PROJECT REPORT

Finance & Risk Analytics

PREEJA RAJESH
PGP - DSBA
2020 - 2021

Contents

Problem Statement 1: 3

Part 1 - Credit Risk 6

1.1 Outlier Treatment 6

1.2 Missing Value Treatment 14

1.3 Transform Target variable into 0 and 1 14

.... 14

1.4 Univariate analysis: 16

Bi-Variate Analysis: 21

1.5 Train Test Split 23

1.6 Build Logistic Regression Model on most important variables on Train Dataset 23

Model 1 – Logistic Regression 25

Model 2 - Logistic Regression (Statsmodel) 27

1.7 Validate the Model on Test Dataset and state the performance matrices 29

Model 3 - Logistic Regression (Sklearn) 34

1.8 Build a Random Forest Model on Train Dataset 35

Model 4 - Random Forest 35

1.9 Validate the Random Forest Model on test Dataset and state the performance matrices 35

1.10 Build an LDA Model on Train Dataset 36

Model 5 - Linear Discriminant Analysis (LDA) 36

1.11 Validate the LDA Model on test Dataset and state the performance matrices 36

1.12 Compare the performances of all the three models (include ROC Curve) 37

1.13 Recommendation: 39

Part 2 Market Risk: 40

2.1 Draw Stock Price Chart for any 2 variables 42

2.2 Calculate Returns 43

2.3 Calculate Stock Means and Standard Deviation 44

2.4 Draw a plot of Stock Means vs Standard Deviation and share insights 45

Problem Statement 1:

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

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

Data that is available includes information from the financial statement of the companies for the previous year (2015). Also, information about the Networth of the company in the following year (2016) is provided which can be used to drive the labeled field.

Data set for the Problem: Company_Data2015-2.xlsx

Importing the dataset

Data set:

	Co_Code	Co_Name	Networth Next Year	Equity Paid Up	Networth	Capital Employed	Total Debt	Gross Block	Net Working Capital
0	16974	Hind.Cables	-8021.60	419.36	-7027.48	-1007.24	5936.03	474.30	-1076.34
1	21214	Tata Tele. Mah.	-3986.19	1954.93	-2968.08	4458.20	7410.18	9070.86	-1098.88
2	14852	ABG Shipyard	-3192.58	53.84	506.86	7714.68	6944.54	1281.54	4496.25
3	2439	GTL	-3054.51	157.30	-623.49	2353.88	2326.05	1033.69	-2612.42
4	23505	Bharati Defence	-2967.36	50.30	-1070.83	4675.33	5740.90	1084.20	1836.23

Exploratory Data Analysis:

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

Data types of all variables

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 3586 entries, 0 to 3585
Data columns (total 67 columns):
 #   Column                         Non-Null Count  Dtype  
---  ------                         --------------  -----  
 0   Co_Code                        3586 non-null   int64  
 1   Co_Name                        3586 non-null   object 
 2   Networth_Next_Year             3586 non-null   float64
 3   Equity_Paid_Up                 3586 non-null   float64
 4   Networth                       3586 non-null   float64
 5   Capital_Employed               3586 non-null   float64
 6   Total_Debt                     3586 non-null   float64
 7   Gross_Block                    3586 non-null   float64
 8   Net_Working_Capital            3586 non-null   float64
 9   Curr_Assets                    3586 non-null   float64
 10  Curr_Liab_and_Prov             3586 non-null   float64
 11  Total_Assets_to_Liab           3586 non-null   float64
 12  Gross_Sales                    3586 non-null   float64
 13  Net_Sales                      3586 non-null   float64
 14  Other_Income                   3586 non-null   float64
 15  Value_Of_Output                3586 non-null   float64
 16  Cost_of_Prod                   3586 non-null   float64
 17  Selling_Cost                   3586 non-null   float64
 18  PBIDT                          3586 non-null   float64
 19  PBDT                           3586 non-null   float64
 20  PBIT                           3586 non-null   float64
 21  PBT                            3586 non-null   float64
 22  PAT                            3586 non-null   float64
 23  Adjusted_PAT                   3586 non-null   float64
 24  CP                             3586 non-null   float64
 25  Rev_earn_in_forex              3586 non-null   float64
 26  Rev_exp_in_forex               3586 non-null   float64
 27  Capital_exp_in_forex           3586 non-null   float64
 28  Book_Value_Unit_Curr           3586 non-null   float64
 29  Book_Value_Adj_Unit_Curr       3582 non-null   float64
 30  Market_Capitalisation          3586 non-null   float64
 31  CEPS_annualised_Unit_Curr      3586 non-null   float64
 32  Cash_Flow_From_Oper            3586 non-null   float64
 33  Cash_Flow_From_Inv             3586 non-null   float64
 34  Cash_Flow_From_Fin             3586 non-null   float64
 35  ROG_Net_Worth_perc             3586 non-null   float64
 36  ROG_Capital_Employed_perc      3586 non-null   float64
 37  ROG_Gross_Block_perc           3586 non-null   float64
 38  ROG_Gross_Sales_perc           3586 non-null   float64
 39  ROG_Net_Sales_perc             3586 non-null   float64
 40  ROG_Cost_of_Prod_perc          3586 non-null   float64
 41  ROG_Total_Assets_perc          3586 non-null   float64
 42  ROG_PBIDT_perc                 3586 non-null   float64
 43  ROG_PBDT_perc                  3586 non-null   float64
 44  ROG_PBIT_perc                  3586 non-null   float64
 45  ROG_PBT_perc                   3586 non-null   float64
 46  ROG_PAT_perc                   3586 non-null   float64
 47  ROG_CP_perc                    3586 non-null   float64
```

