__Net_worth_perc
Contingent_liabilities__to__Net_worth_perc
Histogram of Net_fixed_assets
Net_fixed_assets
Histogram of Current_assets
Current_assets
Histogram of Net_working_capital
Net_working_capital
Histogram of Quick_ratio_times
Quick_ratio_times


Histogram of Current_ratio_times
Current_ratio_times
Histogram of Debt_to_equity_ratio_times
Debt_to_equity_ratio_times
Histogram of Cash_to_current_liabilities_times
Cash_to_current_liabilities_times
Histogram of Cash_to_average_cost_of_sales_per_day
Cash_to_average_cost_of_sales_per_day
Histogram of Creditors_turnover
Creditors_turnover
Histogram of Debtors_turnover
Debtors_turnover
Histogram of Finished_goods_turnover
Finished_goods_turnover
Histogram of WIP_turnover
WIP_turnover
Histogram of Raw_material_turnover
Raw_material_turnover


Histogram of Shares_outstanding
Shares_outstanding
1e9
Histogram of Equity_face_value
Equity_face_value
Histogram of EPS
EPS

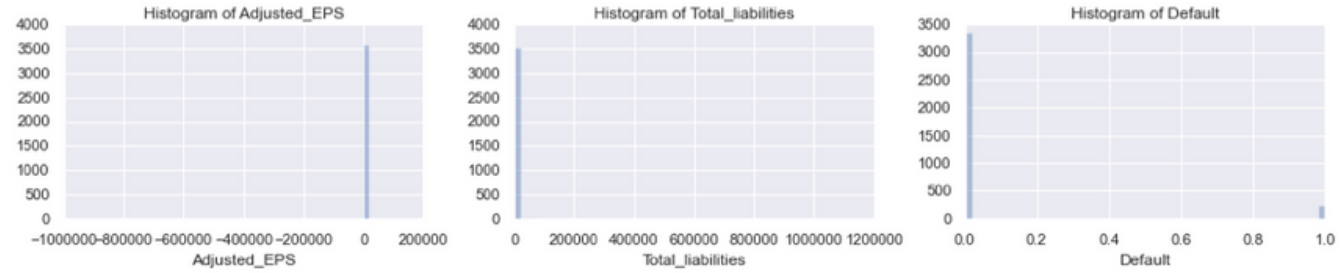
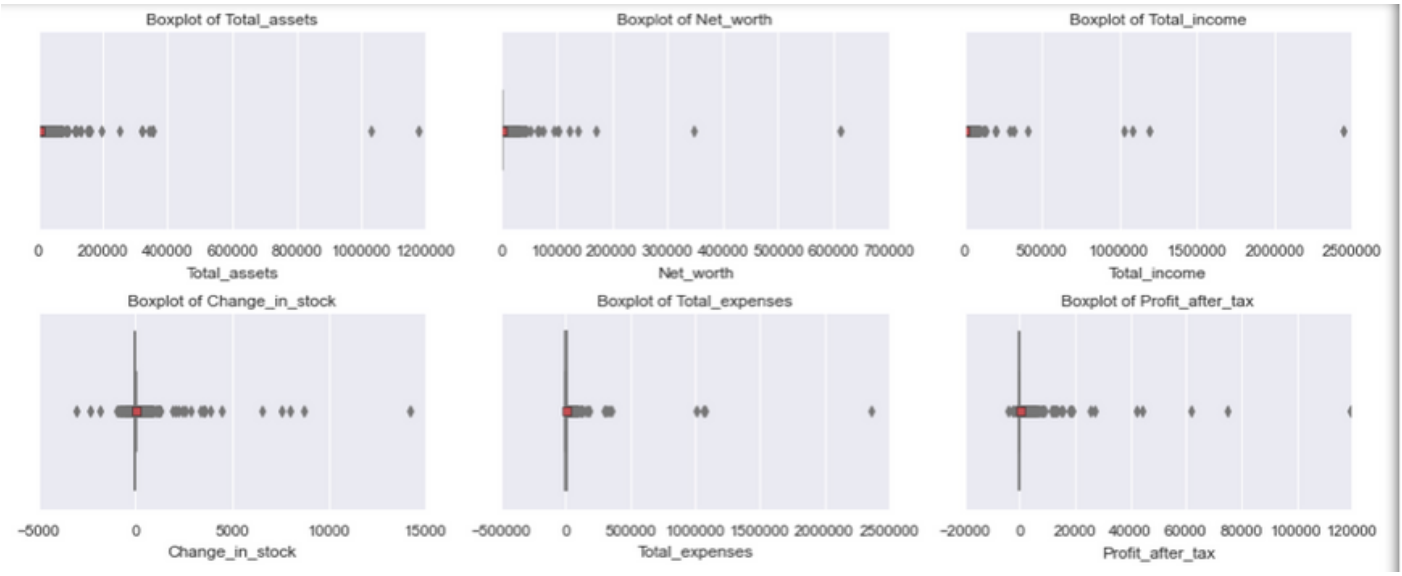
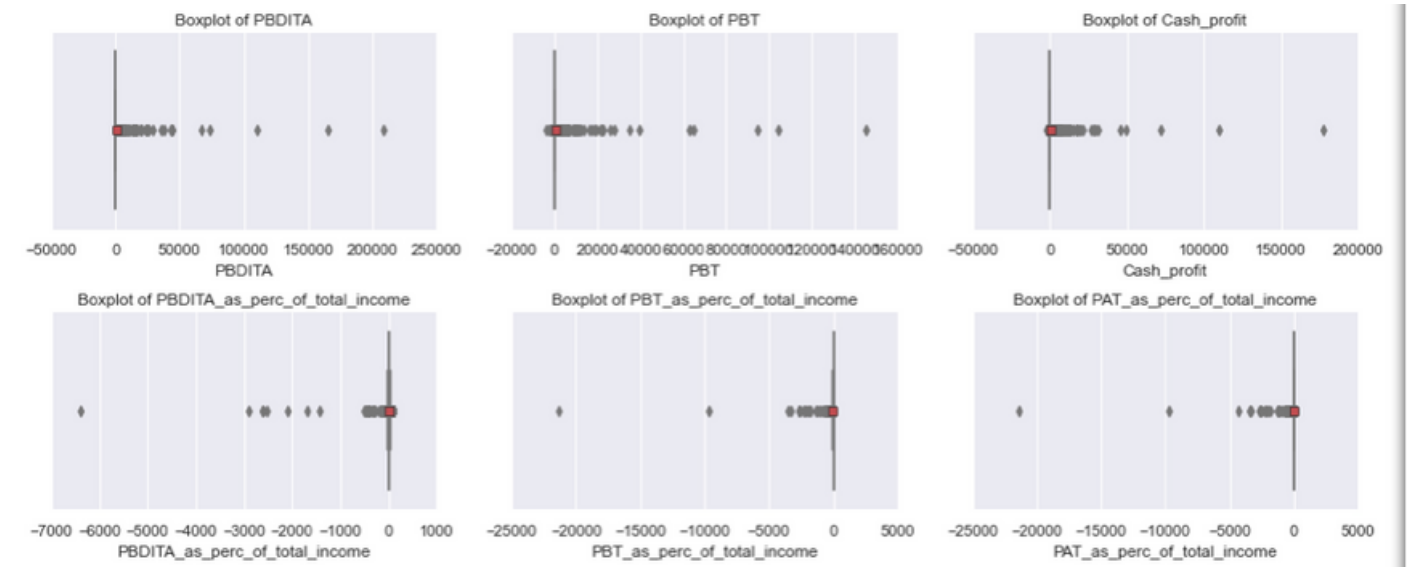
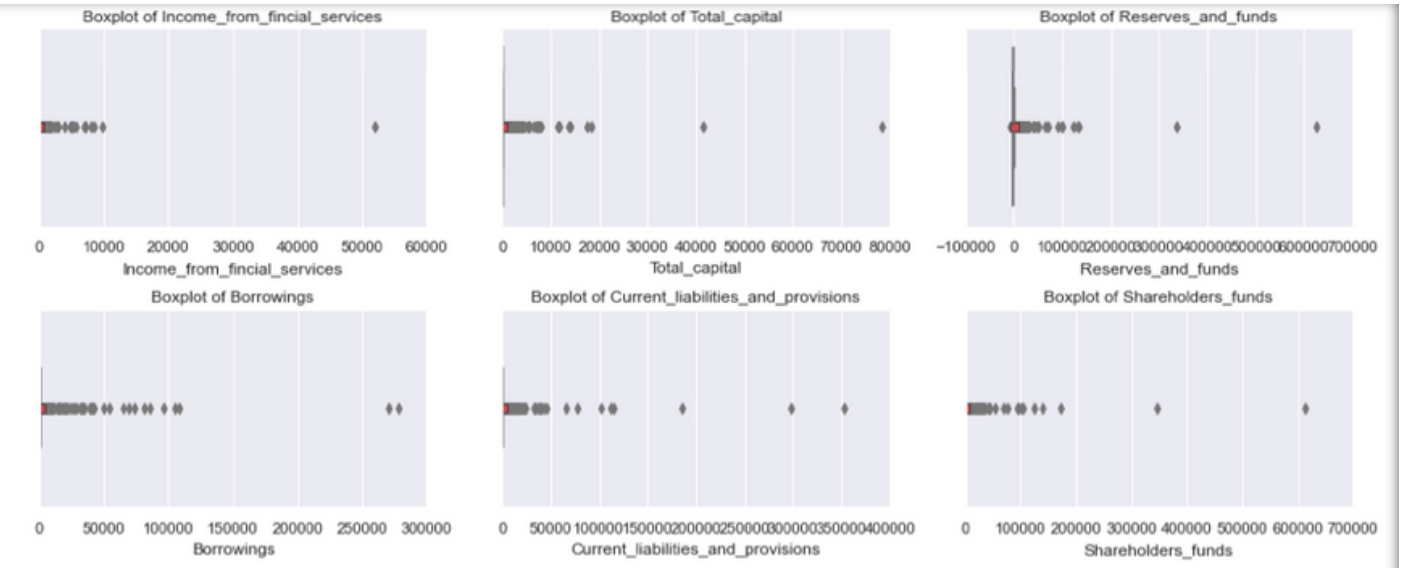
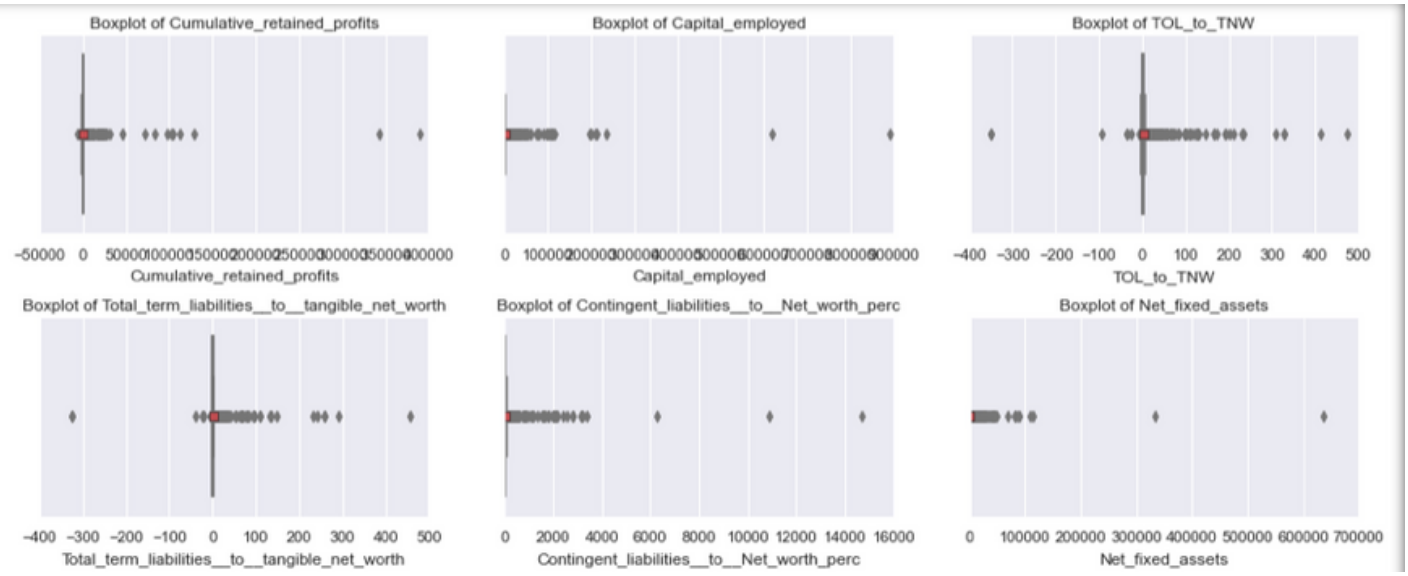


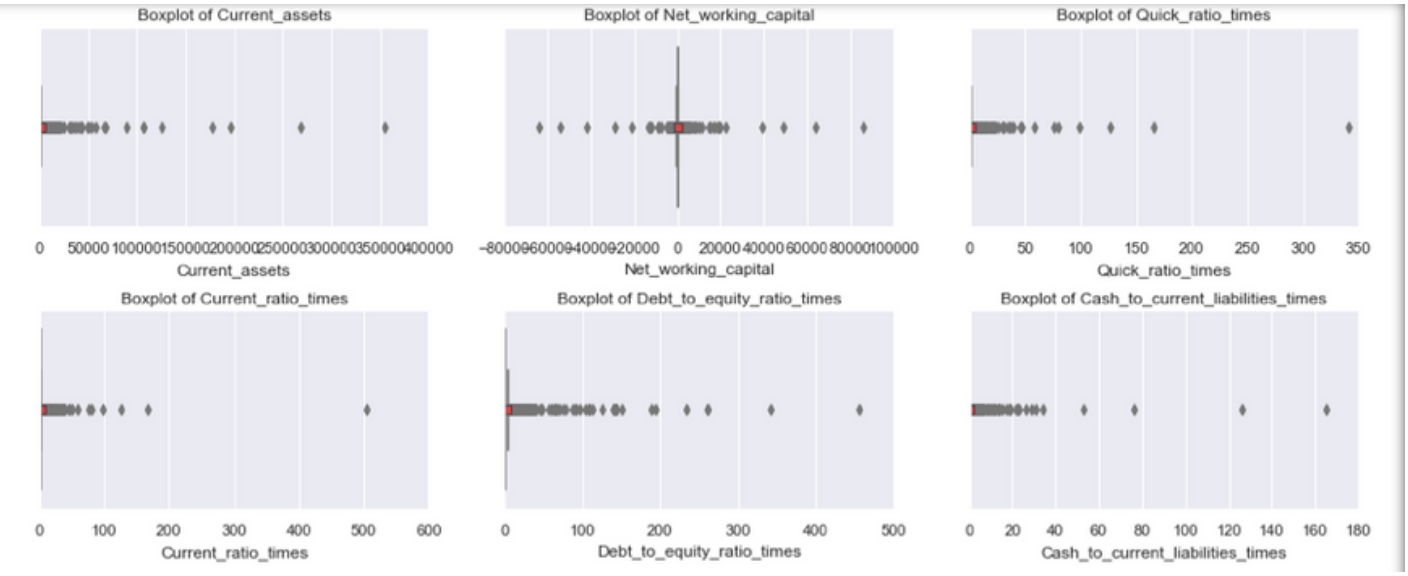
Fig: 3 Histogram of all the Variables






Boxplot of Income_from_fincial_services
Income_from_fincial_services
Boxplot of Total_capital
Total_capital
Boxplot of Reserves_and_funds
Reserves_and_funds
Boxplot of Borrowings
Borrowings
Boxplot of Current_liabilities_and_provisions
Current_liabilities_and_provisions
Boxplot of Shareholders_funds
Shareholders_funds


Boxplot of Cumulative_retained_profits
Cumulative_retained_profits
Boxplot of Capital_employed
Capital_employed
Boxplot of TOL_to_TNW
TOL_to_TNW
Boxplot of Total_term_liabilities__to__tangible_net_worth
Total_term_liabilities__to__tangible_net_worth
Boxplot of Contingent_liabilities__to__Net_worth_perc
Contingent_liabilities__to__Net_worth_perc
Boxplot of Net_fixed_assets
Net_fixed_assets


Boxplot of Current_assets
Current_assets
Boxplot of Net_working_capital
Net_working_capital
Boxplot of Quick_ratio_times
Quick_ratio_times
Boxplot of Current_ratio_times
Current_ratio_times
Boxplot of Debt_to_equity_ratio_times
Debt_to_equity_ratio_times
Boxplot of Cash_to_current_liabilities_times
Cash_to_current_liabilities_times

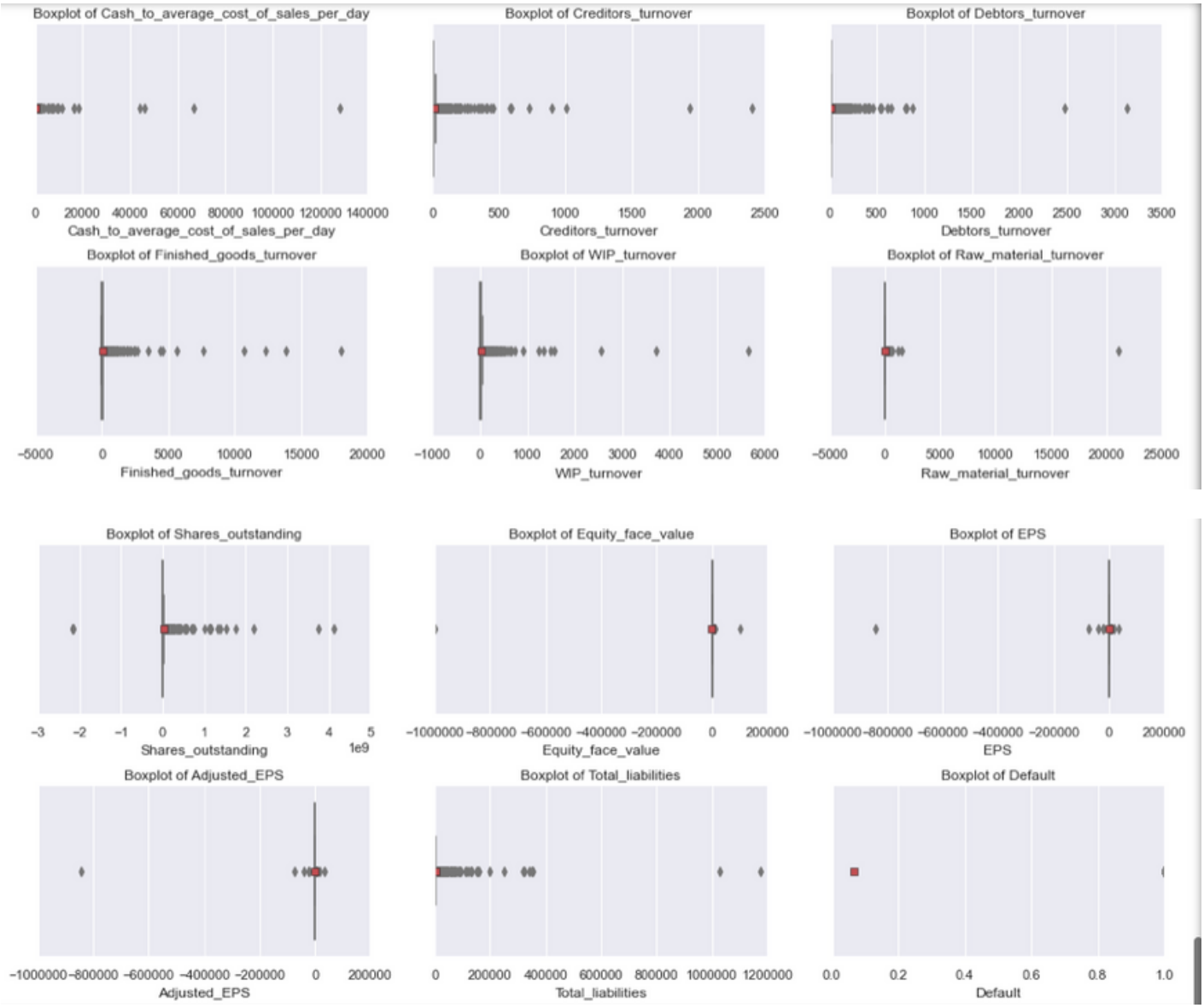


Fig: 4 Box-Plot of all the Variables

Insights of Univariate Analysis

- Total_assets - Total assets of customer ranges from a minimum of 0.100 to maximum of 1176509.
- Average Total assets of customer is around 3575.297.
- Total_assets has outliers.
- Net_worth - Net worth of the customer of present year ranges from a minimum of 0.00 to maximum of 613151.60.
- Average Net worth of the customer of present year is around 1352.58.
- Net_worth has outliers.
- Total_income - Total income of the customer ranges from a minimum of 0.00 to maximum of 2442828.
- Average Total income of the customer is around 4688.190.
- Total_income has outliers.
- Change_in_stock - difference between value of current stock and the value of stock in last trading day ranges from a minimum of -3029.400 to maximum of 14185.500.

- Average difference between value of current stock and the value of stock in last trading day is around 43.702.
- Change_in_stock has outliers.
- Total_expenses - Total expense done by customer ranges from a minimum of -0.100 to maximum of 2366035.
- Average of Total expense done by customer is around 4356.301.
- Total_expenses has outliers.
- Profit_after_tax - Profit after tax deduction ranges from a minimum of -3908.300 to maximum of 119439.100.
- Average Profit after tax deduction is around 295.050.
- Profit_after_tax has outliers.
- PBDITA - Profit before depreciation, income tax and amortization ranges from a minimum of -440.700 to maximum of 208576.500.
- Average PBDITA is around 605.940.
- PBDITA has outliers.
- PBT - Profit before tax deduction ranges from a minimum of -3894.800 to maximum of 145292.600.
- Average Profit before tax deduction is around 410.259.
- PBT has outliers.
- Cash_profit - Total Cash profit ranges from a minimum of -2245.700 to maximum of 176911.800.
- Average of Total Cash profit is around 408.267.
- Cash_profit has outliers.
- PBDITA_as_perc_of_total_income - PBDITA / Total income ranges from a minimum of -6400.00 to maximum of 100.00.
- Average PBDITA_as_perc_of_total_income is around 3.181.
- PBDITA_as_perc_of_total_income has outliers.
- PBT_as_perc_of_total_income - PBT / Total income ranges from a minimum of -21340.00 to maximum of 100.00.
- Average PBT_as_perc_of_total_income is around -18.205.
- PBT_as_perc_of_total_income has outliers.
- PAT_as_perc_of_total_income - PAT / Total income ranges from a minimum of -21340.00 to maximum of 150.00.
- Average PAT_as_perc_of_total_income is around -20.043.
- PAT_as_perc_of_total_income has outliers.

- Cash_profit_as_perc_of_total_income - Cash Profit / Total income ranges from a minimum of -15020.00 to maximum of 100.00.
- Average Cash_profit_as_perc_of_total_income is around -9.025.
- Cash_profit_as_perc_of_total_income has outliers.
- PAT_as_perc_of_net_worth - PAT / Net worth ranges from a minimum of -748.720 to maximum of 2466.670.
- Average PAT_as_perc_of_net_worth is around 10.172.
- PAT_as_perc_of_net_worth has outliers.
- Sales - Sales done by customer ranges from a minimum of 0.100 to maximum of 2384984.
- Average Sales done by customer is around 4645.685.
- Sales has outliers.
- Total_capital - Total capital of the customer ranges from a minimum of 0.100 to maximum of 78273.200.
- Average Total capital of the customer is around 224.663.
- Total_capital has outliers

- Reserves_and_funds - Total reserves and funds of the customer ranges from a minimum of -6525.900 to maximum of 625137.800.
- Average of Total reserves and funds of the customer is around 1210.561.
- Reserves_and_funds has outliers.
- Borrowings - Total amount borrowed by customer ranges from a minimum of 0.100 to maximum of 278257.300.
- Average Total amount borrowed by customer is around 1176.248.
- Borrowings has outliers.

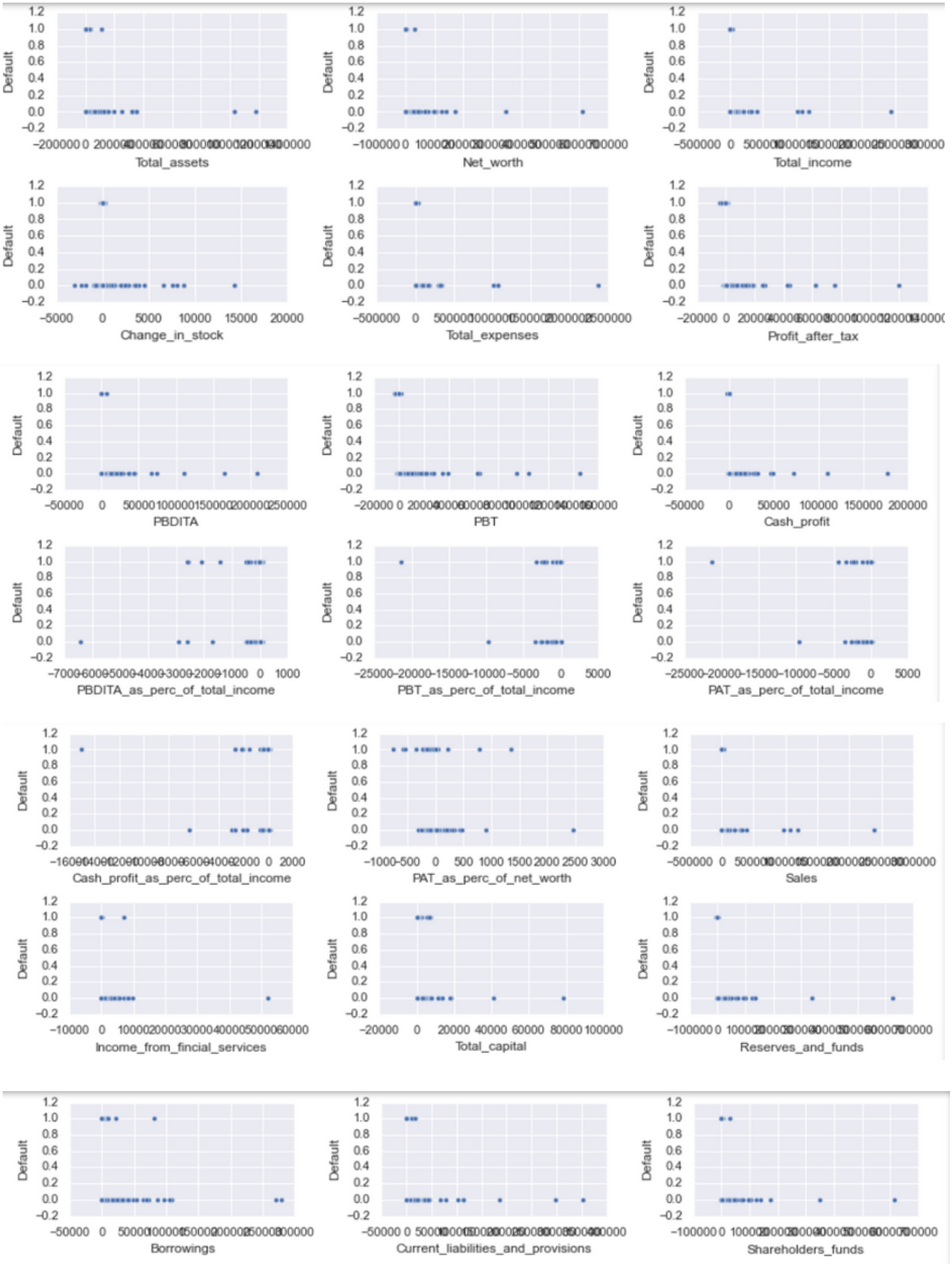
- Current_liabilities_and_provisions - current liabilities of the customer ranges from a minimum of 0.100 to maximum of 352240.300.
- Average current liabilities of the customer is around 960.631.
- Current_liabilities_and_provisions has outliers.
- Shareholders_funds - Amount of equity in a company, which is belong to shareholder ranges from a minimum of 0.00 to maximum of 613151.600.
- Average Amount of equity in a company, which is belong to shareholder is 1377.133.
- Shareholders_funds has outliers.
- Cumulative_retained_profits - Total cumulative profit retained by customer ranges from a minimum of -6534.300 to maximum of 390133.800.
- Average Total cumulative profit retained by customer is around 937.181.
- Cumulative_retained_profits has outliers.
- Capital_employed - Current asset minus current liabilities ranges from a minimum of 0.00 to maximum of 891408.900.
- Average Current asset minus current liabilities is around 2434.761.
- Capital_employed has outliers.
- TOL_to_TNW - Total liabilities of the customer divided by Total net worth ranges from a minimum of -350.480 to maximum of 473.00.
- Average TOL_to_TNW is around 4.027.
- TOL_to_TNW has outliers.
- Total_term_liabilities__to__tangible_net_worth - Short + long term liabilities divided by tangible net worth ranges from a minimum of -325.600 to maximum of 456.00.
- Average Total_term_liabilities__to__tangible_net_worth is around 1.855.
- Total_term_liabilities__to__tangible_net_worth has outliers.
- Contingent_liabilities__to__Net_worth_perc - Contingent liabilities / Net worth ranges from a minimum of 0.00 to maximum of 14704.270.
- Average Contingent_liabilities__to__Net_worth_perc is 55.733.
- Contingent_liabilities__to__Net_worth_perc has outliers.
- Net_fixed_assets - purchase price of all fixed assets ranges from a minimum of 0.00 to maximum of 636604.600.
- Average Net_fixed_assets is around 1209.486.
- Net_fixed_assets has outliers.
- Current_assets - Assets that are expected to be converted to cash within a year ranges from a minimum of 0.100 to maximum of 354815.200.
- Average Current_assets is around 1351.006.
- Current_assets has outliers.
- Net_working_capital - Difference of current liabilities and current assets ranges from a minimum of -63839.00 to maximum of 85782.800.
- Average Net_working_capital is around 162.951.
- Net_working_capital has outliers.
- Quick_ratio_times - Total cash divided by current liabilities ranges from a minimum of 0.00 to maximum of 341.00.
- Average Quick_ratio_times is around 1.497.
- Quick_ratio_times has outliers.
- Current_ratio_times - Current assets divided by current liabilities ranges from a minimum of 0.00 to maximum of 505.00.
- Average Current_ratio_times is around 2.257.
- Current_ratio_times has outliers.
- Debt_to_equity_ratio_times - Total liabilities divided by its shareholder equity ranges from a minimum of 0.00 to maximum of 456.00.
- Average Debt_to_equity_ratio_times is around 2.872.
- Debt_to_equity_ratio_times has outliers.
- Cash_to_current_liabilities_times - Total liquid cash divided by current liabilities ranges from a minimum of 0.00 to maximum of 165.00.
- Average Cash_to_current_liabilities_times is around 0.528.
- Cash_to_current_liabilities_times has outliers.
- Cash_to_average_cost_of_sales_per_day - Total cash divided by average cost of the sales ranges from a minimum of 0.00 to maximum of 128040.760.
- Average Cash_to_average_cost_of_sales_per_day is around 145.157.

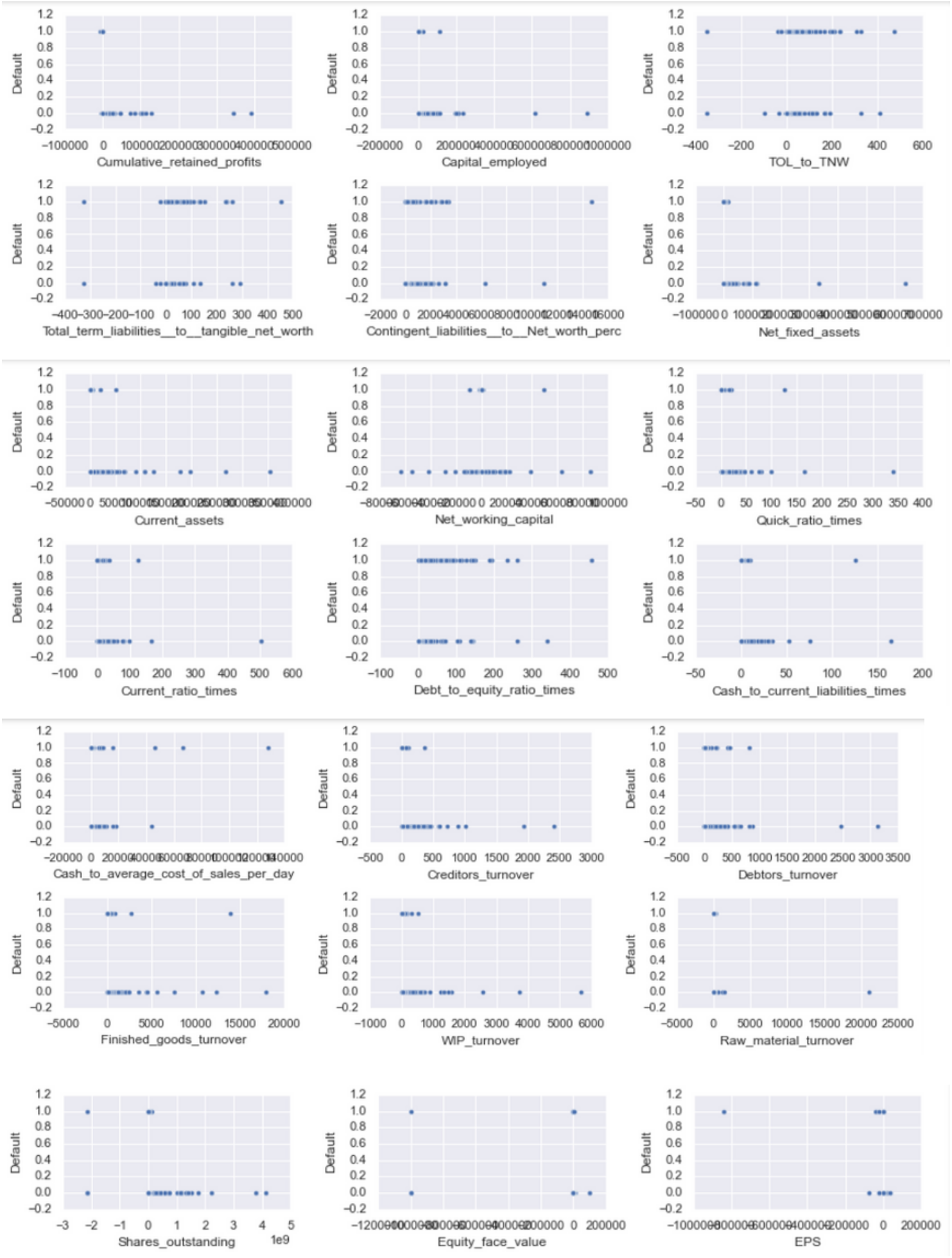
- Cash_to_current_liabilities_times has outliers.
- Cash_to_average_cost_of_sales_per_day - Total cash divided by average cost of the sales ranges from a minimum of 0.00 to maximum of 128040.760.
- Average Cash_to_average_cost_of_sales_per_day is around 145.157.
- Cash_to_average_cost_of_sales_per_day has outliers.
- Creditors_turnover - Net credit purchase divided to average trade creditors ranges from a minimum of 0.00 to maximum of 2401.00.
- Average Creditors_turnover is around 16.812.
- Creditors_turnover has outliers.
- Debtors_turnover - Net credit sales divided by average accounts receivable ranges from a minimum of 0.00 to maximum of 3135.200.
- Average Debtors_turnover is around 17.929.
- Debtors_turnover has outliers.
- Finished_goods_turnover - Annual sales divided by average inventory ranges from a minimum of -0.090 to maximum of 17947.600.
- Average Finished_goods_turnover is around 84.369.
- Finished_goods_turnover has outliers.
- WIP_turnover - The cost of goods sold for a period divided by the average inventory for that period ranges from a minimum of -0.180 to maximum of 5651.400.
- Average WIP_turnover is around 28.684.
- WIP_turnover has outliers.
- Raw_material_turnover - Cost of goods sold is divided by the average inventory for the same period ranges from a minimum of -2.000 to maximum of 21092.000.
- Average Raw_material_turnover is around 17.733.
- Raw_material_turnover has outliers.
- Shares_outstanding - Number of issued shares minus the number of share held in the company ranges from a minimum of -2147484000 to maximum of 4130401000.
- Average Shares_outstanding is around 23764910.
- Shares_outstanding has outliers.
- Equity_face_value - cost of the equity at the time of issuing ranges from a minimum of -999998.900 to maximum of 100000.000.
- Average Equity_face_value is around -1094.828.
- Equity_face_value has outliers.
- EPS - Net income divided by total number of outstanding share ranges from a minimum of -843181.820 to maximum of 34522.530.
- Average EPS is around -196.309.
- EPS has outliers.
- Adjusted_EPS - Adjusted net earning divided by the weighted average number of common share outstanding on a diluted basis during the plan year ranges from a minimum of -843181.820 to maximum of 34522.530.
- Average Adjusted_EPS is around -197.620.
- Adjusted_EPS has outliers.
- Total_liabilities - Sum of all type of liabilities ranges from a minimum of 0.100 to maximum of 1176509.
- Average Total_liabilities is around 3575.297.
- Total_liabilities has outliers.

Bivariate Analysis -

Scatter Plot -

A scatter plot (aka scatter chart, scatter graph) uses dots to represent values for two different numeric variables. The position of each dot on the horizontal and vertical axis indicates values for an individual data point. Scatter plots are used to observe relationships between variables.


Default
Total_assets
Net_worth
Total_income
Change_in_stock
Total_expenses
Profit_after_tax
PBDITA
PBT
Cash_profit
PBDITA_as_perc_of_total_income
PBT_as_perc_of_total_income
PAT_as_perc_of_total_income
Cash_profit_as_perc_of_total_income
PAT_as_perc_of_net_worth
Sales
Income_from_fincial_services
Total_capital
Reserves_and_funds
Borrowings
Current_liabilities_and_provisions
Shareholders_funds


Default
Cumulative_retained_profits
Capital_employed
TOL_to_TNW
Total_term_liabilities__to__tangible_net_worth
Contingent_liabilities__to__Net_worth_perc
Net_fixed_assets
Current_assets
Net_working_capital
Quick_ratio_times
Current_ratio_times
Debt_to_equity_ratio_times
Cash_to_current_liabilities_times
Cash_to_average_cost_of_sales_per_day
Creditors_turnover
Debtors_turnover
Finished_goods_turnover
WIP_turnover
Raw_material_turnover
Shares_outstanding
1e9
Equity_face_value
EPS

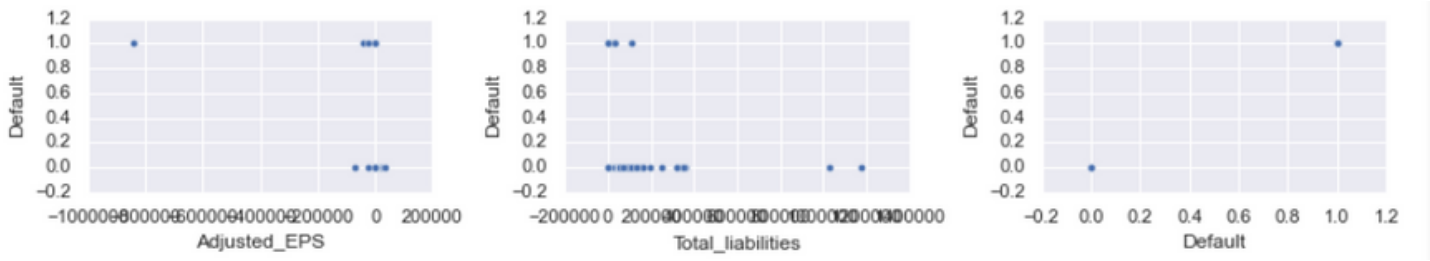


Fig: 5 Scatter Plot of Default VS All the Variables

Insights -

- For the upper range (-1000 – 0) of PBDITA as % of total income, chances are high that the customer will default.
- For the upper range (-4000 – 0) of PBT as % of total income, chances are high that the customer will default.
- For the upper range (-5000 -0) of PAT as % of total income, chances are high that the customer will default.
- For the upper range (-3000 – 0) of Cash profit as % of total income, chances are high that the customer will default.
- For the lower range (-500 – 0) of PAT as % of net worth , chances are high that the customer will default.
- Customers in range 0-200 of TOL_to_TNW seem more likely to default.
- Customers in range 0-300 of Total_term_liabilities seem more likely to default.
- Customers in range 0-4000 of Contingent_liabilities seem more likely to default.
- Customers in range 0-250 of Debt_to_equity_ratio_times seem more likely to default.
- Customers in range 0-500 of creditors_turnover seems more likely to default.
- Customers in range 0-1000 of debtors_turnover seems more likely to default.
- Customers in range 0-2500 of Finished_goods_turnover seems more likely to default.

Multivariate Analysis - Check for Multi- Collinearity

Heatmap -

A correlation heatmap uses colored cells, typically in a monochromatic scale, to show a 2D correlation matrix (table) between two discrete dimensions or event types.Correlation heatmaps are ideal for comparing the measurement for each pair of dimension values.

Darker Shades have higher Correlation , while lighter shades have smaller values of Correlation as compared to darker shades values.Correlation values near to 1 or -1 are highly positively correlated and highly negatively correlated respectively. Correlation values near to 0 are not correlated to each other.