```
 48   ROG_Rev_earn_in_forex_perc         3586 non-null   float64
 49   ROG_Rev_exp_in_forex_perc          3586 non-null   float64
 50   ROG_Market_Capitalisation_perc     3586 non-null   float64
 51   Curr_Ratio_Latest                  3585 non-null   float64
 52   Fixed_Assets_Ratio_Latest          3585 non-null   float64
 53   Inventory_Ratio_Latest             3585 non-null   float64
 54   Debtors_Ratio_Latest               3585 non-null   float64
 55   Total_Asset_Turnover_Ratio_Latest  3585 non-null   float64
 56   Interest_Cover_Ratio_Latest        3585 non-null   float64
 57   PBIDTM_perc_Latest                 3585 non-null   float64
 58   PBITM_perc_Latest                  3585 non-null   float64
 59   PBDTM_perc_Latest                  3585 non-null   float64
 60   CPM_perc_Latest                    3585 non-null   float64
 61   APATM_perc_Latest                  3585 non-null   float64
 62   Debtors_Vel_Days                   3586 non-null   int64
 63   Creditors_Vel_Days                 3586 non-null   int64
 64   Inventory_Vel_Days                 3483 non-null   float64
 65   Value_of_Output_to_Total_Assets    3586 non-null   float64
 66   Value_of_Output_to_Gross_Block     3586 non-null   float64
dtypes: float64(63), int64(3), object(1)
memory usage: 1.8+ MB
```

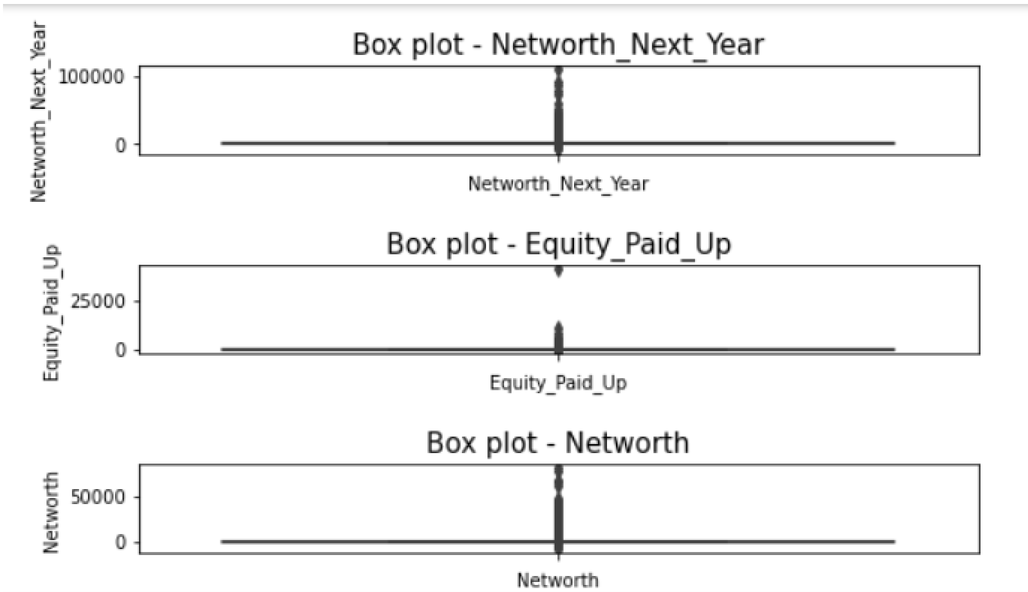
5 Point summary:

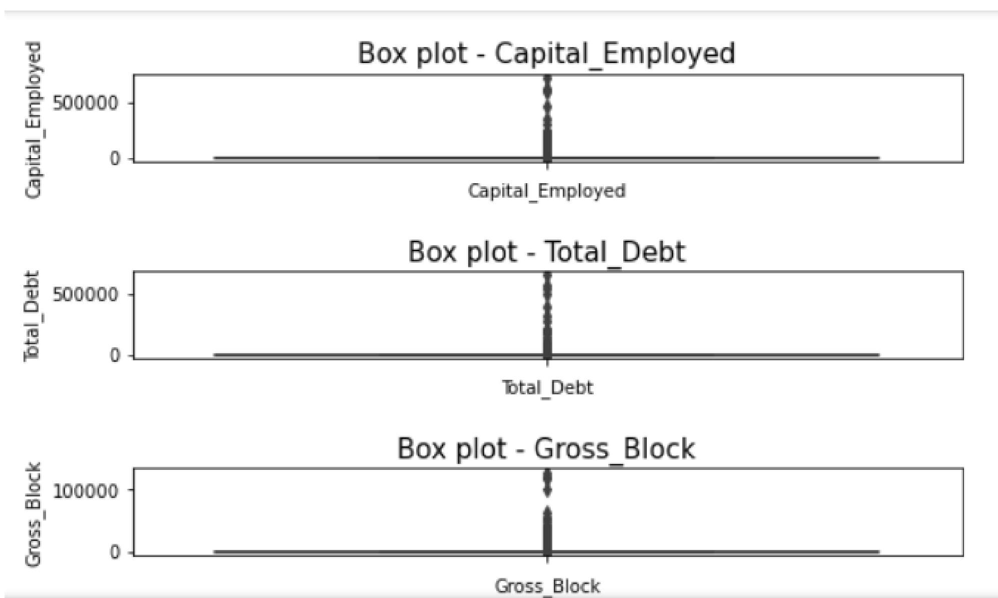
	Co_Code	Networth_Next_Year	Equity_Paid_Up	Networth	Capital_Employed
count	3586.000000	3586.000000	3586.000000	3586.000000	3586.000000
mean	16065.388734	725.045251	62.966584	649.746299	2799.611054