Checking for Correlations -

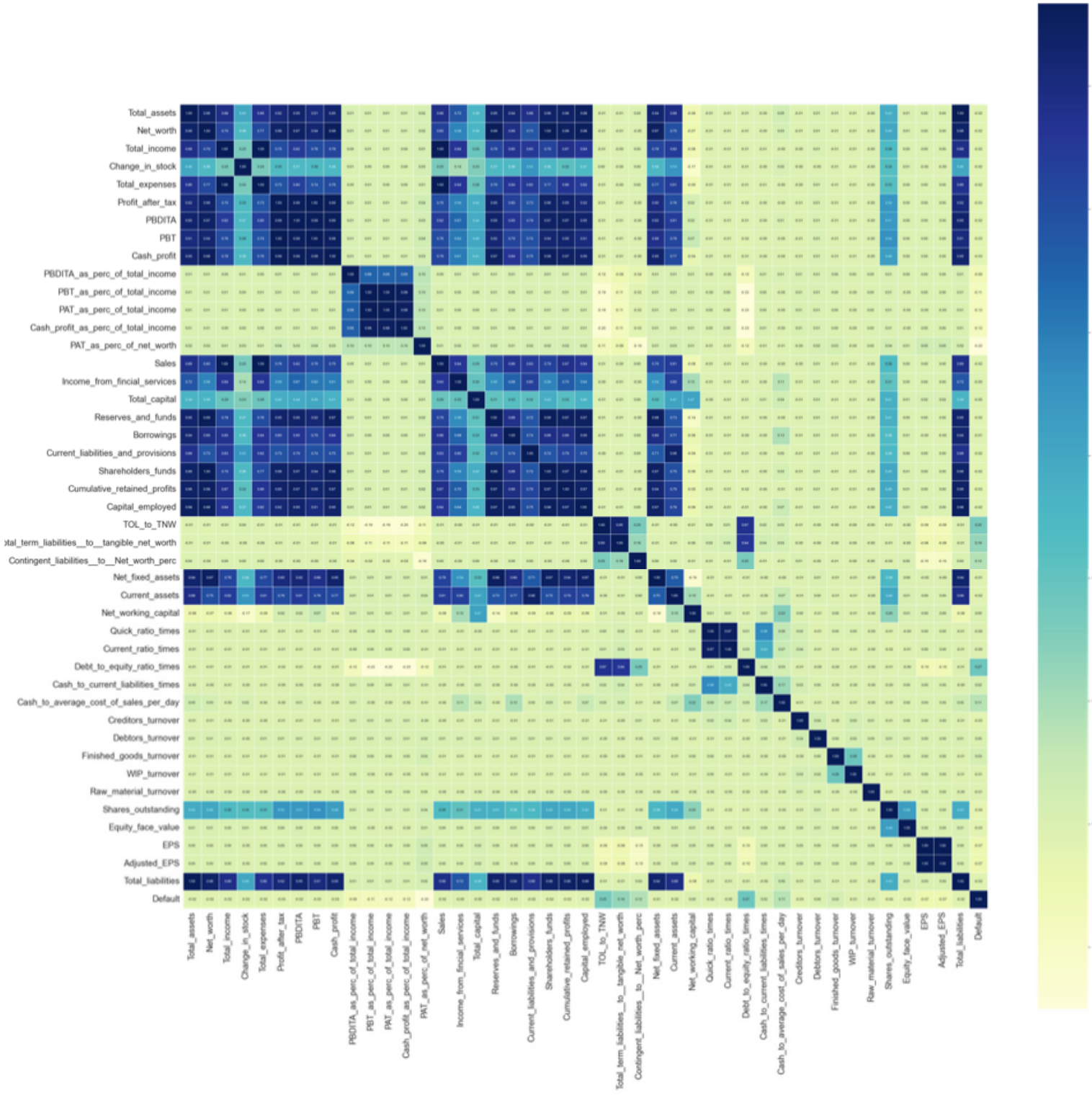


Fig: 6 Heat Map of Variables to Check Multicollinearity

Insights:

High correlation is present between the variables, which leads to multi-collinearity.

Perfect correlation is present between these variables:

- Net_worth - Total_assets
- Total_income - Total_assets, Net_worth
- Total_expenses - Total_assets, Net_worth, Total_income
- Profit_after_tax - Total_assets, Net_worth, Total_income, Total_expenses

- Total_liabilities - Total_assets, Net_worth, Total_income, Total_expenses, Profit_after_tax, PBDITA, PBT, Cash_profit, Sales, Other_income, Borrowings, Current_liabilities_and_provisions, Deferred_tax_liability, Shareholders_funds, Cumulative_retained_profits, Capital_employed

We'll use VIF as our feature reduction technique to get the most important features for the model.

Null Values Treatment -

As we see above columns which have null % more than 30 % we drop them and continue with the columns which have null % values less 30%. Now , we will impute these null values by using Simple Imputer with median values i.e. as outliers is present in the data so median will be effective choice for null value imputation.

Checking for Null Values After Imputation

Total_assets	0
Net_worth	0
Total_income	0
Change_in_stock	0
Total_expenses	0
Profit_after_tax	0
PBDITA	0
PBT	0
Cash_profit	0
PBDITA_as_perc_of_total_income	0
PBT_as_perc_of_total_income	0
PAT_as_perc_of_total_income	0
Cash_profit_as_perc_of_total_income	0
PAT_as_perc_of_net_worth	0
Sales	0
Income_from_fincial_services	0
Total_capital	0
Reserves_and_funds	0
Borrowings	0
Current_liabilities_and_provisions	0
Shareholders_funds	0
Cumulative_retained_profits	0
Capital_employed	0

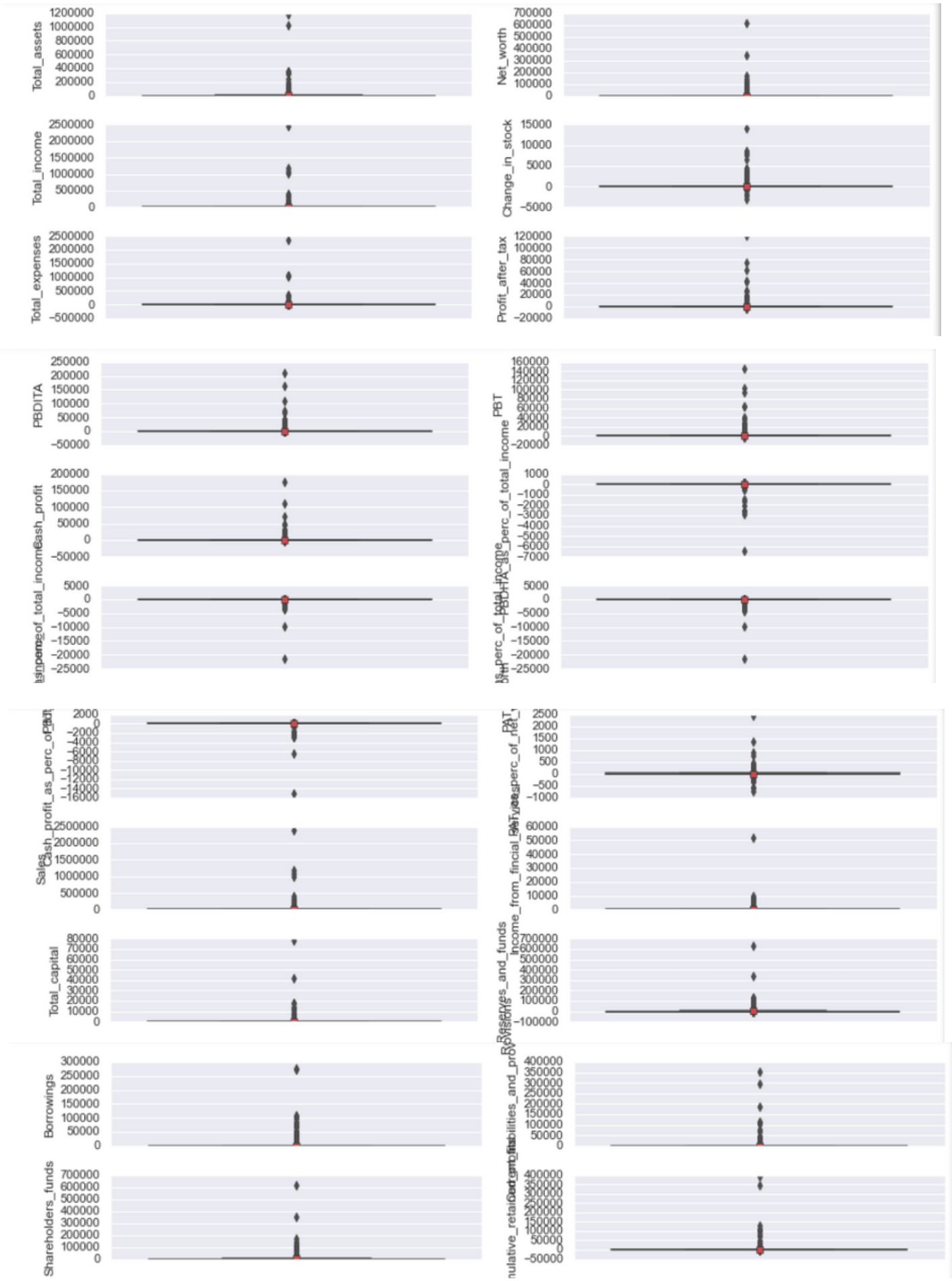
TOL_to_TNW	0
Total_term_liabilities__to__tangible_net_worth	0
Contingent_liabilities__to__Net_worth_perc	0
Net_fixed_assets	0
Current_assets	0
Net_working_capital	0
Quick_ratio_times	0
Current_ratio_times	0
Debt_to_equity_ratio_times	0
Cash_to_current_liabilities_times	0
Cash_to_average_cost_of_sales_per_day	0
Creditors_turnover	0
Debtors_turnover	0
Finished_goods_turnover	0
WIP_turnover	0
Raw_material_turnover	0
Shares_outstanding	0
Equity_face_value	0
EPS	0
Adjusted_EPS	0
Total_liabilities	0
Default	0

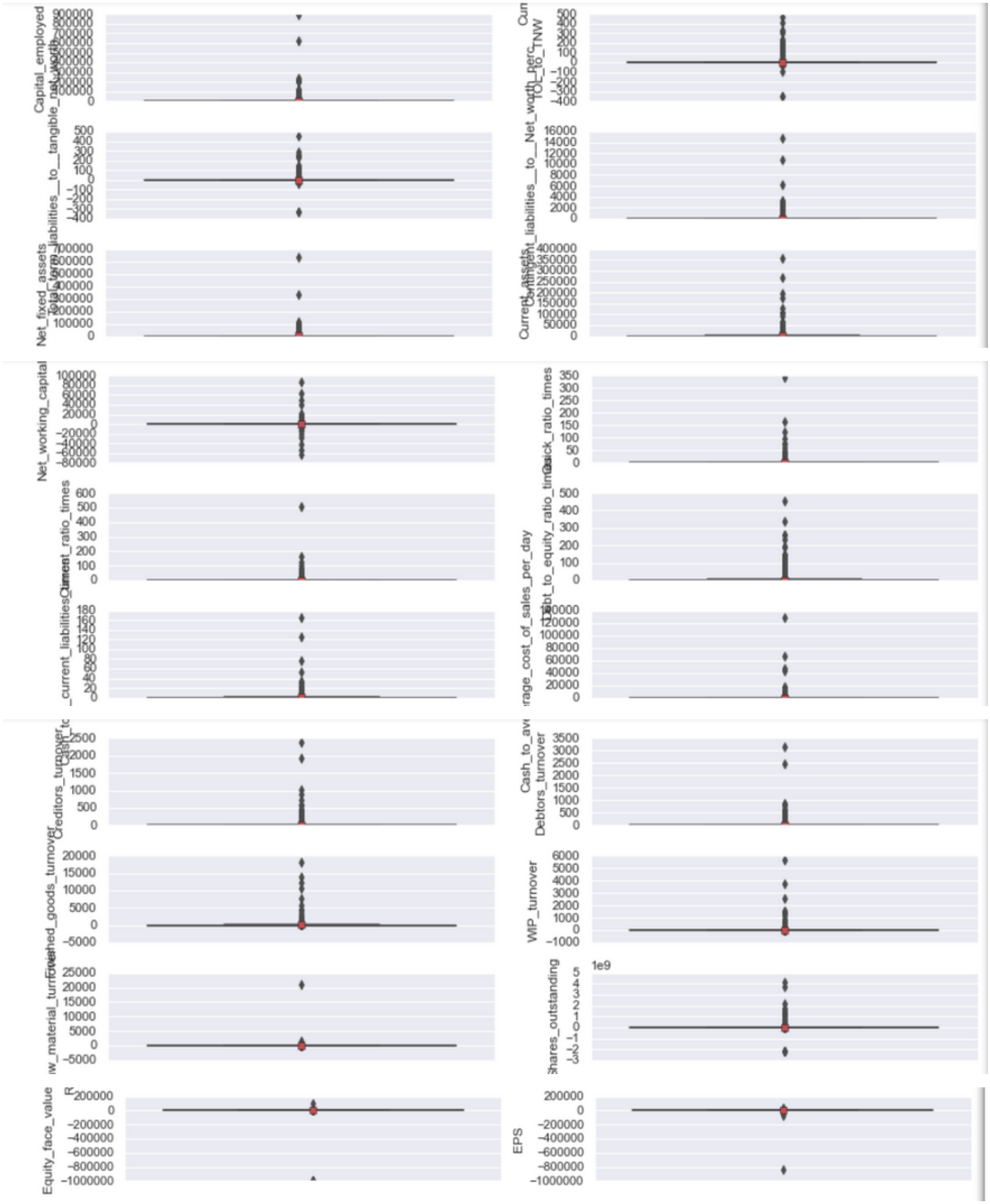
Tab: 11 Checking for Null Values After Imputation

Insights -

Now we don't have any null value present in the dataset which might effect our analysis.

Checking for Outliers in the dataset - To check for outliers, we will be plotting the box plots of all the varaivbles . Any value greater than Q3 + 1.5 IQR and less than Q1 - 1.5 IQR will be consider as outlier for that particular feature.


Total_assets
Net_worth
Total_income
Change_in_stock
Total_expenses
Profit_after_tax
PBDITA
PBT
Cash_profit
PBDITA_as_perc_of_total_income
Sales
PAT_as_perc_of_net_worth
Total_capital
Reserves_and_funds
Borrowings
Shareholders_funds


Capital_employed
Net_fixed_assets
Net_working_capital
Equity_face_value
TOL_to_TNW
Current_liabilities__to__Net_worth_perc
Quick_ratio_times
Debt_to_equity_ratio_times
Current_ratio_times
Cash_to_average_cost_of_sales_per_day
Creditors_turnover
Debtors_turnover
Finished_goods_turnover
WIP_turnover
Raw_material_turnover
Shares_outstanding
1e9
EPS

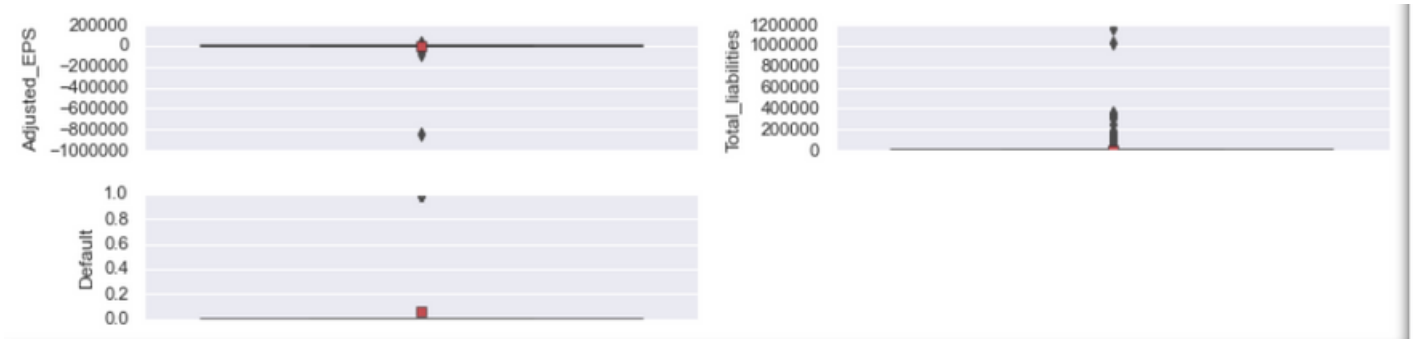


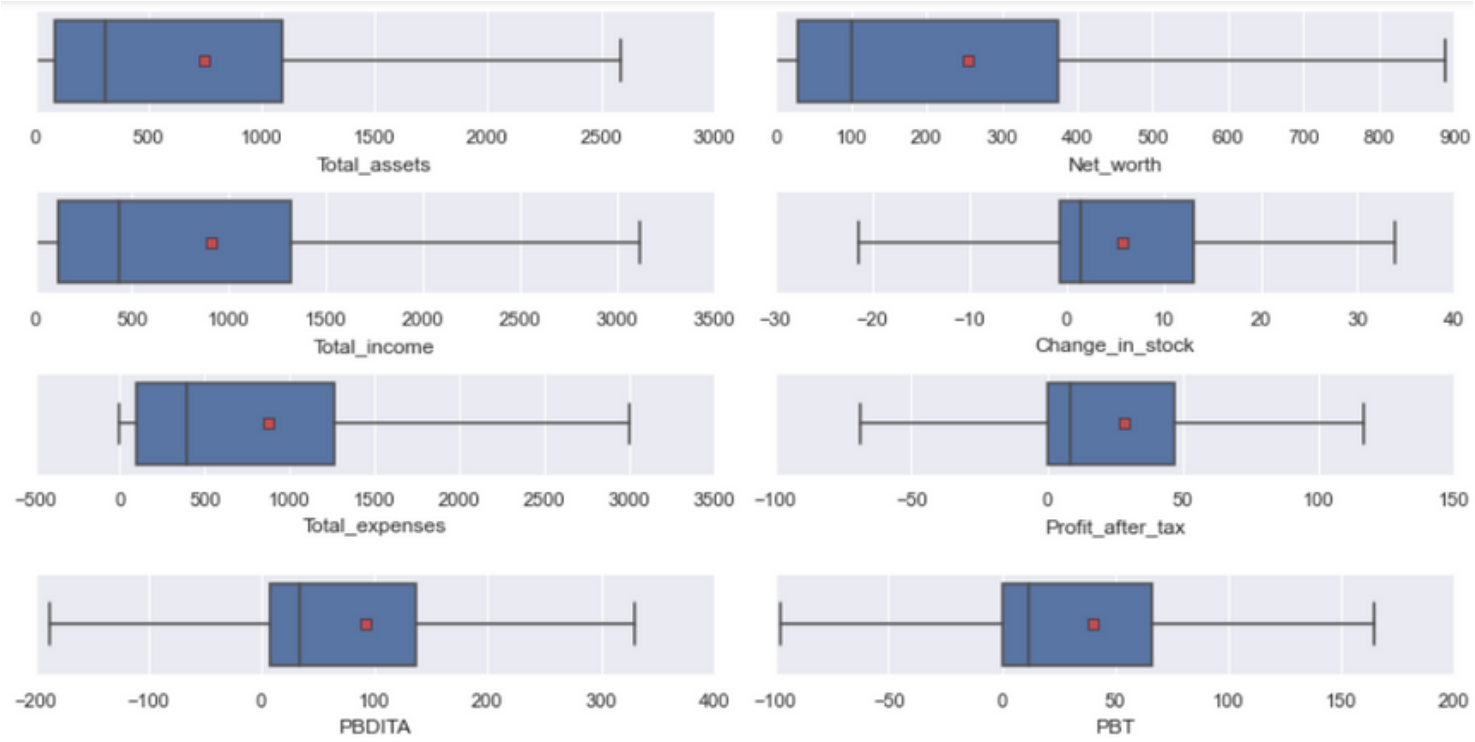
Fig: 7 Checking for Outliers in the dataset

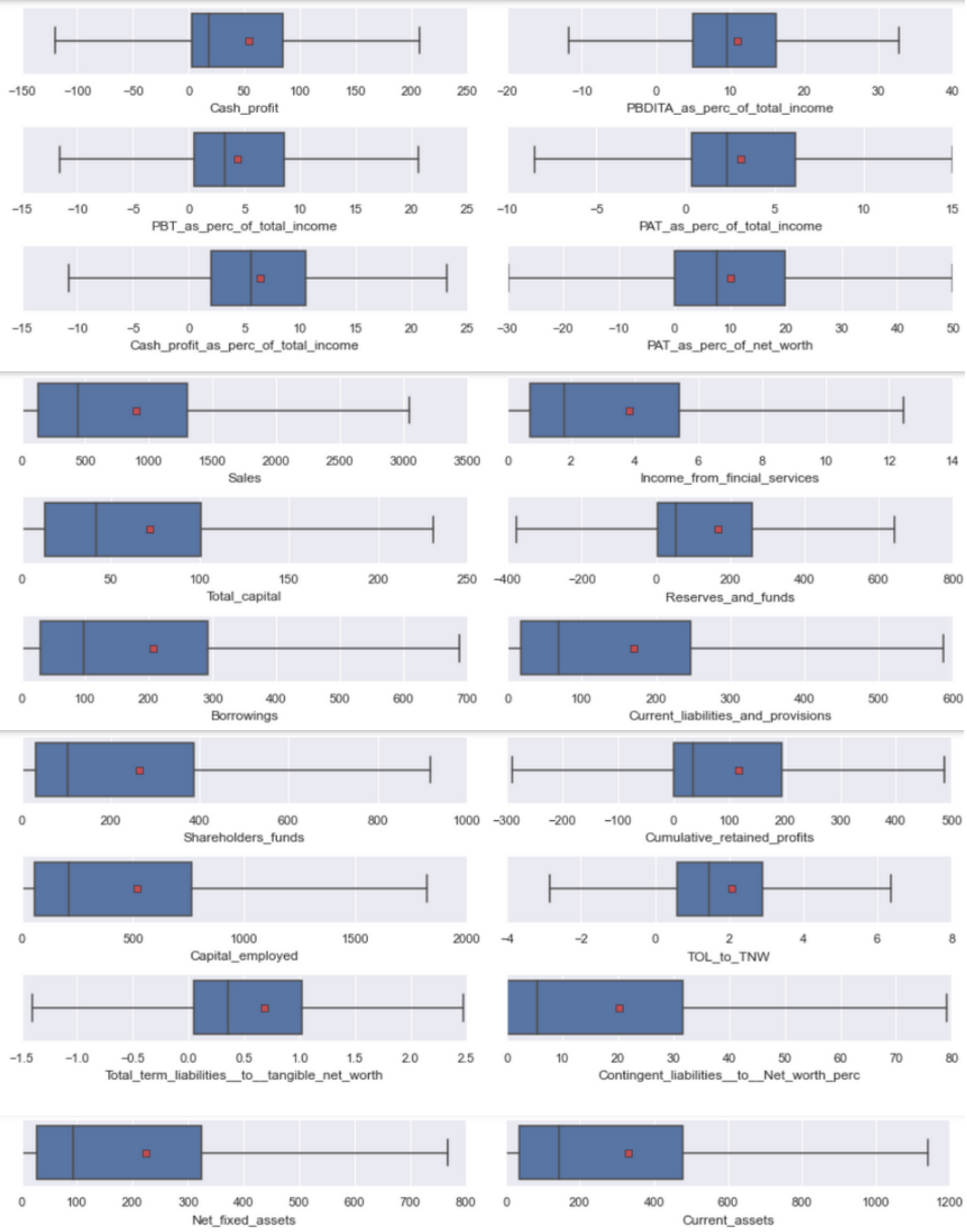
Observation -

- As we saw all the variables has outliers.Before outlier treatment we split x and y because we don't prefer outlier treatment of the dependent variable.Because it will effect the analysis.
- Outliers are unusual values in your dataset, and they can distort statistical analyses and violate their assumptions. Outliers increase the variability in your data, which decreases statistical power. Consequently, excluding outliers can cause your results to become statistically significant. That's why are doing the outliers treatment of independent variables.