std	19776.817379	4769.681004	778.761744	4091.988792	26975.135385
min	4.000000	-8021.600000	0.000000	-7027.480000	-1824.750000
25%	3029.250000	3.985000	3.750000	3.892500	7.602500
50%	6077.500000	19.015000	8.290000	18.580000	39.090000
75%	24269.500000	123.802500	19.517500	117.297500	226.605000
max	72493.000000	111729.100000	42263.460000	81657.350000	714001.250000

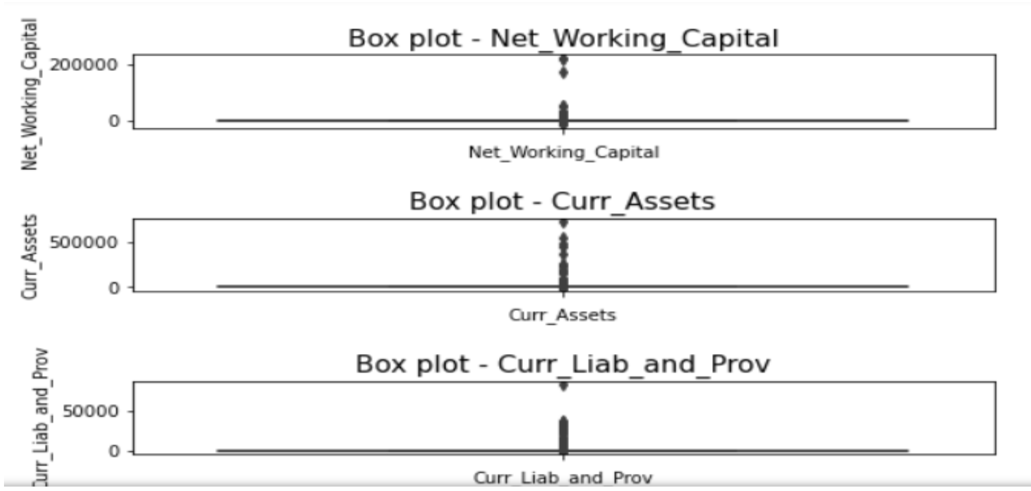
Part 1 - Credit Risk

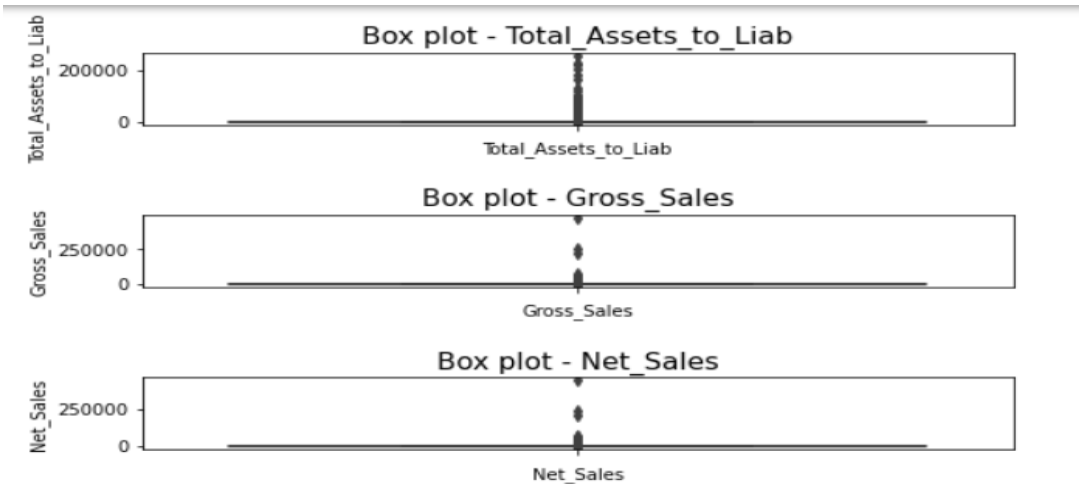
1.1 Outlier Treatment

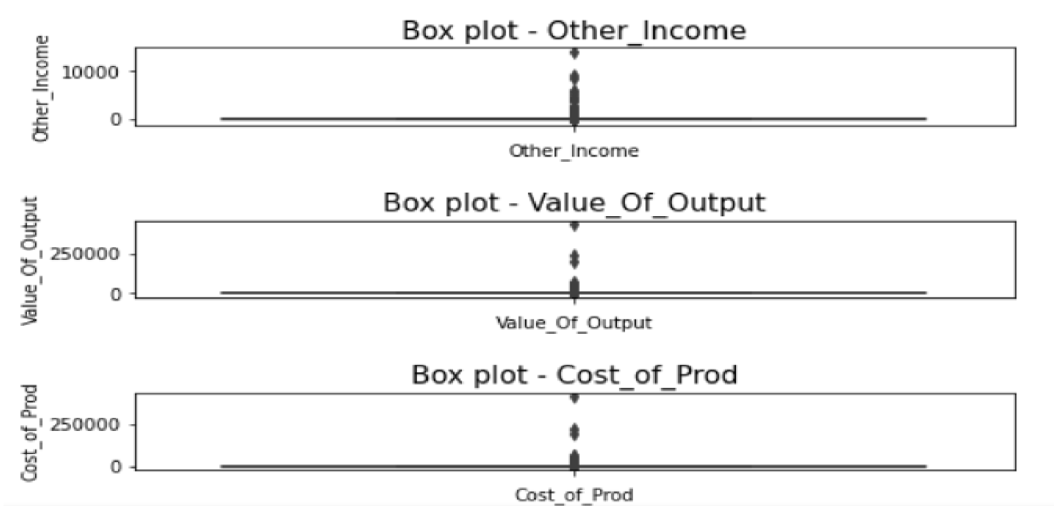
Creating outlier identification (Lower & Upper whiskers) function

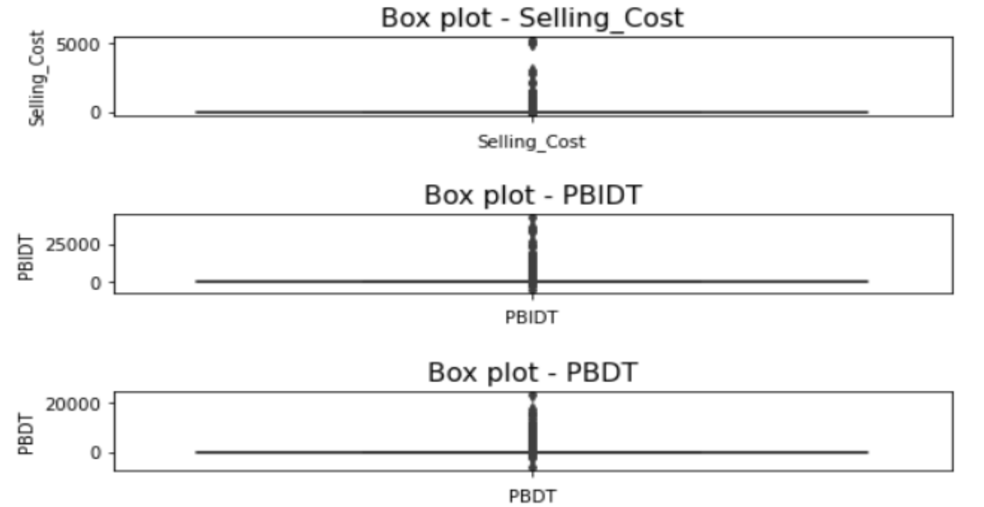


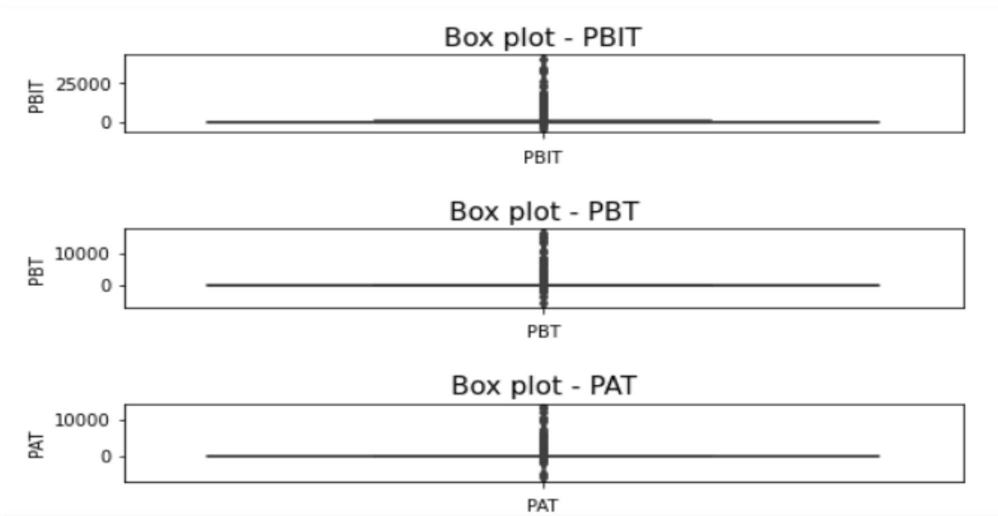


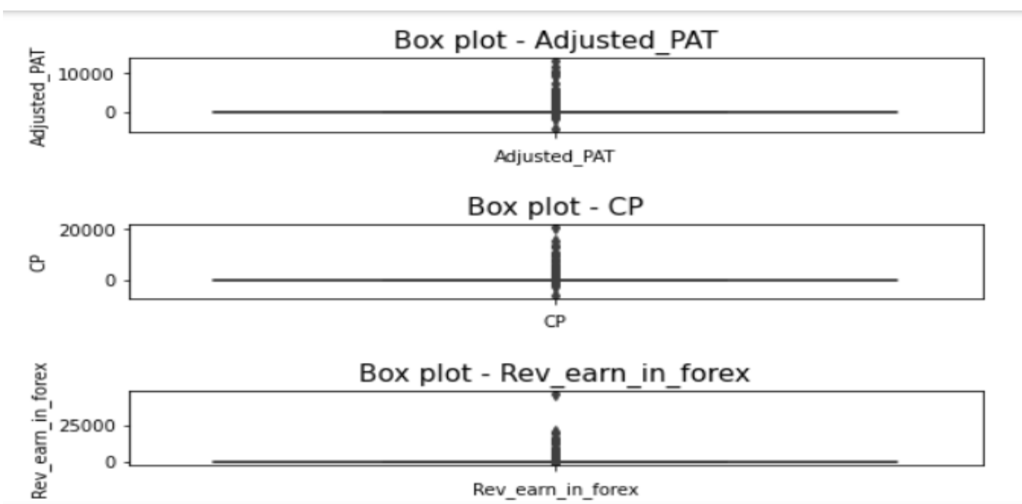

Box plot - Net_Working_Capital