Treatment of Outliers

We are caping the outliers with upper range and lower range of their respective box-plots.




Cash_profit
PBDITA_as_perc_of_total_income
PBT_as_perc_of_total_income
PAT_as_perc_of_total_income
Cash_profit_as_perc_of_total_income
PAT_as_perc_of_net_worth
Sales
Income_from_fincial_services
Total_capital
Reserves_and_funds
Borrowings
Current_liabilities_and_provisions
Shareholders_funds
Cumulative_retained_profits
Capital_employed
TOL_to_TNW
Total_term_liabilities__to__tangible_net_worth
Contingent_liabilities__to__Net_worth_perc
Net_fixed_assets
Current_assets

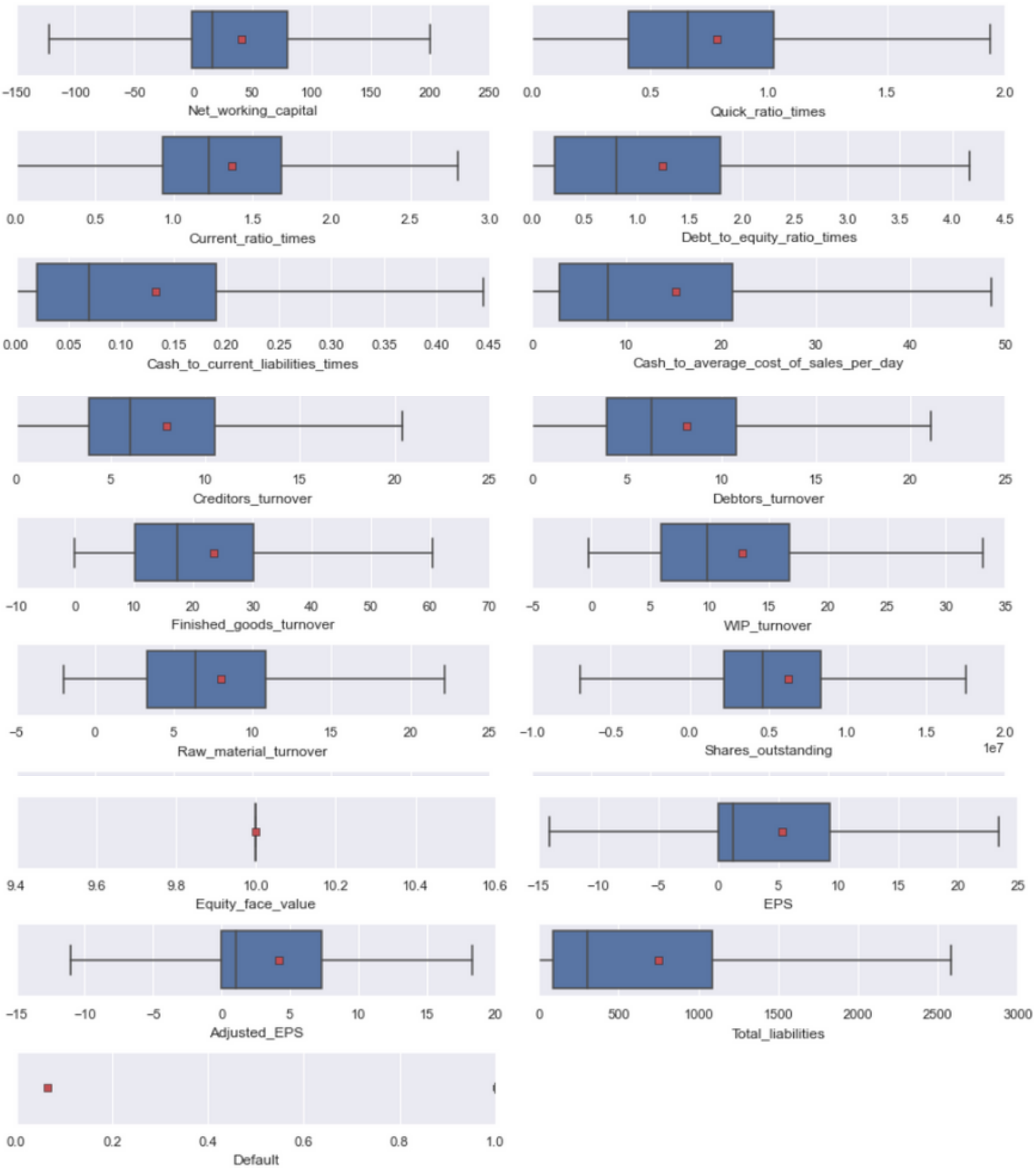


Fig: 8 Checking for Outliers in the dataset after Outlier Treatment

Results -

We have successfully treat the outliers , now we don't have any outliers present in the independent variables of the given dataset. Our data set is almost ready for the model building excercise.

Model Building -

Here , we are building various Machine Learning models like **Logistics Regression** , **Random Forest** , **Gradient Boosting** model with given data. So before building the machine learning models we need to split the data into train and test set in ratio (70:30).

Then We build the model on a train dataset and check the model performance measures on the validation dataset.

Proportion of 1s and 0s -

Proportion of 1s and 0s	Ratio
0	93.5
1	6.5

Tab: 12 Checking Proportion of 0s and 1s

Here , we saw that 1s are only 6.5 % and 0s are I the data set , which might create issue of class imbalance and creates our model overfit or underfit.

First , we create models on the given dataset and check the model performance on train and test set. If performance is as per standards then we will apply SMOTE on the training data set to get rid of class imbalance problem . Again we build models on SMOTE data and validate the model performance on train and test set.

Extracting the target column into separate vectors for Training set and Test set :

```
X = df_1.drop("Default", axis=1)

y = df_1.pop("Default")
```

Train-Test Split for Machine Learning Model -

The train-test split is a technique for evaluating the performance of a machine learning algorithm. It can be used for classification or regression problems and can be used for any supervised learning algorithm. The procedure involves taking a dataset and dividing it into two subsets.

In the given problem, we are advised to split the training and the testing data in the ratio of (70: 30).Here we are split the data into train and test part , like x_train , x_test , train_labels & test_labels ,by using train_test_split func() from sk-learn library here , we are taking 70 % data for training and 30 % data for testing.

Dimensions of the Training and Test Data :

```
Shape of x_train :  (2513, 44)
Shape of x_test :  (1078, 44)
Shape of y_train :  (2513,)
Shape of y_test :  (1078,)
```

Tab : 13 Checking Dimensions of the Training and Test Data

Model 1 - Logistic Regression Using Statsmodel library.

Logistic regression is a statistical analysis method used to predict a data value based on prior observations of a data set. . A logistic regression model predicts a dependent data variable by analyzing the relationship between one or more existing independent variables.

It is a Machine Learning algorithm which is used for the classification problems, it is a predictive analysis algorithm and based on the concept of probability.The hypothesis of logistic regression tends it to limit the cost function between 0 and 1 .

In logistic regression, the dependent variable is a binary variable that contains data coded as 1 (yes, success, etc.) or 0 (no, failure, etc.). In other words, the logistic regression model predicts P(Y=1) as a function of X.

Logistic Regression uses a more complex cost function, this cost function can be defined as the '**Sigmoid function' or also known as the 'logistic function' -** In order to map predicted values to probabilities, we use the Sigmoid function. The function maps any real value into another value between 0 and 1. In machine learning, we use sigmoid to map predictions to probabilities.

#Statsmodel requires the labelled data, therefore, concatinating the y label to the train set.

Creating Logistic Regression Model Using All Variables

```
import statsmodels.formula.api as SM

model_1 = SM.logit(formula = 'Default ~ Total_assets+ Net_worth + Total_income +
Change_in_stock + Total_expenses + Profit_after_tax + PBDITA + PBT + Cash_profit +
PBDITA_as_perc_of_total_income + PBT_as_perc_of_total_income +
PAT_as_perc_of_total_income + Cash_profit_as_perc_of_total_income +
PAT_as_perc_of_net_worth + Sales + Income_from_fincial_services + Total_capital +
Reserves_and_funds + Borrowings + Current_liabilities_and_provisions +
Shareholders_funds  + Cumulative_retained_profits + Capital_employed +
TOL_to_TNW + Total_term_liabilities__to__tangible_net_worth +
Contingent_liabilities__to__Net_worth_perc + Net_fixed_assets + Current_assets +
Net_working_capital  + Quick_ratio_times + Current_ratio_times +
Debt_to_equity_ratio_times + Cash_to_current_liabilities_times +
Cash_to_average_cost_of_sales_per_day  + Creditors_turnover  + Debtors_turnover
 + Finished_goods_turnover + WIP_turnover + Raw_material_turnover +
Shares_outstanding  + Equity_face_value  + EPS + Adjusted_EPS + Total_liabilities' ,
data =Default_train).fit()
```

#As we fit the tain data into the logistic regression model , now we look the summary of the logistic regression model and validate its performance on the train and test set.

Model 1 - Summary of Logistic Regression

Logit Regression Results

Dep. Variable:	Default	**No. Observations:**	2513
Model:	Logit	**Df Residuals:**	2468
Method:	MLE	**Df Model:**	44
Date:	Wed, 20 Jul 2022	**Pseudo R-squ.:**	0.4687
Time:	00:35:35	**Log-Likelihood:**	-320.64
converged:	False	**LL-Null:**	-603.48
Covariance Type:	nonrobust	**LLR p-value:**	9.347e-92

	coef	std err	z	P>|z|	[0.025	0.975]
Intercept	0.3827	nan	nan	nan	nan	nan
Total_assets	0.0011	nan	nan	nan	nan	nan
Net_worth	-0.0031	0.003	-1.136	0.256	-0.008	0.002
Total_income	0.0009	0.001	1.110	0.267	-0.001	0.003
Change_in_stock	0.0223	0.012	1.934	0.053	-0.000	0.045
Total_expenses	-0.0005	0.001	-0.569	0.569	-0.002	0.001
Profit_after_tax	-0.0030	0.016	-0.191	0.849	-0.034	0.028
PBDITA	-0.0005	0.003	-0.176	0.860	-0.006	0.005
PBT	0.0113	0.013	0.843	0.399	-0.015	0.038
Cash_profit	-0.0157	0.006	-2.559	0.011	-0.028	-0.004
PBDITA_as_perc_of_total_income	-0.0140	0.017	-0.850	0.396	-0.046	0.018
PBT_as_perc_of_total_income	-0.0166	0.074	-0.223	0.824	-0.163	0.129
PAT_as_perc_of_total_income	0.0195	0.099	0.197	0.844	-0.174	0.213
Cash_profit_as_perc_of_total_income	-0.0129	0.028	-0.468	0.640	-0.067	0.041
PAT_as_perc_of_net_worth	-0.0320	0.009	-3.433	0.001	-0.050	-0.014
Sales	-0.0006	0.001	-1.026	0.305	-0.002	0.001
Income_from_fincial_services	-0.0148	0.055	-0.270	0.787	-0.122	0.093
Total_capital	-0.0061	0.004	-1.574	0.115	-0.014	0.001
Reserves_and_funds	-0.0012	0.001	-0.860	0.390	-0.004	0.001
Borrowings	0.0018	0.002	0.881	0.378	-0.002	0.006
Current_liabilities_and_provisions	-0.0015	0.003	-0.516	0.606	-0.007	0.004
Shareholders_funds	0.0047	0.003	1.677	0.093	-0.001	0.010
Cumulative_retained_profits	-0.0086	0.003	-3.159	0.002	-0.014	-0.003
Capital_employed	-0.0036	0.002	-1.677	0.094	-0.008	0.001
TOL_to_TNW	0.1723	0.094	1.831	0.067	-0.012	0.357
Total_term_liabilities__to__tangible_net_worth	-0.2227	0.214	-1.039	0.299	-0.643	0.197
Contingent_liabilities__to__Net_worth_perc	0.0016	0.004	0.392	0.695	-0.006	0.009
Net_fixed_assets	0.0012	0.001	0.796	0.426	-0.002	0.004
Current_assets	-0.0001	0.002	-0.091	0.928	-0.003	0.003
Net_working_capital	0.0044	0.003	1.473	0.141	-0.001	0.010
Quick_ratio_times	-0.1181	0.451	-0.262	0.794	-1.003	0.767
Current_ratio_times	-0.7253	0.304	-2.387	0.017	-1.321	-0.130
Debt_to_equity_ratio_times	0.4134	0.120	3.455	0.001	0.179	0.648
Cash_to_current_liabilities_times	3.3188	1.306	2.540	0.011	0.758	5.879
Cash_to_average_cost_of_sales_per_day	-0.0078	0.009	-0.844	0.398	-0.026	0.010
Creditors_turnover	-0.0349	0.027	-1.290	0.197	-0.088	0.018
Debtors_turnover	-0.0129	0.024	-0.550	0.582	-0.059	0.033
Finished_goods_turnover	0.0042	0.010	0.408	0.683	-0.016	0.024
WIP_turnover	-0.0227	0.021	-1.088	0.276	-0.063	0.018

	coef	std err	z	P>|z|	[0.025	0.975]
Raw_material_turnover	-0.0124	0.021	-0.602	0.547	-0.053	0.028
Shares_outstanding	-3.123e-08	3.79e-08	-0.824	0.410	-1.06e-07	4.3e-08
Equity_face_value	-0.2173	nan	nan	nan	nan	nan
EPS	0.0766	0.103	0.746	0.456	-0.125	0.278
Adjusted_EPS	-0.1809	0.125	-1.444	0.149	-0.426	0.065
Total_liabilities	8.467e-05	nan	nan	nan	nan	nan

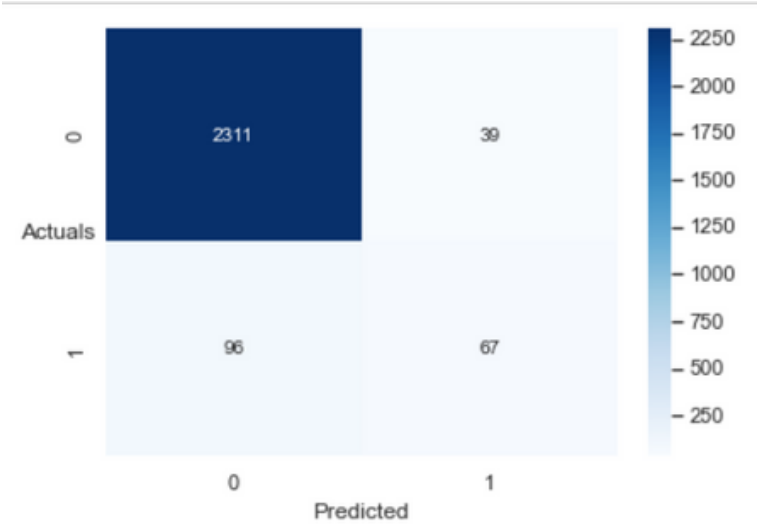
Tab : 14 Model 1 - Summary of Logistic Regression

Insights -

- Possibly complete quasi-separation: A fraction 0.20 of observations can be perfectly predicted. This might indicate that there is complete quasi-separation. In this case some parameters will not be identified.
- Here in the model 1 summary the value of **Pseudo R-squ. is 0.4687** as we know **pseudo R-squared value between of 0.2 to 0.4** indicates good fit.But here we get slightly higher value. So we do feature selection and check again the the value of Pseudo R-squ.
- **Log-Likelihood value is** -320.64 as we know Log Likelihood value is a measure of goodness of fit for any model. Higher the value, better is the model. We should remember that Log Likelihood can lie between -Inf to +Inf. Hence, the absolute look at the value cannot give any indication. We can only compare the Log Likelihood values between multiple models.So we again build the model on selected features and check the Log-Likelihood value.
- As we saw most of the variables p value is more than 0.05.As there is some kind of multicollinearity or not helpful for predicting the target variable.So we will do feature selection and choose the features and build the model again. Then check its summary and do the vailidation of the model.

Model 1 Evaluation on the Training Data -

Model 1 Confusion Matrix for the Training Data :



Train Data Accuracy

0.946

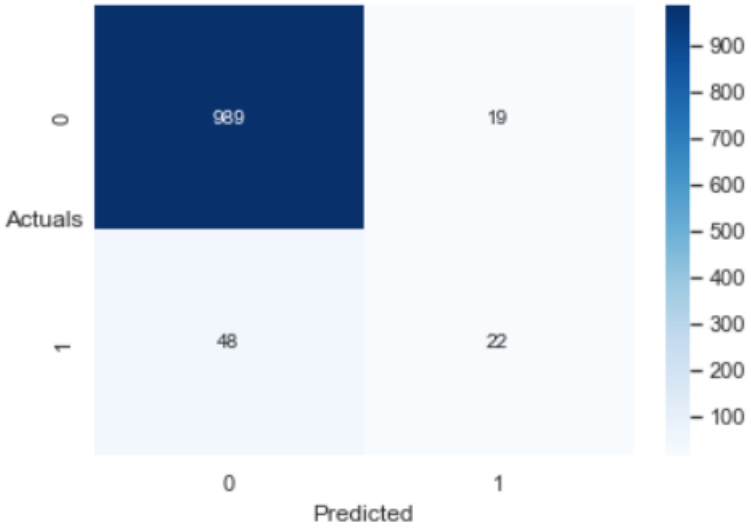
Fig: 9 Confusion Matrix Plot Logistic Regression Model 1 (Train Data)

Classification Report of Training Data :

	precision	recall	f1-score	support
0	0.960	0.983	0.972	2350
1	0.632	0.411	0.498	163
accuracy			0.946	2513
macro avg	0.796	0.697	0.735	2513
weighted avg	0.939	0.946	0.941	2513

Tab: 15 Model 1 Logistic Regression Classification Report Train Data

Model 1 Confusion Matrix for the Test Data :



Test Data Accuracy

0.938

Fig: 10 Confusion Matrix Plot Logistic Regression Model 1 (Test Data)

Classification Report of Test Data :

	precision	recall	f1-score	support
0	0.954	0.981	0.967	1008
1	0.537	0.314	0.396	70
accuracy			0.938	1078
macro avg	0.745	0.648	0.682	1078
weighted avg	0.927	0.938	0.930	1078

Tab: 16 Model 1 Logistic Regression Classification Report Test Data

Conclusion Model 1 Logistic Regression Model :

Train Data Class 0:

- Accuracy: 94.6%
- Precision: 96%
- Recall: 98.3%
- f1-Score: 97.2%

Test Data Class 0:

- Accuracy: 93.8%
- Precision: 95.4%
- Recall: 98.1%
- f1-Score: 96.7%

Train Data Class 1:

- Accuracy: 94.6%
- Precision: 63.2%
- Recall: 41.1%
- f1-Score: 49.8%

Test Data Class 1:

- Accuracy: 93.8%
- Precision: 53.7%
- Recall: 31.4%
- f1-Score: 39.6%

- On comparing the Train & Test results of the Model 1 Logistics Regression , we conclude that there is no problem of **underfitting or overfitting** of the model.
- But we have low **precision ,recall & f1 score** values class 1 as we know that proportion of class 1 is only 6.5 % in overall data set just because of this model learn the data but not giving too much good results for class 1.

Model 2 - Logistic Regression on (Selected Features) Using Statsmodel library.

Here we are using VIF to select the feature for our logistic regression model 2.Features whose VIF is less than 5 will consider for the logistic regression model and then will check the model performance on train and test data.