Net_Working_Capital
200000
0
Net_Working_Capital
Box plot - Curr_Assets
Curr_Assets
500000
0
Curr_Assets
Box plot - Curr_Liab_and_Prov
Curr_Liab_and_Prov
50000
0
Curr_Liab_and_Prov

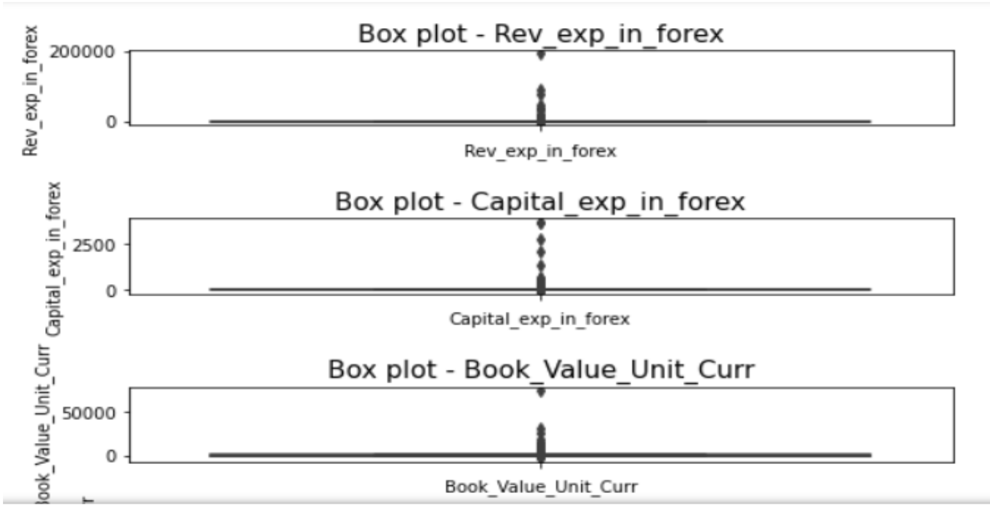

Box plot - Total_Assets_to_Liab
Total_Assets_to_Liab
200000
0
Total_Assets_to_Liab
Box plot - Gross_Sales
Gross_Sales
250000
0
Gross_Sales
Box plot - Net_Sales
Net_Sales
250000
0
Net_Sales

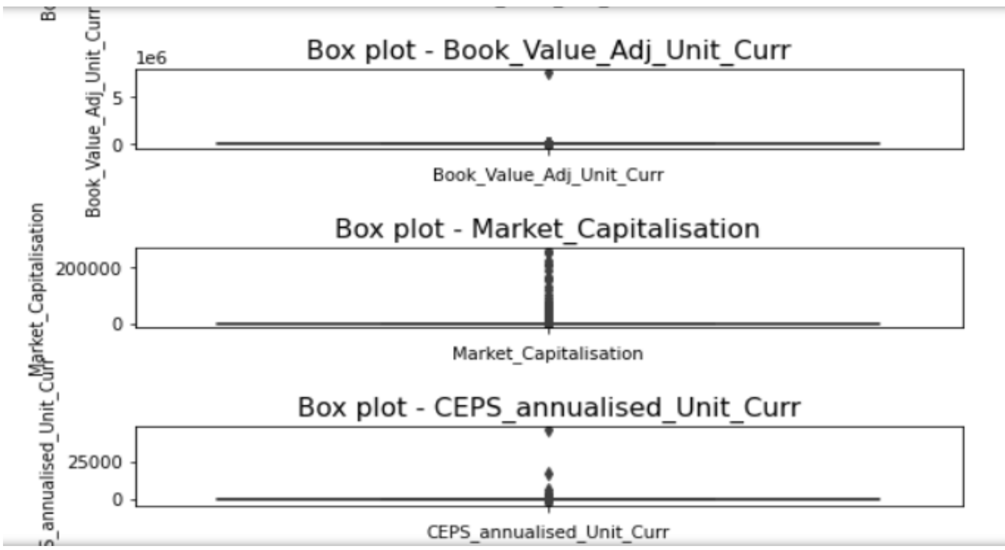

Box plot - Other_Income
Other_Income
10000
0
Other_Income
Box plot - Value_Of_Output
Value_Of_Output
250000
0
Value_Of_Output
Box plot - Cost_of_Prod
Cost_of_Prod
250000
0
Cost_of_Prod

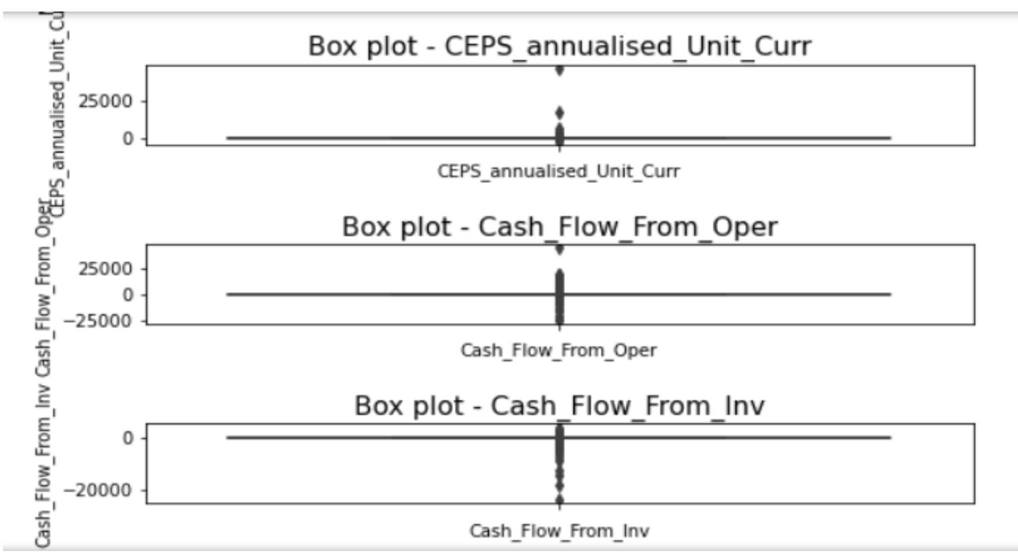

Box plot - Selling_Cost
Selling_Cost
5000
0
Selling_Cost
Box plot - PBIDT
PBIDT
25000
0
PBIDT
Box plot - PBDT