Feature Selection for Logistics Regression Model 2 Using VIF

	Variables	VIF
0	Total_assets	inf
43	Total_liabilities	inf
2	Total_income	220.52
4	Total_expenses	169.72
14	Sales	138.46
20	Shareholders_funds	120.90
1	Net_worth	109.34
22	Capital_employed	83.03
7	PBT	60.06
5	Profit_after_tax	57.55
10	PBT_as_perc_of_total_income	28.05
6	PBDITA	27.12
11	PAT_as_perc_of_total_income	26.51
8	Cash_profit	22.57
40	Equity_face_value	20.90
27	Current_assets	19.81
19	Current_liabilities_and_provisions	14.60
17	Reserves_and_funds	12.32
41	EPS	11.94
42	Adjusted_EPS	10.99
21	Cumulative_retained_profits	8.95
26	Net_fixed_assets	8.48
12	Cash_profit_as_perc_of_total_income	7.84
18	Borrowings	6.59
31	Debt_to_equity_ratio_times	6.50
9	PBDITA_as_perc_of_total_income	6.12
16	Total_capital	5.65
29	Quick_ratio_times	5.53
23	TOL_to_TNW	5.41
24	Total_term_liabilities__to__tangible_net_worth	5.15
30	Current_ratio_times	4.67
32	Cash_to_current_liabilities_times	4.46
39	Shares_outstanding	3.96
33	Cash_to_average_cost_of_sales_per_day	3.45
13	PAT_as_perc_of_net_worth	2.95
37	WIP_turnover	2.62
15	Income_from_fincial_services	2.52
36	Finished_goods_turnover	2.36
28	Net_working_capital	2.16
35	Debtors_turnover	2.03
34	Creditors_turnover	1.85
38	Raw_material_turnover	1.42
3	Change_in_stock	1.28
25	Contingent_liabilities__to__Net_worth_perc	1.27

Tab: 17 VIF Values for all the Variables

	Variables	VIF
30	Current_ratio_times	4.67
32	Cash_to_current_liabilities_times	4.46
39	Shares_outstanding	3.96
33	Cash_to_average_cost_of_sales_per_day	3.45
13	PAT_as_perc_of_net_worth	2.95
37	WIP_turnover	2.62
15	Income_from_fincial_services	2.52
36	Finished_goods_turnover	2.36
28	Net_working_capital	2.16
35	Debtors_turnover	2.03
34	Creditors_turnover	1.85
38	Raw_material_turnover	1.42
3	Change_in_stock	1.28
25	Contingent_liabilities__to__Net_worth_perc	1.27

Tab: 18 Features with VIF Values < 5

Observation -

As these features have VIF less than 5 , so will going to use these features in our logistics regression model.

Creating Logistic Regression Model 2 Using Selected Variables

```
import statsmodels.formula.api as SM

model_2 = SM.logit(formula = 'Default ~ Current_ratio_times +
Cash_to_current_liabilities_times + Shares_outstanding +
Cash_to_average_cost_of_sales_per_day + PAT_as_perc_of_net_worth + WIP_turnover
+ Income_from_fincial_services + Finished_goods_turnover + Net_working_capital +
Debtors_turnover + Creditors_turnover + Raw_material_turnover + Change_in_stock +
Contingent_liabilities__to__Net_worth_perc' , data =train_new).fit()
```

#As we fit the tain data into the logistic regression model 2 , now we look the summary of the logistic regression model and validate its performance on the train and test set.

Model 2 - Summary of Logistic Regression Model with Selected Features

Logit Regression Results

Dep. Variable:	Default	**No. Observations:**	2513
Model:	Logit	**Df Residuals:**	2498
Method:	MLE	**Df Model:**	14
Date:	Wed, 20 Jul 2022	**Pseudo R-squ.:**	0.3479
Time:	00:35:41	**Log-Likelihood:**	-393.55
converged:	True	**LL-Null:**	-603.48
Covariance Type:	nonrobust	**LLR p-value:**	8.287e-81

	coef	std err	z	P>|z|	[0.025	0.975]
Intercept	-1.3303	0.302	-4.412	0.000	-1.921	-0.739
Current_ratio_times	-0.8426	0.186	-4.534	0.000	-1.207	-0.478
Cash_to_current_liabilities_times	2.3734	1.040	2.281	0.023	0.334	4.412
Shares_outstanding	-1.873e-08	2.08e-08	-0.901	0.367	-5.95e-08	2.2e-08
Cash_to_average_cost_of_sales_per_day	-0.0043	0.008	-0.562	0.574	-0.019	0.011
PAT_as_perc_of_net_worth	-0.0884	0.006	-13.841	0.000	-0.101	-0.076
WIP_turnover	-0.0078	0.019	-0.406	0.685	-0.046	0.030
Income_from_fincial_services	-0.1009	0.037	-2.758	0.006	-0.173	-0.029
Finished_goods_turnover	-0.0017	0.010	-0.179	0.858	-0.021	0.017
Net_working_capital	-0.0015	0.002	-0.850	0.395	-0.005	0.002
Debtors_turnover	-0.0104	0.019	-0.533	0.594	-0.049	0.028
Creditors_turnover	-0.0309	0.024	-1.281	0.200	-0.078	0.016
Raw_material_turnover	-0.0142	0.019	-0.753	0.451	-0.051	0.023
Change_in_stock	0.0133	0.008	1.579	0.114	-0.003	0.030
Contingent_liabilities__to__Net_worth_perc	0.0075	0.003	2.235	0.025	0.001	0.014

Tab : 19 Model 2 - Summary of Logistic Regression with Selected Features

Insights -

- In the model 1 summary the value of **Pseudo R-squ. is 0.4687** as we know **pseudo R-squared value between of 0.2 to 0.4** indicates good fit.But here we get **Pseudo R-squ.: 0.347** which better than previous model.

- Previously in model 1 **Log-Likelihood value is** -320.64 as we know Log Likelihood value is a measure of goodness of fit for any model. Higher the value, better is the model. We should remember that Log Likelihood can lie between -Inf to +Inf. Hence, the absolute look at the value cannot give any indication. We can only compare the Log Likelihood values between multiple models. But here we get **Log-Likelihood value is around -393.55** which is less than model 1.

- As we saw here also many of the variables p value is more than 0.05.

Model 2 Evaluation on the Training Data -

Model 2 Confusion Matrix for the Training Data :

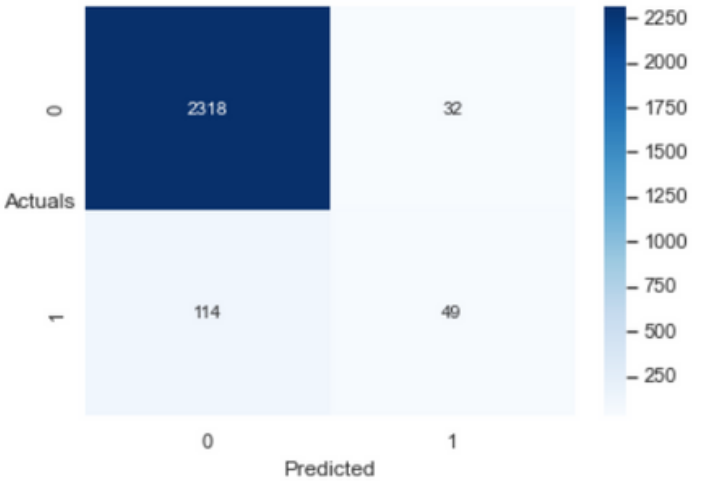


Fig: 11 Confusion Matrix Plot Logistic Regression Model 2 (Train Data)

Train Data Accuracy

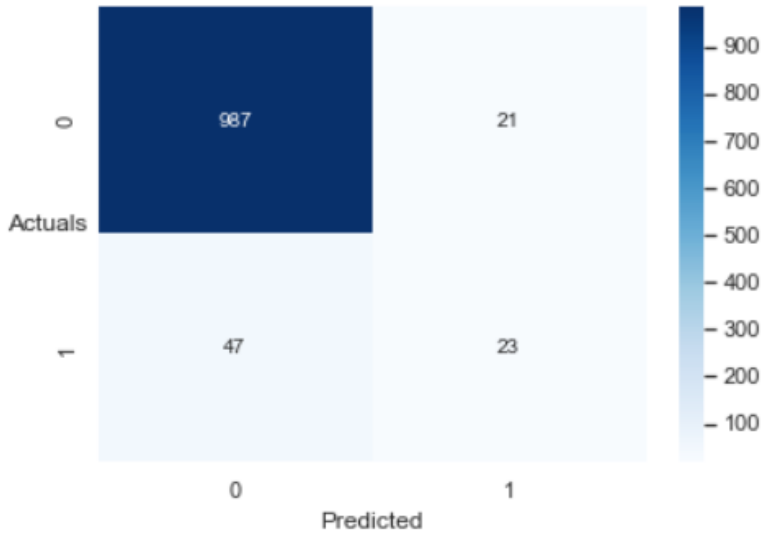
0.942

Classification Report of Training Data :

	precision	recall	f1-score	support
0	0.953	0.986	0.969	2350
1	0.605	0.301	0.402	163
accuracy			0.942	2513
macro avg	0.779	0.643	0.686	2513
weighted avg	0.931	0.942	0.933	2513

Tab: 20 Model 2 Logistic Regression Classification Report (Train Data)

Model 2 Confusion Matrix for the Test Data :



Test Data Accuracy

0.937

Fig: 12 Confusion Matrix Plot Logistic Regression Model 2 (Test Data)

Classification Report of Test Data :

	precision	recall	f1-score	support
0	0.955	0.979	0.967	1008
1	0.523	0.329	0.404	70
accuracy			0.937	1078
macro avg	0.739	0.654	0.685	1078
weighted avg	0.927	0.937	0.930	1078

Tab: 21 Model 2 Logistic Regression Classification Report Test Data

Conclusion Model 2 Logistic Regression Model :

Train Data Class 0:

- Accuracy: 94.2%
- Precision: 95.3%
- Recall: 98.6%
- f1-Score: 96.9%

Test Data Class 0:

- Accuracy: 93.7%
- Precision: 95.5%
- Recall: 97.9%
- f1-Score: 96.7%

Train Data Class 1:

- Accuracy: 94.2%
- Precision: 60.5%
- Recall: 30.1%
- f1-Score: 40.2%

Test Data Class 1:

- Accuracy: 93.7%
- Precision: 52.3%
- Recall: 32.9%
- f1-Score: 40.4%

- On comparing the Train & Test results of the Model 2 Logistics Regression , we conclude that there is no problem of **underfitting or overfitting** of the model.
- But we have low **precision ,recall & f1 score** values class 1 as we know that proportion of class 1 is only 6.5 % in overall data set just because of this model learn the data but not giving too much good results for class 1.

Model 3 - Random Base Forest

- A **Random forest** is a **machine learning technique** that's used to solve regression and classification problems. The (random forest) algorithm establishes the outcome based on the predictions of the decision trees.
- It builds decision trees on different samples and takes their majority vote for classification and average in case of regression.

Building Random Forest Base Model -

```
from sklearn.ensemble import RandomForestClassifier
rfcl = RandomForestClassifier()
rfcl = rfcl.fit(X_train, train_labels)
rfcl
```

We can create Random Forest Base Model by the help of sklearn lib by import Random ForestClassifier. Now fit train data into the model, check predictions on train and test data.

Predicting on Training and Test Dataset

rf_ytrain_predict

```
array([0, 0, 1, ..., 0, 0, 0])
```

rf_ytest_predict

```
array([0, 0, 0, ..., 0, 0, 0])
```

Getting the Predicted Probability :

Train Data -

	0	1
0	1.000	0.000
1	1.000	0.000
2	0.115	0.885
3	0.960	0.040
4	0.980	0.020

Test Data -

	0	1
0	1.00	0.00
1	1.00	0.00
2	1.00	0.00
3	0.99	0.01
4	1.00	0.00

Tab: 22 Random Forest Base Model Predicted Probability on the Train & Test Data

Model Evaluation - Random Forest Base Model

AUC and ROC for the Training Data

```
AUC: 0.999
Text(0, 0.5, 'True Positive Rate')
```

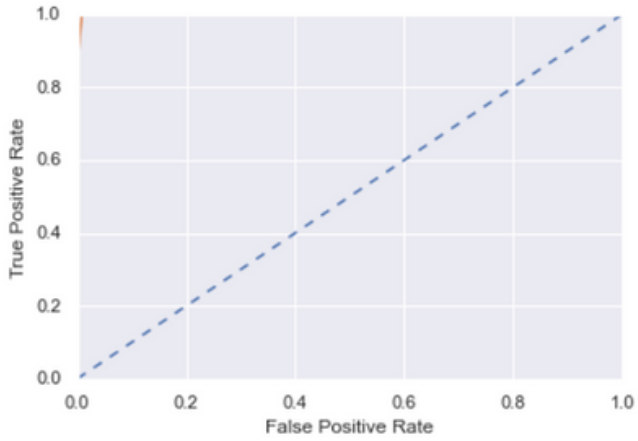


Fig: 13 AUC-ROC Curve Random Forest Base Model (Train Data)

Confusion Matrix for the Training Data

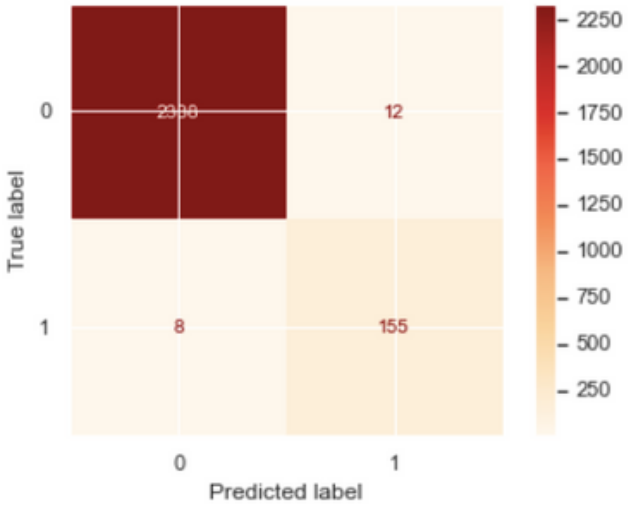


Fig : 14 Confusion Matrix Plot Random Forest Base Model (Train Data)

Classification Report of Training Data

	precision	recall	f1-score	support
0	1.00	0.99	1.00	2350
1	0.93	0.95	0.94	163
accuracy			0.99	2513
macro avg	0.96	0.97	0.97	2513
weighted avg	0.99	0.99	0.99	2513

Tab: 23 Random Forest Base Model Classification Report Train Data

AUC and ROC for the Test Data

```
AUC: 0.881
Text(0, 0.5, 'True Positive Rate')
```

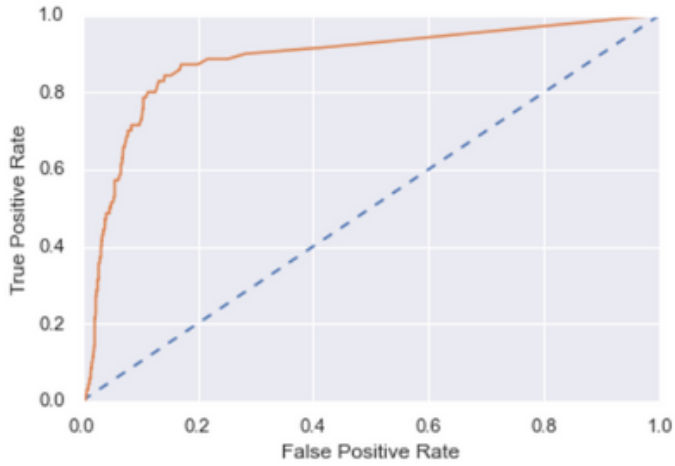


Fig: 15 AUC-ROC Curve Random Forest Base Model (Test Data)

Confusion Matrix for Test Data

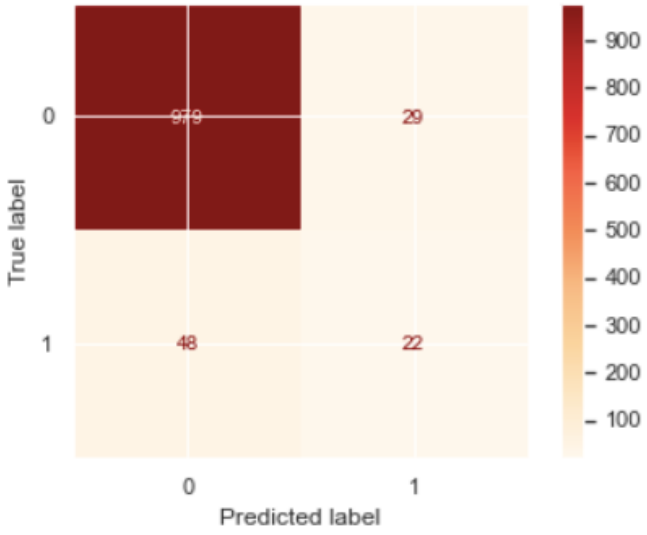


Fig: 16 Confusion Matrix Plot Random Forest Base Model (Test Data)

Classifiaction Report of Test Data :

	precision	recall	f1-score	support
0	0.95	0.97	0.96	1008
1	0.43	0.31	0.36	70
accuracy			0.93	1078
macro avg	0.69	0.64	0.66	1078
weighted avg	0.92	0.93	0.92	1078

Tab: 24 Random Forest Base Model Classification Report Test Data

Conclusion Random Forest Base Model :

Train Data Class 0:

- AUC: 99.9%
- Accuracy: 99%
- Precision: 100%
- Recall: 99%
- f1-Score: 100%

Test Data Class 0:

- AUC: 88.1%
- Accuracy: 93%
- Precision: 95%
- Recall: 97%
- f1-Score: 96%

Train Data Class 1:

- AUC: 99.9%
- Accuracy: 99%
- Precision: 93%
- Recall: 95%
- f1-Score: 94%

Test Data Class 1 :

- AUC: 88.1%
- Accuracy: 93%
- Precision: 43%
- Recall: 31%
- f1-Score: 36%

Variable Importance

	Imp
PAT_as_perc_of_net_worth	0.073348
Cash_profit	0.056442
Shareholders_funds	0.047698
TOL_to_TNW	0.046362
Net_worth	0.045567
Debt_to_equity_ratio_times	0.038242
Cumulative_retained_profits	0.032275
Reserves_and_funds	0.030821
Cash_profit_as_perc_of_total_income	0.029386
Profit_after_tax	0.029327

Tab: 25 Random Forest Base Model Variable Importance

Insights :

PAT_as_perc_of_net_worth , Cash_profit , Shareholders_funds, TOL_to_TNW , Net_worth ,Debt_to_equity_ratio_times ,etc are the important variables for predictions.

- On comparing the Train & Test results of the Random Forest Model , we conclude there is problem of **overfitting** of the model. As the difference between train and test precision recall more than 10% for class 1 so it is a overfit model.
- As **precision ,recall & f1 score** for train is very high and perform very poor on the test data for class 1 as we also know that no model have 100 % accurate in the practical world. Hence we reject this model as it is not a good model to predict results.
- Moreover as saw here in the model has overfit issue we can resolve this issue by applying bagging technique.So in the next model we apply bagging the same random forest and check its perfomance on train & test data.

Bagging of Random Forest - Bagging is an ensemble algorithm that fits multiple models on different subsets of a training dataset, then combines the predictions from all models. Random forest is an extension of bagging that also randomly selects subsets of features used in each data sample.

Bootstrap aggregating, also called bagging (from bootstrap aggregating), is a machine learning ensemble meta-algorithm designed to improve the stability and accuracy of machine learning algorithms used in statistical classification and regression. It also reduces variance and helps to avoid overfitting.Lets build Random Forest with Bagging technique.

Model - 4 Bagging of Random Forest

Building Bagged Random Forest Model -

```
from sklearn.ensemble import BaggingClassifier
from sklearn.ensemble import RandomForestClassifier

rf_bag = RandomForestClassifier()
Bagging_model=BaggingClassifier(base_estimator=rf_bag,n_estimators=100,random_state=1)
Bagging_model.fit(X_train,train_labels)
```

We can create Bagged Random Forest Model by the help of sklearn lib by import import RandomForestClassifier & import BaggingClassifier Now fit train data into the bagging classifier model & in base_estimator insert RandomForestClassifier().Now check predictions on train and test data.