PBDT
20000
0
PBDT

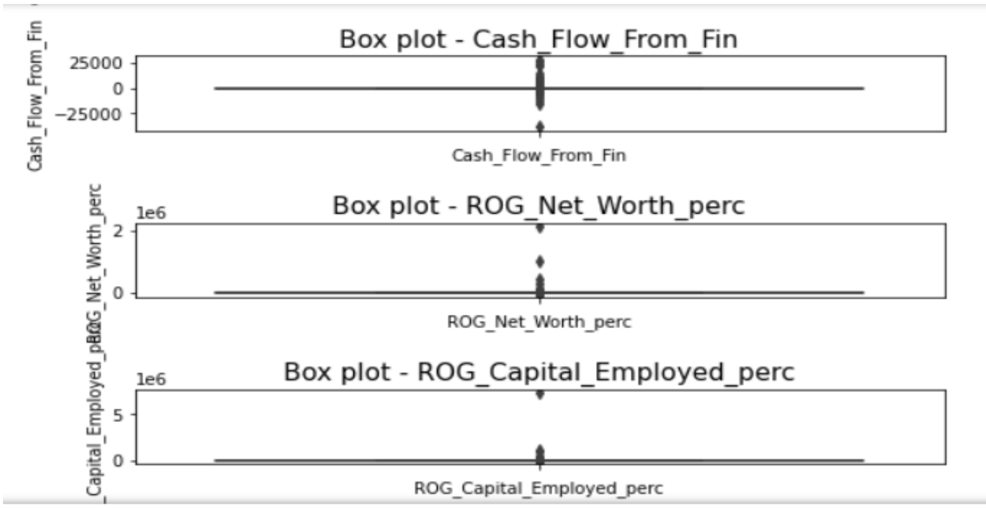

Box plot - PBIT
PBIT
25000
0
PBIT
Box plot - PBT
PBT
10000
0
PBT
Box plot - PAT
PAT
10000
0
PAT

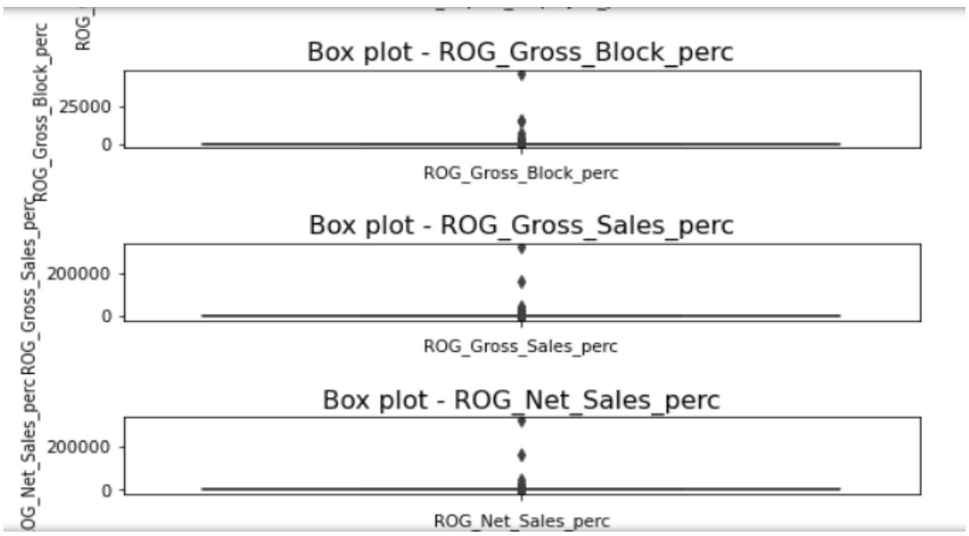

Box plot - Adjusted_PAT
Adjusted_PAT
10000
0
Adjusted_PAT
Box plot - CP
CP
20000
0
CP
Box plot - Rev_earn_in_forex
Rev_earn_in_forex
25000
0
Rev_earn_in_forex

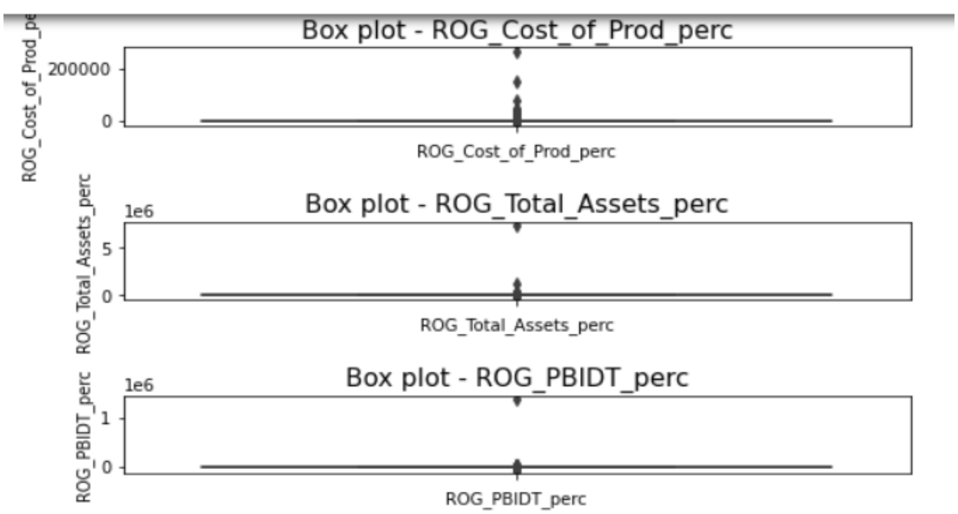

Box plot - Rev_exp_in_forex
Rev_exp_in_forex
200000
0
Rev_exp_in_forex
Box plot - Capital_exp_in_forex
Capital_exp_in_forex
2500
0
Capital_exp_in_forex
Box plot - Book_Value_Unit_Curr
Book_Value_Unit_Curr
50000
0
Book_Value_Unit_Curr

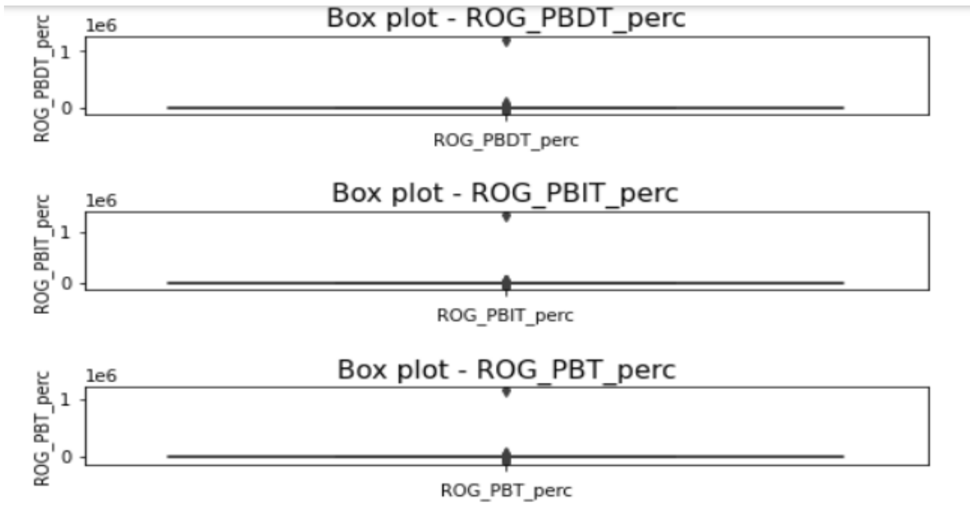

Box plot - Book_Value_Adj_Unit_Curr
Book_Value_Adj_Unit_Curr
1e6
5
0
Book_Value_Adj_Unit_Curr