Predicting the Training and Testing Data:

rf_bag_ytrain_predict

```
array([0, 0, 1, ..., 0, 0, 0])
```

rf_bag_ytest_predict

```
array([0, 0, 0, ..., 0, 0, 0])
```

Getting the Predicted Probability

Train Data -

	0	1
0	0.999500	0.000500
1	0.999300	0.000700
2	0.232319	0.767681
3	0.936767	0.063233
4	0.921368	0.078632

Test Data -

	0	1
0	0.990517	0.009483
1	0.999341	0.000659
2	0.997236	0.002764
3	0.995100	0.004900
4	0.999500	0.000500

Tab: 26 Random Forest Bagged Model Predicted Probability on the Train & Test Data

AUC and ROC for the Training Data

```
AUC: 0.998
Text(0, 0.5, 'True Positive Rate')
```

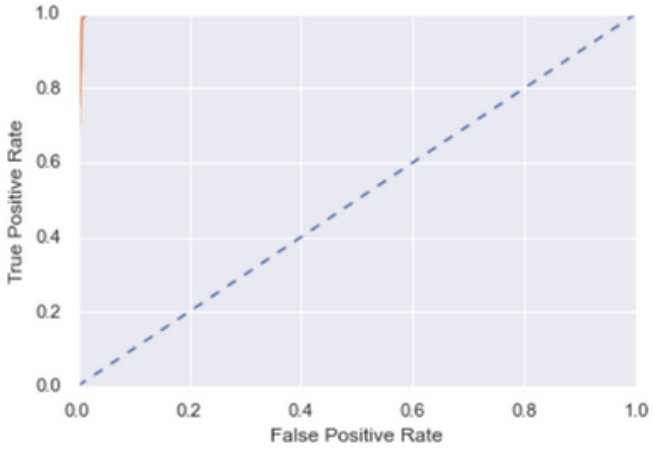


Fig: 17 AUC-ROC Curve Random Forest Bagged Model (Train Data)

Confusion Matrix for the Training Data

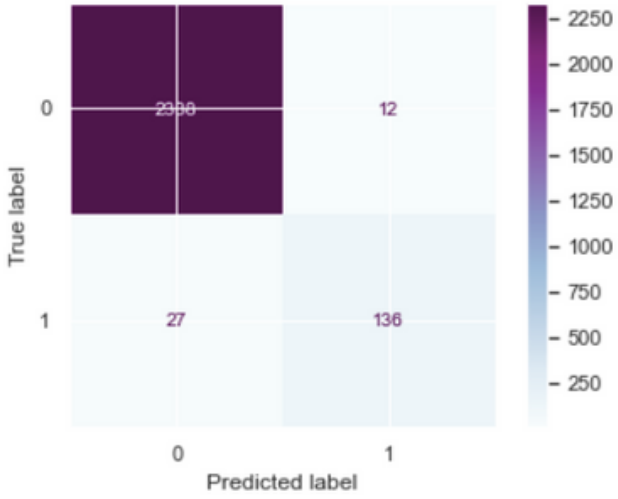


Fig 18 Confusion Matrix Plot Random Forest Bagged Model (Train Data)

Classification Report of Train Data

	precision	recall	f1-score	support
0	0.99	0.99	0.99	2350
1	0.92	0.83	0.87	163
accuracy			0.98	2513
macro avg	0.95	0.91	0.93	2513
weighted avg	0.98	0.98	0.98	2513

Tab: 27 Random Forest Bagged Model Classification Report Train Data

AUC and ROC for the Test Data

```
AUC: 0.902
Text(0, 0.5, 'True Positive Rate')
```

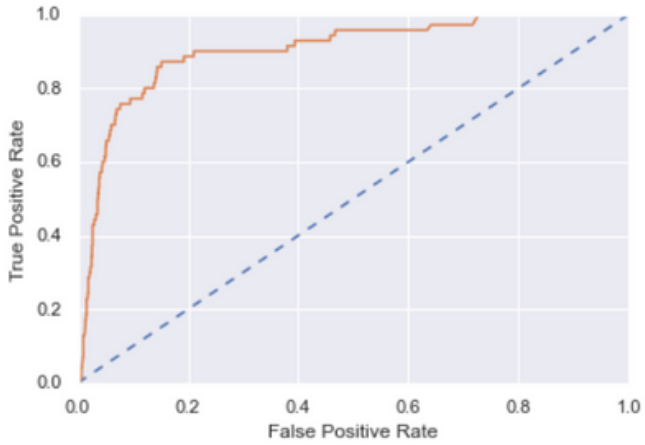


Fig : 19 AUC-ROC Curve Random Forest Bagged Model (Test Data)

Confusion Matrix for Test Data

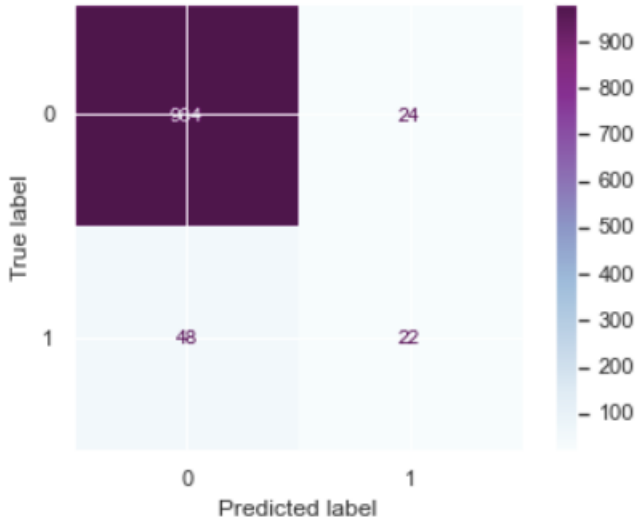


Fig : 20 Confusion Matrix Plot Random Forest Bagged Model (Test Data)

Classifiaction Report of Test Data

	precision	recall	f1-score	support
0	0.95	0.98	0.96	1008
1	0.48	0.31	0.38	70
accuracy			0.93	1078
macro avg	0.72	0.65	0.67	1078
weighted avg	0.92	0.93	0.93	1078

Tab: 28 Random Forest Bagged Model Classification Report Test Data

Conclusion Bagging of Random Forest

Train Data Class 0:

- AUC: 99.8%
- Accuracy: 98%
- Precision: 99%
- Recall: 99%
- f1-Score: 99%

Train Data Class 1:

- AUC: 99.8%
- Accuracy: 97%
- Precision: 92%
- Recall: 83%
- f1-Score: 87%

Test Data Class 0:

- AUC: 90.2%
- Accuracy: 93%
- Precision: 95%
- Recall: 98%
- f1-Score: 96%

Test Data Class 1:

- AUC: 90.2%
- Accuracy: 93%
- Precision:48%
- Recall: 31%
- f1-Score: 38%

Variable Importance

	Imp
PAT_as_perc_of_net_worth	0.073348
Cash_profit	0.056442
Shareholders_funds	0.047698
TOL_to_TNW	0.046362
Net_worth	0.045567
Debt_to_equity_ratio_times	0.038242
Cumulative_retained_profits	0.032275
Reserves_and_funds	0.030821
Cash_profit_as_perc_of_total_income	0.029386
Profit_after_tax	0.029327

Tab: 29 Random Forest Bagged Model Variable Importance

Insights :

PAT_as_perc_of_net_worth , Cash_profit , Shareholders_funds, TOL_to_TNW , Net_worth ,Debt_to_equity_ratio_times ,etc are the important variables for predictions.

- As **accuracy , precision ,recall & f1 score** for train is very high and perform very poor on the test data for class 1 as we have only 6.5 % proportion of class which lead t class imbalance.Hence we reject this model as it is not a good model to predict results.
- Hence we reject this model also as it is overfit and not a good model to predict results.

Model 5 Logistics Regression with sklearn.

Building Logistic Regression Model -

```
from sklearn.linear_model import LogisticRegression
model = LogisticRegression()
model.fit(X_train, train_labels)
```

We can create Logistic Regression model by the help of sklearn lib by import import Logistic Regression. Now fit train data into the model, check predictions on train and test data.

Predicting on Training and Test Dataset :

lr_ytrain_predict

```
array([0, 0, 0, ..., 0, 0, 0])
```

lr_ytest_predict

```
array([0, 0, 0, ..., 0, 0, 0])
```

Getting the Predicted Probability For Train & Test Data :

Train Data -

	0	1
0	0.986601	0.013399
1	0.999998	0.000002
2	0.910885	0.089115
3	0.999994	0.000006
4	0.999769	0.000231

Test Data -

	0	1
0	0.756623	0.243377
1	0.998915	0.001085
2	0.837883	0.162117
3	0.638727	0.361273
4	0.827063	0.172937

Tab: 30 Logistic Regression Predicted Probability on the Train & Test Data

Model Evaluation-Logistic Regression :

AUC and ROC for the Training Data :

```
AUC: 0.704
Text(0, 0.5, 'True Positive Rate')
```

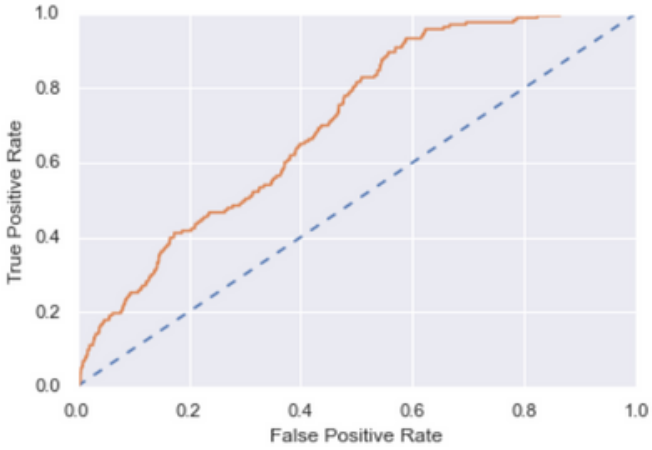


Fig: 21 AUC-ROC Curve Logistic Regression Model (Train Data)

Confusion Matrix for the Training Data :

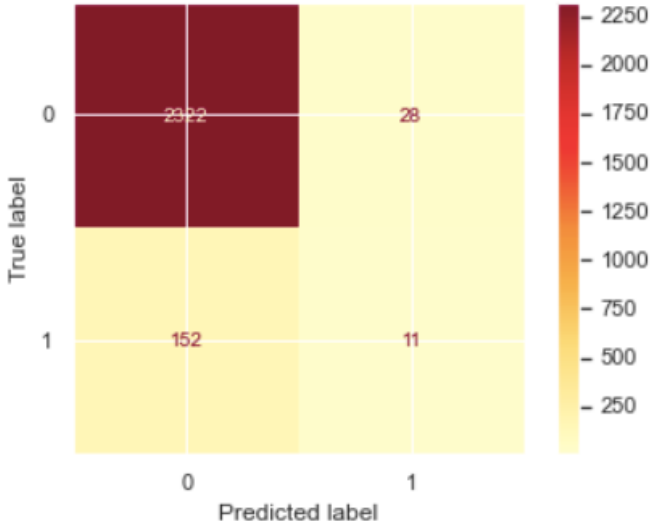


Fig: 22 Confusion Matrix Plot Logistic Regression Model (Train Data)

Classification Report of Training Data :

	precision	recall	f1-score	support
0	0.94	0.99	0.96	2350
1	0.28	0.07	0.11	163
accuracy			0.93	2513
macro avg	0.61	0.53	0.54	2513
weighted avg	0.90	0.93	0.91	2513

Tab: 31 Logistic Regression Classification Report Train Data

AUC and ROC for the Test Data :

```
AUC: 0.738
[<matplotlib.lines.Line2D at 0x7f8d4f6aa640>]
```

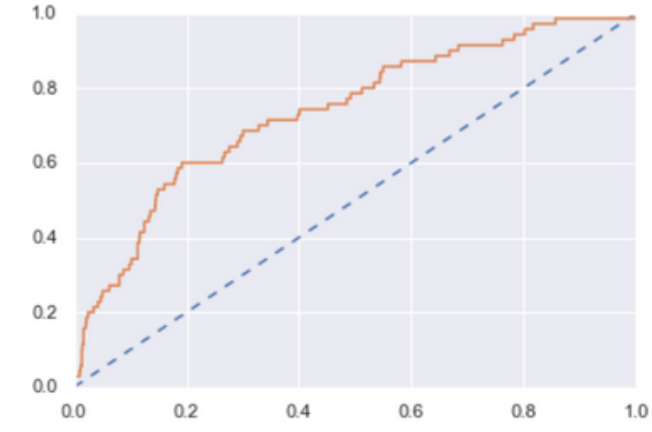


Fig: 23 AUC-ROC Curve Logistic Regression Model (Test Data)

Confusion Matrix for Test Data

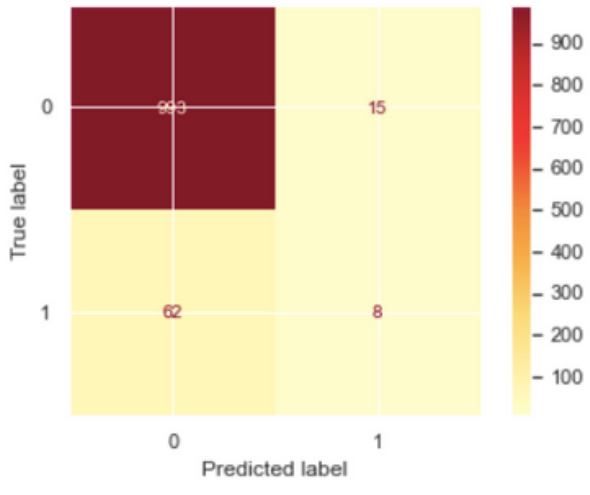


Fig: 24 Confusion Matrix Plot Logistic Regression Model (Test Data)

Classifiaction Report of Test Data :

	precision	recall	f1-score	support
0	0.94	0.99	0.96	1008
1	0.35	0.11	0.17	70
accuracy			0.93	1078
macro avg	0.64	0.55	0.57	1078
weighted avg	0.90	0.93	0.91	1078

Tab: 32 Logistic Regression Classification Report Test Data

Conclusion Logistic Regression Model :

Train Data Class 0:

- AUC: 70.4%
- Accuracy: 93%
- Precision: 94%
- Recall: 99%
- f1-Score: 96%

Test Data Class 0:

- AUC: 73.8%
- Accuracy: 93%
- Precision: 94%
- Recall: 99%
- f1-Score: 96%

Train Data Class 1:

- AUC: 70.4%
- Accuracy: 93%
- Precision: 28%
- Recall: 7%
- f1-Score: 11%

Test Data Class 1:

- AUC: 73.8%
- Accuracy: 93%
- Precision: 35%
- Recall: 11%
- f1-Score: 17%

- On comparing the Train & Test results of the Logistics Regression Model , we conclude there is no problem of **underfitting or overfitting** of the model.
- As **precision ,recall & f1 score** are **very very poor for class 1** as compared to class 0. Hence model is not good to predict the results.

Model 6 - Gradient Boosting Model

- **Gradient Boosting Model-** Gradient boosting is a machine learning technique for regression, classification and other tasks, which produces a prediction model in the form of an ensemble of weak prediction models, typically decision trees.

- Gradient boosting is a type of machine learning boosting. It relies on the intuition that the best possible next model, when combined with previous models, minimizes the overall prediction error. The key idea is to set the target outcomes for this next model inorder to minimize the error.

- Gradient boosting algorithm can be used for predicting not only continuous target variable (as a Regressor) but also categorical target variable (as a Classifier). When it is used as a regressor, the cost function is Mean Square Error (MSE) and when it is used as a classifier then the cost function is Log loss.

Building Gradient Boosting Model :

```
from sklearn.ensemble import GradientBoostingClassifier

gbcl = GradientBoostingClassifier(random_state=1)
gbcl = gbcl.fit(X_train,train_labels)
```

We can create Gradient Boosting Model by the help of sklearn lib by import GradientBoostingClassifier.Now fit train data into the model, check predictions on train and test data.

Predicting the Training and Testing data

gbcl_ytrain_predict

```
array([0, 0, 1, ..., 0, 0, 0])
```

gbcl_ytest_predict

```
array([0, 0, 0, ..., 0, 0, 0])
```

Getting the Predicted Probability

Train Data -

	0	1
0	0.996742	0.003258
1	0.995473	0.004527
2	0.085649	0.914351
3	0.988301	0.011699
4	0.977060	0.022940

Test Data -

	0	1
0	0.990648	0.009352
1	0.997043	0.002957
2	0.993749	0.006251
3	0.996558	0.003442
4	0.996742	0.003258

Tab: 33 Gradient Boosting Model Predicted Probability on the Train & Test Data

Model Evaluation - Gradient Boosting Model

AUC and ROC for the Training Data

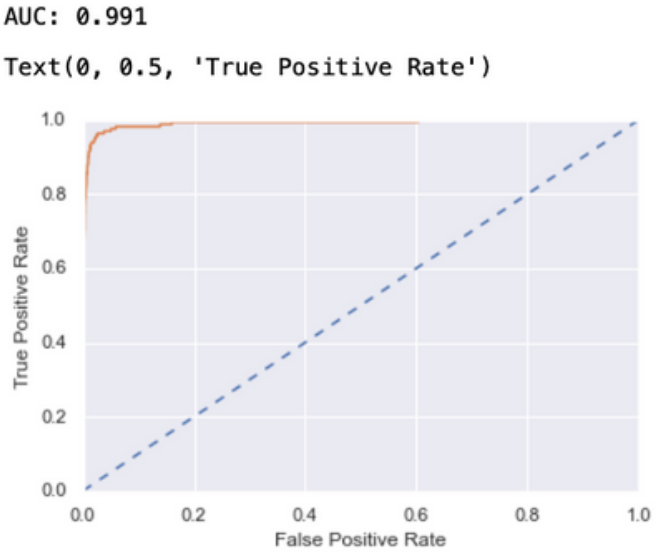


Fig: 25 AUC-ROC Curve Gradient Boosting Model (Train Data)

Confusion Matrix for the Training Data

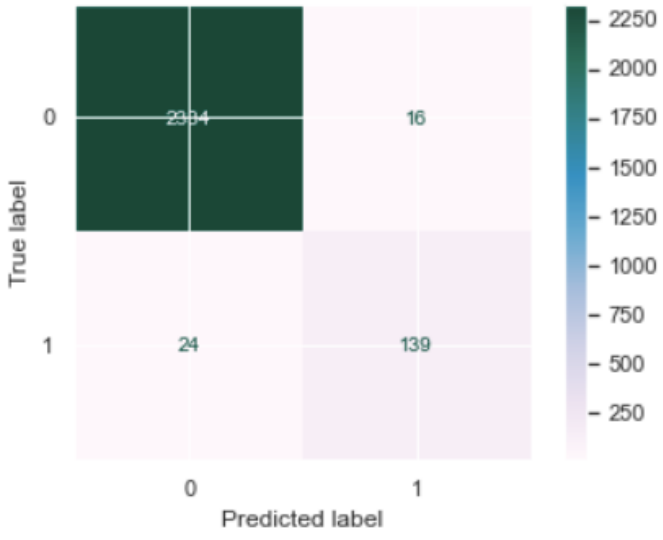


Fig: 26 Confusion Matrix Plot Gradient Boosting Model (Train Data)

Classification Report of Train Data

	precision	recall	f1-score	support
0	0.99	0.99	0.99	2350
1	0.90	0.85	0.87	163
accuracy			0.98	2513
macro avg	0.94	0.92	0.93	2513
weighted avg	0.98	0.98	0.98	2513

Tab: 34 Gradient Boosting Model Classification Report Train Data

AUC and ROC for the Test Data

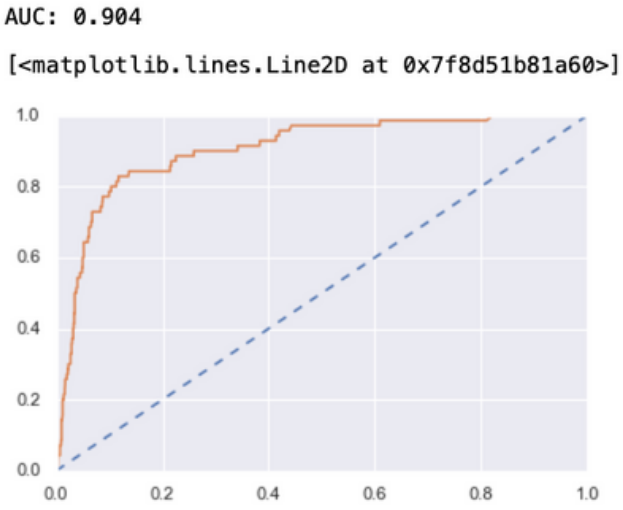


FIG: 27 AUC-ROC Curve Gradient Boosting Model (Test Data)

Confusion Matrix for Test Data

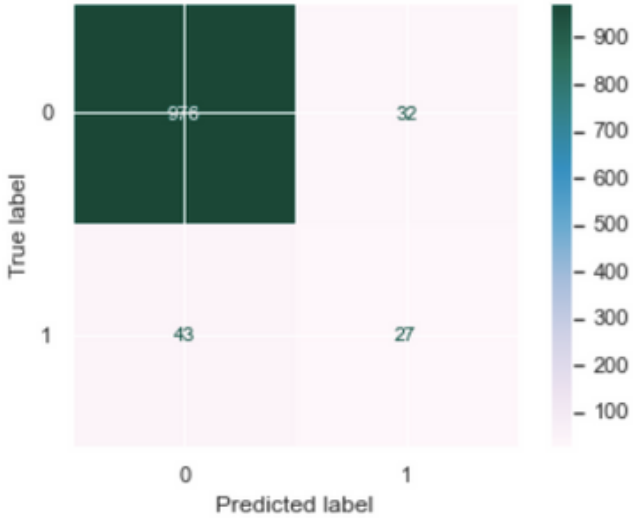


FIG: 28 Confusion Matrix Plot Gradient Boosting Model (Test Data)

Classifiaction Report of Test Data

	precision	recall	f1-score	support
0	0.96	0.97	0.96	1008
1	0.46	0.39	0.42	70
accuracy			0.93	1078
macro avg	0.71	0.68	0.69	1078
weighted avg	0.93	0.93	0.93	1078

Tab: 35 Gradient Boosting Model Classification Report Test Data

Conclusion Gradient Boost Model

Train Data Class 0:

- AUC: 99.1%
- Accuracy: 98%
- Precision: 99%
- Recall: 99%
- f1-Score: 99%

Test Data Class 0:

- AUC: 90.4%
- Accuracy: 93%
- Precision: 96%
- Recall: 97%
- f1-Score: 96%

Train Data Class 1:

- AUC: 99.1%
- Accuracy: 98%
- Precision: 90%
- Recall: 85%
- f1-Score: 87%

Test Data Class 1:

- AUC: 90.4%
- Accuracy: 93%
- Precision: 46%
- Recall: 39%
- f1-Score: 42%

- On comparing the Train & Test results of the Gradient Boost Model , we conclude there is problem of **overfitting** of the model. As the difference between train and test is precision recall and f1 more than 10% for class 1 so it is a overfit model.
- As **precision ,recall & f1 score** for train is very high and perform very poor on the test data for class 1 as we also know that there is class imbalance problem as proportion of class 1 is only 6.5 %.
- Now we can apply SMOTE and try to get rid of problem of class imbalance and check the model performance whether our model will perform better or not.

SMOTE

Synthetic Minority Oversampling Technique (SMOTE) is a statistical technique for increasing the number of cases in your dataset in a balanced way. The component works by generating new instances from existing minority cases that you supply as input.