Box plot - Market_Capitalisation
Market_Capitalisation
200000
0
Market_Capitalisation
Box plot - CEPS_annualised_Unit_Curr
25000
0
CEPS_annualised_Unit_Curr

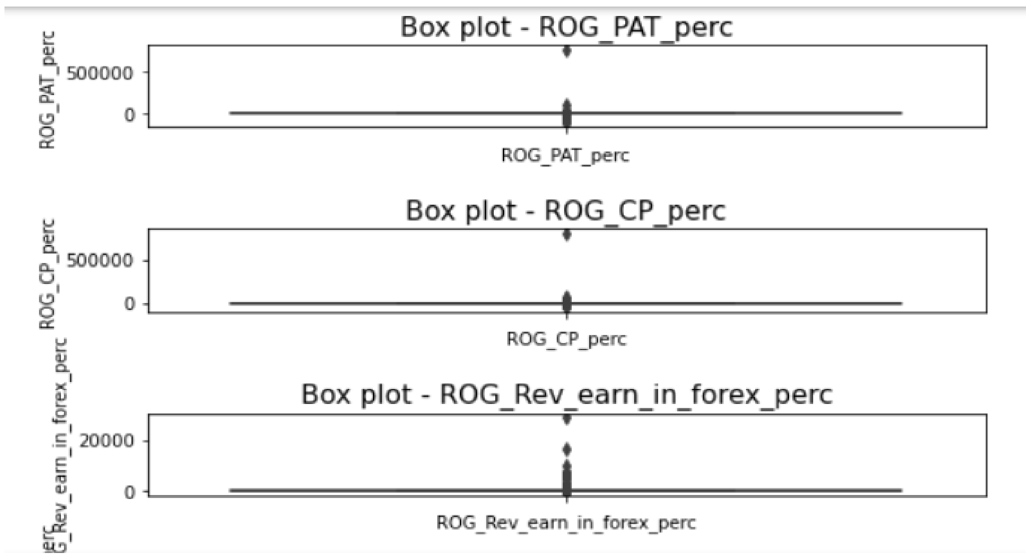

Box plot - CEPS_annualised_Unit_Curr
CEPS_annualised_Unit_Curr
25000
0
CEPS_annualised_Unit_Curr
Box plot - Cash_Flow_From_Oper
Cash_Flow_From_Oper
25000
0
−25000
Cash_Flow_From_Oper
Box plot - Cash_Flow_From_Inv
Cash_Flow_From_Inv
0
−20000
Cash_Flow_From_Inv

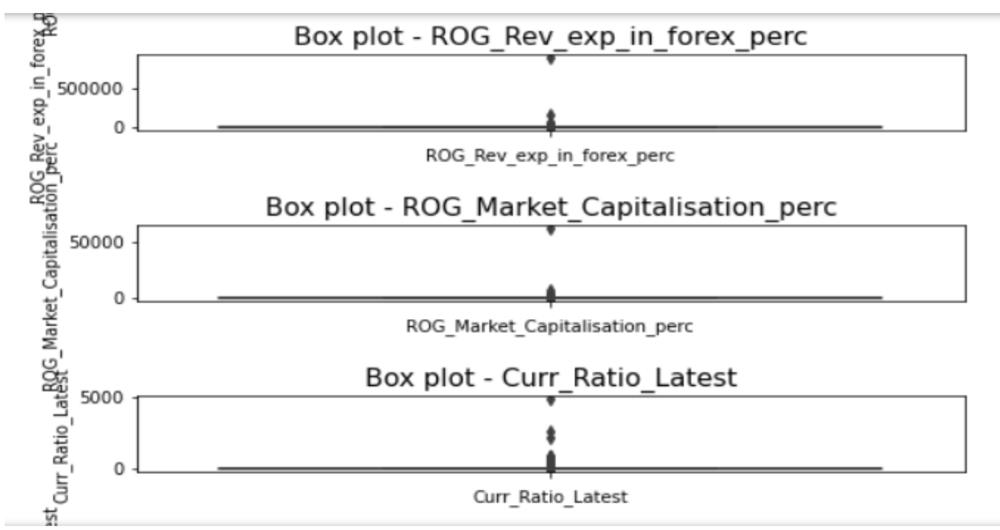

Box plot - Cash_Flow_From_Fin
25000
0
−25000
Cash_Flow_From_Fin
Box plot - ROG_Net_Worth_perc
1e6
2
0
ROG_Net_Worth_perc
Box plot - ROG_Capital_Employed_perc
1e6
5
0
ROG_Capital_Employed_perc

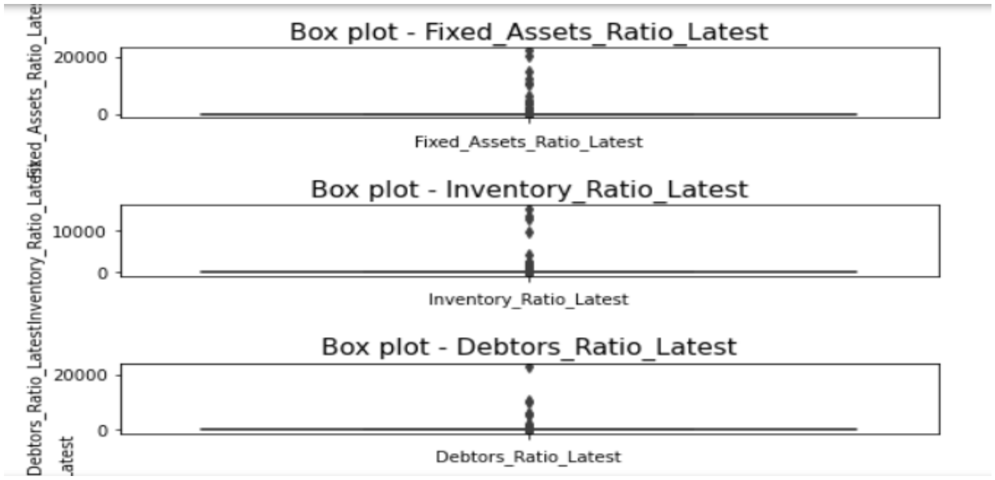

Box plot - ROG_Gross_Block_perc
25000
0
ROG_Gross_Block_perc