Applying Smote on the Training Data

```
from imblearn.over_sampling import SMOTE

#SMOTE is only applied on the train data set
sm = SMOTE(random_state=2)
X_train_res,train_labels_res = sm.fit_resample(X_train, train_labels.ravel())
```

Shape after SMOTE

(4700, 44)

Model 7 - Logistic Regression Model with SMOTE

Building Logistic Regression Model with SMOTE -

```
model = LogisticRegression()
model.fit(X_train_res, train_labels_res)
```

We can create Logistic Regression model by the help of sklearn lib by import import Logistic Regression. Now fit new train data with SMOTE into the model, check predictions on train and test data.

Predicting on Training and Test Dataset :

sm_lr_ytest_predict

array([0, 0, 1, ..., 1, 0, 0])

sm_lr_ytrain_predict_

array([1, 1, 0, ..., 1, 1, 1])

Getting the Predicted Probability For Train & Test Data :

Train Data -

	0	1
0	0.515680	0.484320
1	0.966033	0.033967
2	0.422461	0.577539
3	0.930946	0.069054
4	0.854333	0.145667

Test Data -

	0	1
0	0.384820	0.615180
1	0.454250	0.545750
2	0.547519	0.452481
3	0.474133	0.525867
4	0.491276	0.508724

Tab: 36 Logistic Regression with SMOTE Predicted Probability on the Train & Test Data

Model Evaluation-Logistic Regression with SMOTE:

AUC and ROC for the Training Data :

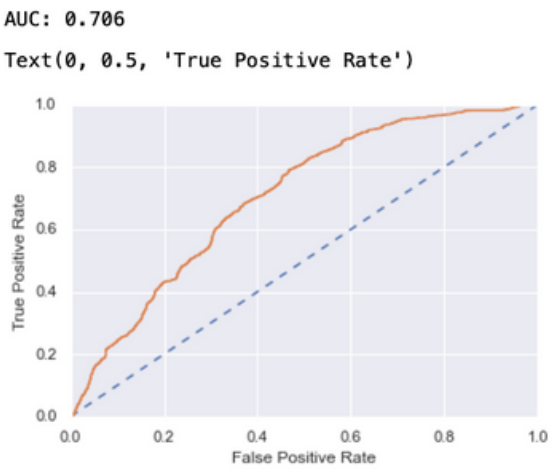


Fig : 29 AUC-ROC Curve Logistic Regression with SMOTE Model (Train Data)

Confusion Matrix for the Training Data :

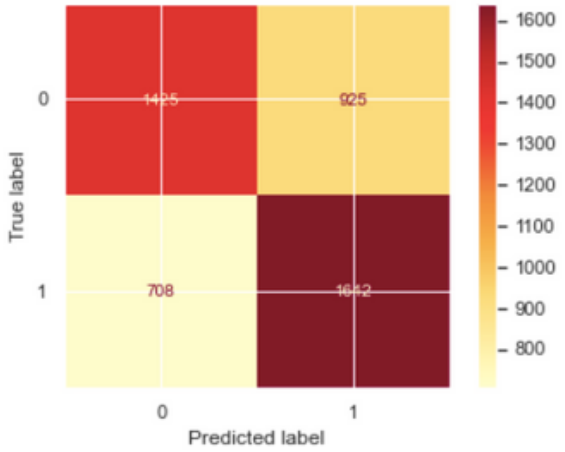


Fig : 30 Confusion Matrix Plot Logistic Regression with SMOTE Model (Train Data)

Classification Report of Training Data :

	precision	recall	f1-score	support
0	0.67	0.61	0.64	2350
1	0.64	0.70	0.67	2350
accuracy			0.65	4700
macro avg	0.65	0.65	0.65	4700
weighted avg	0.65	0.65	0.65	4700

Tab: 37 Logistic Regression with SMOTE Classification Report Train Data

AUC and ROC for the Test Data :

```
AUC: 0.654
[<matplotlib.lines.Line2D at 0x7f8d51c06e80>]
```

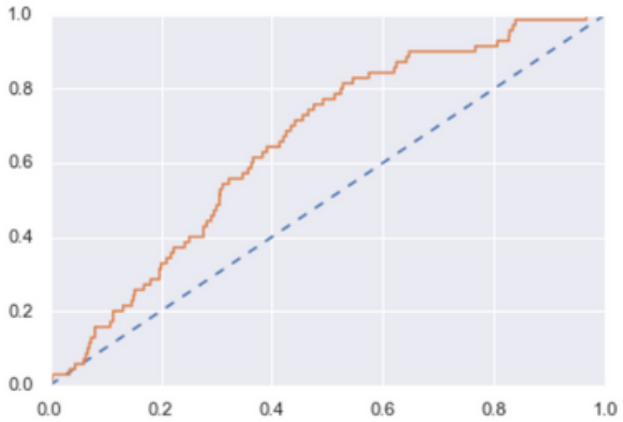


Fig : 31 AUC-ROC Curve Logistic Regression with SMOTE Model (Test Data)

Confusion Matrix for Test Data

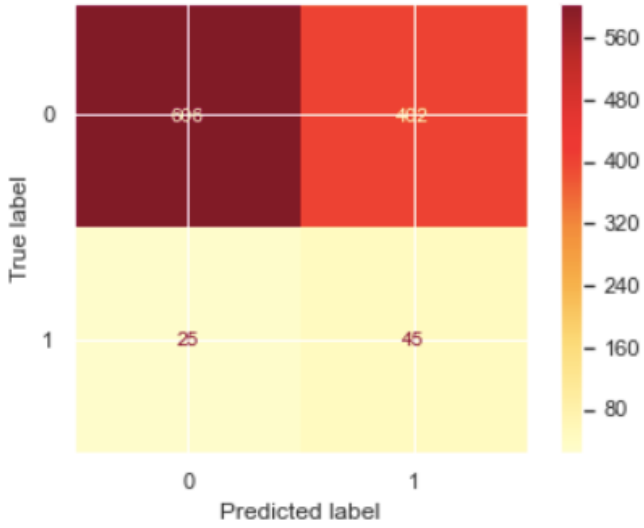


Fig : 32 Confusion Matrix Plot Logistic Regression with SMOTE Model (Test Data)

Classifiaction Report of Test Data :

	precision	recall	f1-score	support
0	0.96	0.60	0.74	1008
1	0.10	0.64	0.17	70
accuracy			0.60	1078
macro avg	0.53	0.62	0.46	1078
weighted avg	0.90	0.60	0.70	1078

Tab: 38 Logistic Regression with SMOTE Classification Report Test Data

Conclusion Logistic Regression with SMOTE Model :

Train Data Class 0:

- AUC: 70.6%
- Accuracy: 65%
- Precision: 67%
- Recall: 61%
- f1-Score: 64%

Test Data Class 0:

- AUC: 65.4%
- Accuracy: 60%
- Precision: 96%
- Recall: 60%
- f1-Score: 74%

Train Data Class 1:

- AUC: 70.6%
- Accuracy: 65%
- Precision: 64%
- Recall: 70%
- f1-Score: 67%

Test Data Class 1:

- AUC: 65.4%
- Accuracy: 60%
- Precision: 10%
- Recall: 64%
- f1-Score: 17%

- On comparing the Train & Test results of the Logistics Regression with SMOTE Model , we conclude model performance is very poor.
- As we look on **recall value** are quite similar for train and test data , but model is horrible it is performing very very poor in terms of precision and f1 score on test data for class 1.

Model 8 - Random Forest with SMOTE

Building Random Forest Model with SMOTE -

```
from sklearn.ensemble import RandomForestClassifier
sm_rfcl = RandomForestClassifier()
sm_rfcl = rfcl.fit(X_train_res, train_labels_res)
sm_rfcl
```

We can create Random Forest Model by the help of sklearn lib by import RandomForestClassifier. Now fit train data with SMOTE into the model, check predictions on train and test data.

Predicting on Training and Test dataset

sm_rf_ytrain_predict

```
array([0, 0, 1, ..., 1, 1, 1])
```

sm_rf_ytest_predict

```
array([0, 0, 0, ..., 0, 0, 0])
```

Getting the Predicted Probability

Train Data -

	0	1
0	1.00	0.00
1	1.00	0.00
2	0.01	0.99
3	0.96	0.04
4	0.93	0.07

Test Data -

	0	1
0	0.97	0.03
1	1.00	0.00
2	0.93	0.07
3	0.98	0.02
4	0.98	0.02

Tab: 39 Random Forest Model with SMOTE Predicted Probability on the Train & Test Data

Model Evaluation - Random Forest Model with SMOTE

AUC and ROC for the Training Data

```
AUC: 1.000
Text(0, 0.5, 'True Positive Rate')
```

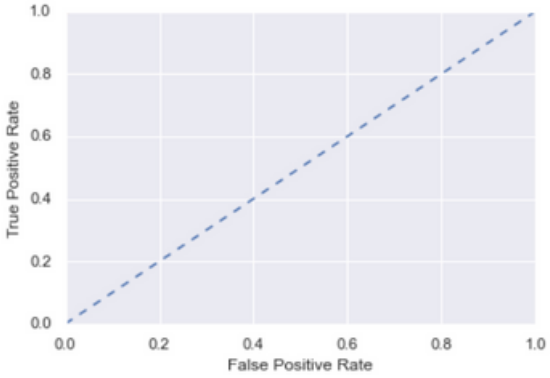


Fig: 33 AUC-ROC Curve Random Forest Model with SMOTE (Train Data)

Confusion Matrix for the Training Data

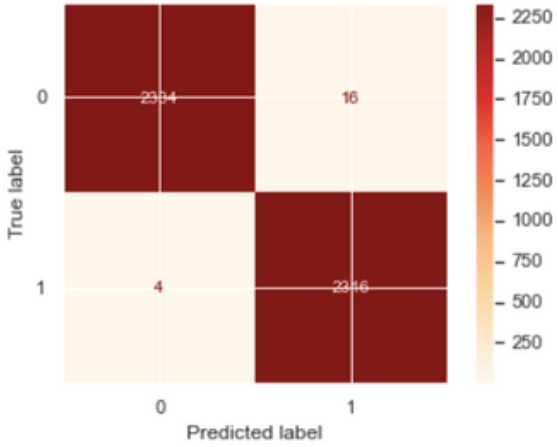


Fig: 34 Confusion Matrix Plot Random Forest Model with SMOTE (Train Data)

Classification Report of Training Data

	precision	recall	f1-score	support
0	1.00	0.99	1.00	2350
1	0.99	1.00	1.00	2350
accuracy			1.00	4700
macro avg	1.00	1.00	1.00	4700
weighted avg	1.00	1.00	1.00	4700

Tab: 40 Random Forest Model with SMOTE Classification Report Train Data

AUC and ROC for the Test Data

```
AUC: 0.875
Text(0, 0.5, 'True Positive Rate')
```

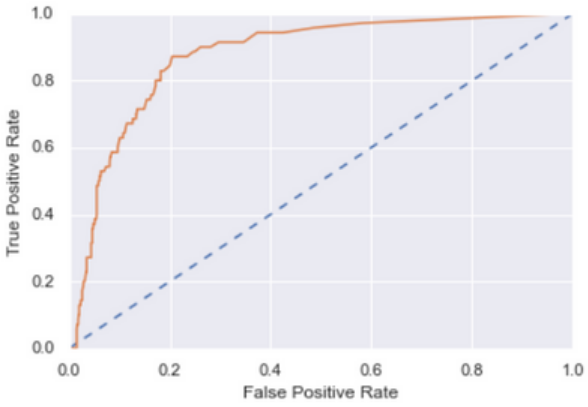


Fig: 35 AUC-ROC Curve Random Forest Model with SMOTE (Test Data)

Confusion Matrix for Test Data

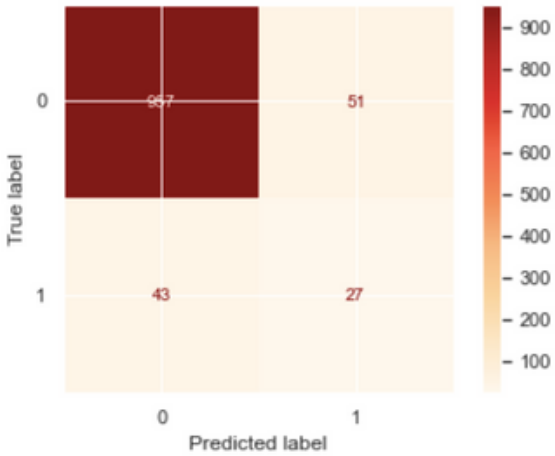


Fig: 36 Confusion Matrix Plot Random Forest Model with SMOTE (Test Data)

Classifiaction Report of Test Data :

	precision	recall	f1-score	support
0	0.96	0.95	0.95	1008
1	0.35	0.39	0.36	70
accuracy			0.91	1078
macro avg	0.65	0.67	0.66	1078
weighted avg	0.92	0.91	0.91	1078

Tab: 41 Random Forest Model with SMOTE Classification Report Test Data

Conclusion Random Forest Model with SMOTE :

Train Data Class 0:

- AUC: 100%
- Accuracy: 100%
- Precision: 100%
- Recall: 99%
- f1-Score: 100%

Test Data Class 0:

- AUC: 87.5%
- Accuracy: 91%
- Precision: 96%
- Recall: 95%
- f1-Score: 95%

Train Data Class 1:

- AUC: 100%
- Accuracy: 100%
- Precision: 99%
- Recall: 100%
- f1-Score: 100%

Test Data Class 1:

- AUC: 89.2%
- Accuracy: 91%
- Precision: 35%
- Recall: 39%
- f1-Score: 36%

Variable Importance

	Imp
PAT_as_perc_of_net_worth	0.102867
PBT	0.076979
EPS	0.068631
Cumulative_retained_profits	0.056609
Adjusted_EPS	0.050459
Cash_profit	0.050261

Tab: 42 Random Forest Model with SMOTE Variable Importance

Insights :

PAT_as_perc_of_net_worth, PBT ,EPS ,Cumulative_retained_profits , Adjusted_EPS , Cash_profit are the important variables for predictions.

- On comparing the Train & Test results of the Random Forest Model , we conclude there is problem of **overfitting** of the model. As the difference between train and test is precision recall more than 10% for class 1 so it is a overfit model.
- As **accuracy , precision ,recall & f1 score** for train is very high and perform very poor on the test data for class 1 as we also know that no model have 100 % accurate in the practical world. Hence we reject this model as it is not a good model to predict results.
- Moreover as saw here in the model has overfit issue we can resolve this issue by applying bagging technique.So in the next model we apply bagging the same random forest and check its perfomance on train & test data.

Model 9 - Bagging of Random Forest with SMOTE

Building Bagged Random Forest Model -

```
from sklearn.ensemble import BaggingClassifier
from sklearn.ensemble import RandomForestClassifier
sm_rf_bag = RandomForestClassifier()
sm_Bagging_model=BaggingClassifier(base_estimator=sm_rf_bag,n_estimators=100
,random_state=1)
sm_Bagging_model.fit(X_train_res,train_labels_res)
```

We can create Bagged Random Forest Model by the help of sklearn lib by import import RandomForestClassifier & import BaggingClassifier Now fit SMOTE train data into the bagging classifier model & in base_estimator insert RandomForestClassifier().Now check predictions on train and test data.

Predicting the Training and Testing data:

sm_rf_bag_ytrain_predict

array([0, 0, 1, ..., 1, 1, 1])

sm_rf_bag_ytest_predict

array([0, 0, 0, ..., 0, 0, 0])

Getting the Predicted Probability

Train Data -

	0	1
0	0.998000	0.002000
1	0.994100	0.005900
2	0.057157	0.942843
3	0.866146	0.133854
4	0.814500	0.185500

Test Data -

	0	1
0	0.902412	0.097588
1	0.998500	0.001500
2	0.851780	0.148220
3	0.986000	0.014000
4	0.965733	0.034267

Tab: 43 Random Forest Bagged Model with SMOTE Predicted Probability on the Train & Test Data

Model Evaluation - Bagging of Random Forest with SMOTE

AUC and ROC for the Training Data

```
AUC: 1.000
Text(0, 0.5, 'True Positive Rate')
```

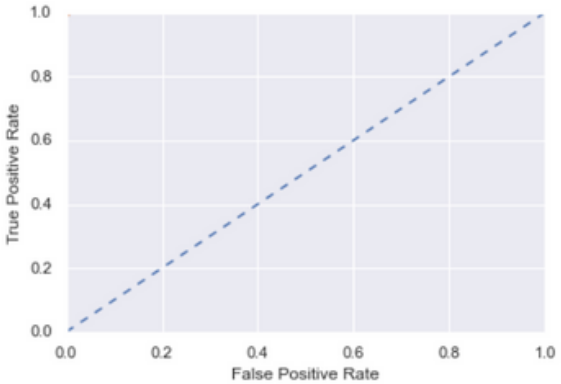


Fig : 37 AUC-ROC Curve Random Forest Bagged Model with SMOTE (Train Data)

Confusion Matrix for the Training Data

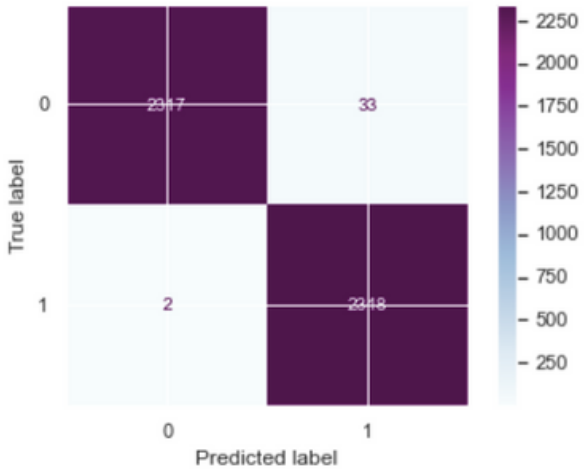


Fig: 38 Confusion Matrix Plot Random Forest Bagged Model with SMOTE (Train Data)

Classification Report of Train Data

	precision	recall	f1-score	support
0	1.00	0.99	0.99	2350
1	0.99	1.00	0.99	2350
accuracy			0.99	4700
macro avg	0.99	0.99	0.99	4700
weighted avg	0.99	0.99	0.99	4700

Tab: 44 Random Forest Bagged Model with SMOTE Classification Report Train Data

AUC and ROC for the Test Data

AUC: 0.894

Text(0, 0.5, 'True Positive Rate')

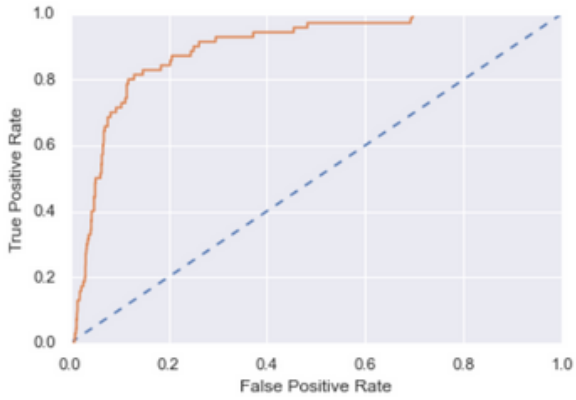


Fig: 39 AUC-ROC Curve Random Forest Bagged Model with SMOTE (Test Data)

Confusion Matrix for Test Data

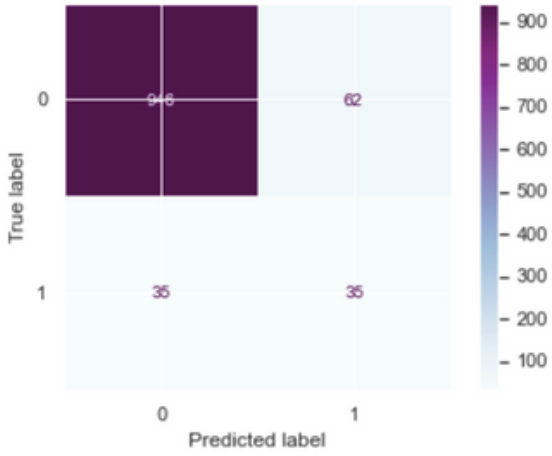


Fig : 40 Confusion Matrix Plot Random Forest Bagged Model with SMOTE (Test Data)

Classifiaction Report of Test Data

	precision	recall	f1-score	support
0	0.96	0.94	0.95	1008
1	0.36	0.50	0.42	70
accuracy			0.91	1078
macro avg	0.66	0.72	0.69	1078
weighted avg	0.93	0.91	0.92	1078

Tab: 45 Random Forest Bagged Model with SMOTE Classification Report Test Data

Conclusion Bagging of Random Forest with SMOTE

Train Data Class 0:

- AUC: 100%
- Accuracy: 99%
- Precision: 100%
- Recall: 99%
- f1-Score: 99%

Train Data Class 1:

- AUC: 99.7%
- Accuracy: 97%
- Precision: 99%
- Recall: 100%
- f1-Score: 99%

Test Data Class 0:

- AUC: 89.4%
- Accuracy: 91%
- Precision: 96%
- Recall: 94%
- f1-Score: 95%

Test Data Class 1:

- AUC: 89.4%
- Accuracy: 91%
- Precision: 36%
- Recall: 50%
- f1-Score: 42%

- On comparing the Train & Test results of the Bagged Random Forest Model with SMOTE , we conclude there is also problem of **overfitting** of the model.As the difference between train and test metrics for class 1 is more than 10% model perform poor on test data as compared to the train data , it is a overfit model too.
- Hence we reject this model also as it is not a good model to predict results.

Model 10 - Gradient Boosting Model with SMOTE

Building Gradient Boosting Model with SMOTE :

```
from sklearn.ensemble import GradientBoostingClassifier
sm_gbcl = GradientBoostingClassifier(random_state=1)
sm_gbcl = sm_gbcl.fit(X_train_res,train_labels_res)
```

We can create Gradient Boosting Model by the help of sklearn lib by import GradientBoostingClassifier.Now fit Smote train data into the model, check predictions on train and test data.

Predicting the Training and Testing data

sm_gbcl_ytrain_predict

array([0, 0, 1, ..., 1, 1, 1])

sm_gbcl_ytest_predict

array([0, 0, 0, ..., 0, 0, 0])

Getting the Predicted Probability

Train Data -

	0	1
0	0.980698	0.019302
1	0.984065	0.015935
2	0.024854	0.975146
3	0.956223	0.043777
4	0.943928	0.056072

Test Data -

	0	1
0	0.912892	0.087108
1	0.988396	0.011604
2	0.767304	0.232696
3	0.992899	0.007101
4	0.944093	0.055907

Tab: 46 Gradient Boosting Model with SMOTE Predicted Probability on the Train & Test Data

Model Evaluation - Gradient Boosting ModeL with SMOTE

AUC and ROC for the Training Data

Fig : 41 AUC-ROC Curve Gradient Boosting Model with SMOTE (Train Data)

Confusion Matrix for the Training Data

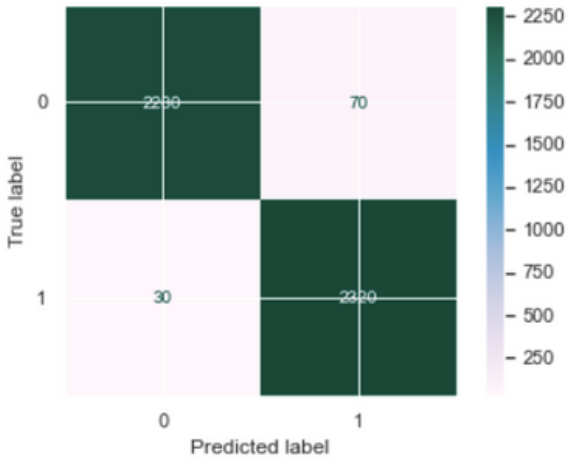


Fig: 42 Confusion Matrix Plot Gradient Boosting Model with SMOTE (Train Data)

Classification Report of Train Data

	precision	recall	f1-score	support
0	0.99	0.97	0.98	2350
1	0.97	0.99	0.98	2350
accuracy			0.98	4700
macro avg	0.98	0.98	0.98	4700
weighted avg	0.98	0.98	0.98	4700

Tab: 47 Gradient Boosting Model with SMOTE Classification Report Train Data

AUC and ROC for the Test Data

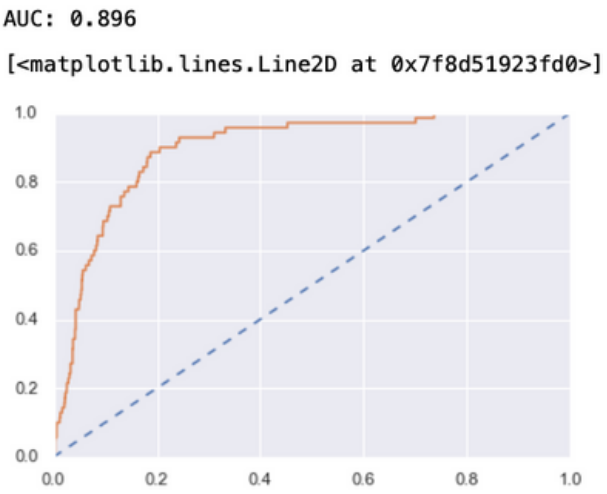


Fig: 43 AUC-ROC Curve Gradient Boosting Model with SMOTE (Test Data)

Confusion Matrix for Test Data

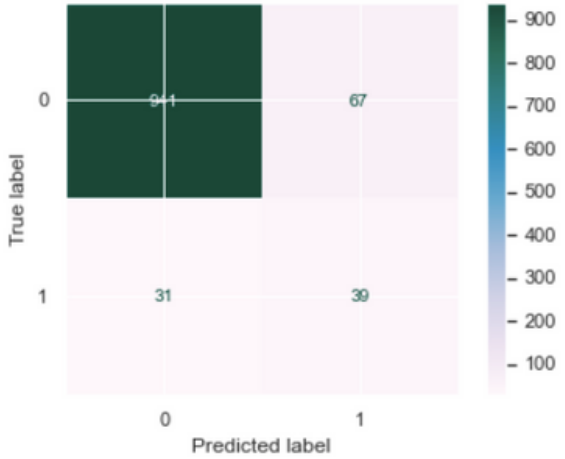


Fig: 44 Confusion Matrix Plot Gradient Boosting Model with SMOTE (Test Data)

Classifiaction Report of Test Data

	precision	recall	f1-score	support
0	0.97	0.93	0.95	1008
1	0.37	0.56	0.44	70
accuracy			0.91	1078
macro avg	0.67	0.75	0.70	1078
weighted avg	0.93	0.91	0.92	1078

Tab: 48 Gradient Boosting Model with SMOTE Classification Report Test Data

Conclusion Gradient Boost Model with SMOTE

Train Data Class 0:

- AUC: 99..7%
- Accuracy: 98%
- Precision: 99%
- Recall: 97%
- f1-Score: 98%

Test Data Class 0:

- AUC: 89.6%
- Accuracy: 91%
- Precision: 97%
- Recall: 93%
- f1-Score: 95%

Train Data Class 1:

- AUC: 99.7%
- Accuracy: 98%
- Precision: 97%
- Recall: 99%
- f1-Score: 98%

Test Data Class 1:

- AUC: 89.6%
- Accuracy: 91%
- Precision: 37%
- Recall: 56%
- f1-Score: 44%

- On comparing the Train & Test results of the Bagged Random Forest Model with SMOTE , we conclude their is also problem of **overfitting** of the model.As the difference between train and test metrics for class 1 is more than 10% model perform poor on test data as compared to the train data , it is a overfit model too.
- Hence we reject this model also as it is not a good model to predict results.

Comparison of the Performance Metrics of All the Models on Train Data and Test Data

All the models are built above with their train & test accuracy , AUC AND ROC curve and score and Confusion Matrix and classification report. Here we will compare all the models on basis of their performance,AUC/ROC Score , Accuracy , Precision ,Recall for train & test data with tabular representation below & decide the final model for the deployment.

In the given problem our is work to build Indian Credit Risk (Default) Model on the basis of given data in spreadsheet . We have to build a various machine learning models to predict which company is going to be Defaulter or Non-Defaulter on the basis of the given information.

According the context of the of the business problem here in the given problem we noticed that there is a problem of class imbalance as our class 1 is only 6.5 % in the dataset. So we didn't' depend on accuracy metric here and we get less ,precision and recall values for class 1 too. So here in this problem f1 score will play vital role so we will compare f1 values for each model with a balance precision & recall value and select our final model for deployment.

Comparison of the Performance Metrics of All the Models on Train Data

Sr. No.	Model Name (Train Data)	Precision Class 0 (Train Data)	Precision Class 1 (Train Data)	Recall Class 0 (Train Data)	Recall Class 1 (Train Data)	F1 Score Class 0 (Train Data)	F1 Score Class 01 (Train Data)	Model Accuracy (Train Data)	AUC / ROC (Train Data)
1	Logistic Regression using stats-model	96	63.2	93	41.1	97.2	49.8	94.6	-----
2	Logistic Regression using stats-model (Selected Features)	95.3	60.5	98.6	30.1	96.9	40.2	94.2	-----
3	Random Forest Base Model	100	93	99	95	100	94	99	99.9
4	Bagging of Random Forest Model	99	92	99	83	99	87	98	99.8
5	Logistic Regression using sk-learn	94	28	99	7	96	11	93	70.4
6	Gradient Boosting Model	99	90	99	85	99	87	98	99.1
7	Logistic Regression using sk-learn with SMOTE	67	64	61	70	64	67	65	70.6
8	Random Forest Model with SMOTE	100	99	99	100	100	100	100	100
9	Bagging of Random Forest Model with SMOTE	100	99	99	100	99	99	99	99
10	Gradient Boosting Model with SMOTE	99	97	97	99	98	98	98	99.7

Tab: 49 Comparison of the Performance Metrics of All the Models (Train Data)

Comparison of the Performance Metrics of All the Models on Test Data

Sr. No.	Model Name (Test Data)	Precision Class 0 (Test Data)	Precision Class 1 (Test Data)	Recall Class 0 (Test Data)	Recall Class 1 (Test Data)	F1 Score Class 0 (Test Data)	F1 Score Class 01 (Test Data)	Model Accuracy (Test Data)	AUC / ROC (Test Data)
1	Logistic Regression using stats-model	95.4	53.7	98.1	31.4	96.7	39.6	93.8	-----
2	Logistic Regression using stats-model (Selected Features)	95.5	52.3	97.9	32.9	96.7	40.4	93.7	-----
3	Random Forest Base Model	95	43	97	31	96	36	93	88.1
4	Bagging of Random Forest Model	95	48	98	31	96	38	93	90.2
5	Logistic Regression using sk-learn	94	35	99	11	96	17	93	73.8
6	Gradient Boosting Model	96	46	97	39	96	42	93	90.4
7	Logistic Regression using sk-learn with SMOTE	96	10	60	64	74	17	60	65.4
8	Random Forest Model with SMOTE	96	35	95	39	95	36	91	87.5
9	Bagging of Random Forest Model with SMOTE	96	36	94	50	95	42	91	89.4
10	Gradient Boosting Model with SMOTE	97	37	93	56	95	44	91	89.6

Tab: 50 Comparison of the Performance Metrics of All the Models (Test Data)

Conclusion of Comparison :

Study over the classification reports of all models and deployment of performance improvement techniques/methods and fine-tuning the model, we find that among the acceptable models are:

- **Logistic Regression using stats-model**
- **Logistic Regression using stats-model with (Selected Features)**

This is because the above models are balanced and show some degree of performance improvement when run on test data. Also, they do not have an overfitting problem on the Training Data and Test Data for class 1. However, we will choose the best model among the above.

The models that are being rejected due the problem of overfitting are:

- Random Forest Base Model
- Bagging of Random Forest Model
- Gradient Boosting Model
- Random Forest Model with SMOTE
- Bagging of Random Forest Model with SMOTE
- Gradient Boosting Model with SMOTE

Random Forest Base Model & Bagged Random Forest model and above listed models are suffer from the problem of overfitting and therefore these models are unreliable.

The models that are being rejected due the very poor performance are:

- Logistic Regression Model using sk-learn
- Logistic Regression Model using sk-learn with SMOTE

As we saw in the above comparison of performance metrics of various models ,we found these models performing poorly in terms of precision , recall and f1 score for class 1 on train and test data. Therefore these models are unreliable.

Final Model Selection :

As per the context of business problem and the given dataset we finalised Logistics Regression Model using stats-model with selected features model 2 is a good choice over Regression Model using stats-model 1 because its precision , recall and f1 score are okay as comparison with all other models. Plus we have better pseudo R-squared for model 2 as compared to model 1.So Logistics Regression Model using stats-model with selected features model 2 will predict better that which company will be defaulter or non-defaulter.

One more thing , as we found class proportion is unbalanced. We have only 6.5 % of class 1 and class 0 is 93.5 %, due to this problem model learns from training data but not predicting too much well on test data. We also try smote to make it balanced but didn't get the desired results. If we want to make this model better than we need more data in balanced form to increase its performance.

Insights from the Analysis

- The Company(FRA) .csv data set has 4256 observations (rows) and 50 variables (columns) in the dataset
- Out of 50 variables 49 are float64 and 1 variable is of int64 d-type. Memory used by
- the dataset: 1.6 MB.
- PE_on BSE (61.72%) ,Investments (40.30%) ,Other_income (36.56%) ,Contingent_liabilities (32.94%) and Deferred_tax_liability (32.17%) these are the variables with more than 30% null values.
- There is no Anomalies present in the dataset , but have nan values in most of columns.
- Number of duplicated rows in data set i.e. 665
- 93.5% of the data-points belongs to class 0.
- Only 6.5% of the data-points belongs to class 1.

- Total_assets - Total assets of customer ranges from a minimum of 0.100 to maximum of 1176509.
- Net_worth - Net worth of the customer of present year ranges from a minimum of 0.00 to maximum of 613151.60.
- Total_income - Total income of the customer ranges from a minimum of 0.00 to maximum of 2442828.
- Change_in_stock - difference between value of current stock and the value of stock in last trading day ranges from a minimum of -3029.400 to maximum of 14185.500.
- Total_expenses - Total expense done by customer ranges from a minimum of -0.100 to maximum of 2366035.
- Profit_after_tax - Profit after tax deduction ranges from a minimum of -3908.300 to maximum of 119439.100.
- PBDITA - Profit before depreciation, income tax and amortisation ranges from a minimum of -440.700 to maximum of 208576.500.
- PBT - Profit before tax deduction ranges from a minimum of -3894.800 to maximum of 145292.600.
- Cash_profit - Total Cash profit ranges from a minimum of -2245.700 to maximum of 176911.800.
- PBDITA_as_perc_of_total_income - PBDITA / Total income ranges from a minimum of -6400.00 to maximum of 100.00.
- PAT_as_perc_of_total_income - PAT / Total income ranges from a minimum of -21340.00 to maximum of 150.00.
- Cash_profit_as_perc_of_total_income - Cash Profit / Total income ranges from a minimum of -15020.00 to maximum of 100.00.
- PAT_as_perc_of_net_worth - PAT / Net worth ranges from a minimum of -748.720 to maximum of 2466.670.
- Sales - Sales done by customer ranges from a minimum of 0.100 to maximum of 2384984.
- Total_capital - Total capital of the customer ranges from a minimum of 0.100 to maximum of 78273.200.
- Reserves_and_funds - Total reserves and funds of the customer ranges from a minimum of -6525.900 to maximum of 625137.800.
- Borrowings - Total amount borrowed by customer ranges from a minimum of 0.100 to maximum of 278257.300.
- Current_liabilities_and_provisions - current liabilities of the customer ranges from a minimum of 0.100 to maximum of 352240.300.
- Shareholders_funds - Amount of equity in a company, which is belong to shareholder ranges from a minimum of 0.00 to maximum of 613151.600.
- Cumulative_retained_profits - Total cumulative profit retained by customer ranges from a minimum of -6534.300 to maximum of 390133.800.
- Capital_employed - Current asset minus current liabilities ranges from a minimum of 0.00 to maximum of 891408.900.
- TOL_to_TNW - Total liabilities of the customer divided by Total net worth ranges from a minimum of -350.480 to maximum of 473.00.
- Total_term_liabilities__to__tangible_net_worth - Short + long term liabilities divided by tangible net worth ranges from a minimum of -325.600 to maximum of 456.00.
- Contingent_liabilities__to__Net_worth_perc - Contingent liabilities / Net worth ranges from a minimum of 0.00 to maximum of 14704.270.
- Net_fixed_assets - purchase price of all fixed assets ranges from a minimum of 0.00 to maximum of 636604.600.
- Current_assets - Assets that are expected to be converted to cash within a year ranges from a minimum of 0.100 to maximum of 354815.200.
- Net_working_capital - Difference of current liabilities and current assets ranges from a minimum of -63839.00 to maximum of 85782.800.
- Quick_ratio_times - Total cash divided by current liabilities ranges from a minimum of 0.00 to maximum of 341.00.
- Current_ratio_times - Current assets divided by current liabilities ranges from a minimum of 0.00 to maximum of 505.00.
- Debt_to_equity_ratio_times - Total liabilities divided by its shareholder equity ranges from a minimum of 0.00 to maximum of 456.00.

- Cash_to_current_liabilities_times - Total liquid cash divided by current liabilities ranges from a minimum of 0.00 to maximum of 165.00.
- Cash_to_average_cost_of_sales_per_day - Total cash divided by average cost of the sales ranges from a minimum of 0.00 to maximum of 128040.760.
- Cash_to_average_cost_of_sales_per_day - Total cash divided by average cost of the sales ranges from a minimum of 0.00 to maximum of 128040.760.
- Creditors_turnover - Net credit purchase divided to average trade creditors ranges from a minimum of 0.00 to maximum of 2401.00.
- Debtors_turnover - Net credit sales divided by average accounts receivable ranges from a minimum of 0.00 to maximum of 3135.200.
- Finished_goods_turnover - Annual sales divided by average inventory ranges from a minimum of -0.090 to maximum of 17947.600.
- WIP_turnover - The cost of goods sold for a period divided by the average inventory for that period ranges from a minimum of -0.180 to maximum of 5651.400.
- Raw_material_turnover - Cost of goods sold is divided by the average inventory for the same period ranges from a minimum of -2.000 to maximum of 21092.000.
- Shares_outstanding - Number of issued shares minus the number of share held in the company ranges from a minimum of -2147484000 to maximum of 4130401000.
- Equity_face_value - cost of the equity at the time of issuing ranges from a minimum of -999998.900 to maximum of 100000.000.
- EPS - Net income divided by total number of outstanding share ranges from a minimum of -843181.820 to maximum of 34522.530.
- Adjusted_EPS - Adjusted net earning divided by the weighted average number of common share outstanding on a diluted basis during the plan year ranges from a minimum. of -843181.820 to maximum of 34522.530.
- Total_liabilities - Sum of all type of liabilities ranges from a minimum of 0.100 to maximum of 1176509.
- For the upper range (-1000 – 0) of PBDITA as % of total income, chances are high that the customer will default.
- For the upper range (-4000 – 0) of PBT as % of total income, chances are high that the
- customer will default.
- For the upper range (-5000 -0) of PAT as % of total income, chances are high that the
- customer will default.
- For the upper range (-3000 – 0) of Cash profit as % of total income, chances are high
- that the customer will default.
- For the lower range (-500 – 0) of PAT as % of net worth , chances are high that the
- customer will default.
- Customers in range 0-200 of TOL_to_TNW seem more likely to default.
- Customers in range 0-300 of Total_term_liabilities seem more likely to default.
- Customers in range 0-4000 of Contingent_liabilities seem more likely to default.
- Customers in range 0-250 of Debt_to_equity_ratio_times seem more likely to default.
- Customers in range 0-500 of creditors_turnover seems more likely to default.
- Customers in range 0-1000 of debtors_turnover seems more likely to default.
- Customers in range 0-2500 of Finished_goods_turnover seems more likely to default.
- **Perfect correlation is present between these variables: -**

1. Net_worth - Total_assets
2. Total_income - Total_assets, Net_worth
3. Total_expenses - Total_assets, Net_worth, Total_income
4. Profit_after_tax - Total_assets, Net_worth, Total_income, Total_expenses
5. Total_liabilities - Total_assets, Net_worth, Total_income, Total_expenses, Profit_after_tax, PBDITA, PBT, Cash_profit, Sales, Other_income, Borrowings, Current_liabilities_and_provisions, Deferred_tax_liability, Shareholders_funds, Cumulative_retained_profits, Capital_employed

- All the variables has outliers
- **Features which have VIF less than 5 are :**

1. Current_ratio_times
2. Cash_to_current_liabilities_times
3. Shares_outstanding
4. Cash_to_average_cost_of_sales_per_day

5. PAT_as_perc_of_net_worth
6. WIP_turnover
7. Income_from_fincial_services
8. Finished_goods_turnover
9. Net_working_capital
10. Debtors_turnover
11. Creditors_turnover
12. Raw_material_turnover
13. Change_in_stock
14. Contingent_liabilities__to__Net_worth_perc

These features help to build logistic regression model using stats-model , which gives results better than all other models for class 1 for precision , recall and f1.

- **Variable Importance** - PAT_as_perc_of_net_worth , Cash_profit , Shareholders_funds, TOL_to_TNW , Net_worth ,Debt_to_equity_ratio_times , cumulative_retained_profits , Reserves_and_funds , cash_profits_as_perc_of _total_income and Profit_after_ tax are the important variables for predicting the Defaulter and Non-defaulter.

Recommendations :

- As we saw class proportion is unbalanced. We have only 6.5 % of class 1 and class 0 is 93.5 %, due to this problem model learns from training data but not predicting too much well on test data. We also try smote to make it balanced but didn't get the desired results. If company want to increase the performance of this model than we need more data in balanced form. So we are suggesting company to increase the sample size i.e. dataset size.

- Secondly we suggest company to use variables which have high importance in predicting the target column.Instead of using or asking for all the features or attributes comapny should focus on PAT_as_perc_of_net_worth , Cash_profit , Shareholders_funds, TOL_to_TNW , Net_worth ,Debt_to_equity_ratio_times , cumulative_retained_profits , Reserves_and_funds , cash_profits_as_perc_of _total_income and Profit_after_ tax are the important variables for predicting the Defaulter and Non-defaulter.

- Before providing loan or credit to companies please check these ranges , as it will reduce the credit risk :

- For the upper range (-1000 – 0) of PBDITA as % of total income, chances are high that the customer will default.
- For the upper range (-4000 – 0) of PBT as % of total income, chances are high that the customer will default.
- For the upper range (-5000 -0) of PAT as % of total income, chances are high that the customer will default.
- For the upper range (-3000 – 0) of Cash profit as % of total income, chances are high that the customer will default.
- For the lower range (-500 – 0) of PAT as % of net worth , chances are high that the customer will default.
- Customers in range 0-200 of TOL_to_TNW seem more likely to default.
- Customers in range 0-500 of creditors_turnover seems more likely to default.
- Customers in range 0-1000 of debtors_turnover seems more likely to default.
- Do a proper check of customer's credit history and credit bureau report will also reduce credit risk.