Box plot - ROG_Gross_Sales_perc
200000
0
ROG_Gross_Sales_perc
Box plot - ROG_Net_Sales_perc
200000
0
ROG_Net_Sales_perc

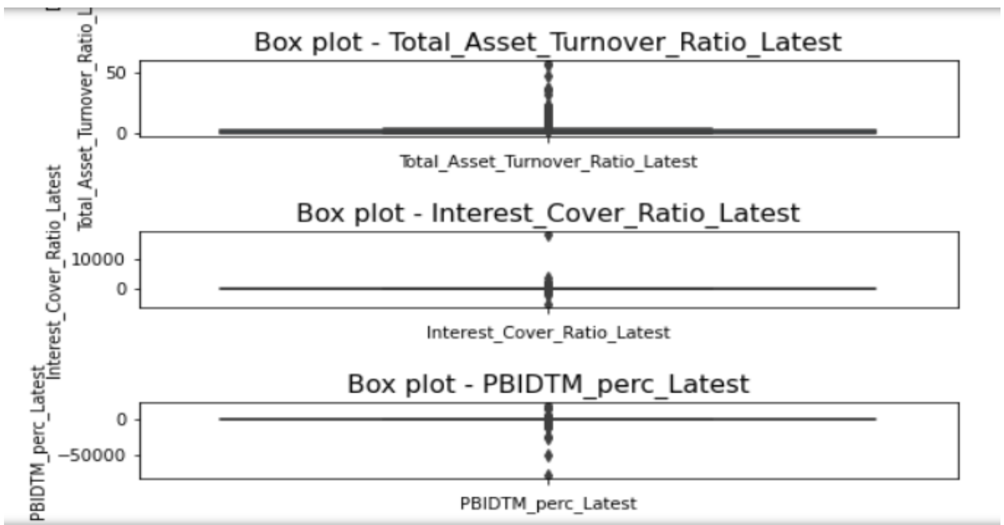

Box plot - ROG_Cost_of_Prod_perc
200000
0
ROG_Cost_of_Prod_perc
Box plot - ROG_Total_Assets_perc
1e6
5
0
ROG_Total_Assets_perc
Box plot - ROG_PBIDT_perc
1e6
1
0
ROG_PBIDT_perc

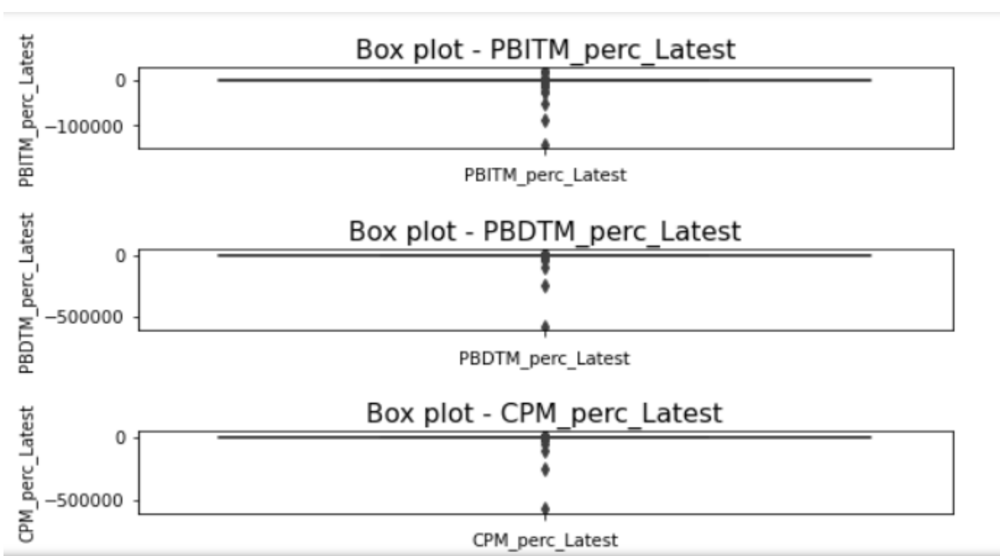

Box plot - ROG_PBDT_perc
ROG_PBDT_perc
1e6
1
0
ROG_PBDT_perc
Box plot - ROG_PBIT_perc
ROG_PBIT_perc
1e6
1
0
ROG_PBIT_perc
Box plot - ROG_PBT_perc
ROG_PBT_perc
1e6
1
0
ROG_PBT_perc

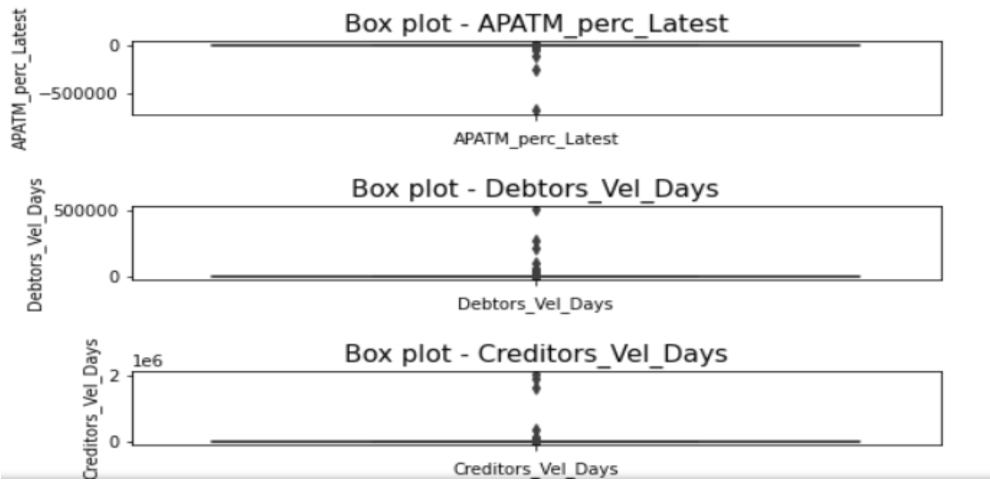

Box plot - ROG_PAT_perc
ROG_PAT_perc
500000
0
ROG_PAT_perc
Box plot - ROG_CP_perc
ROG_CP_perc
500000
0
ROG_CP_perc
Box plot - ROG_Rev_earn_in_forex_perc
20000
0
ROG_Rev_earn_in_forex_perc

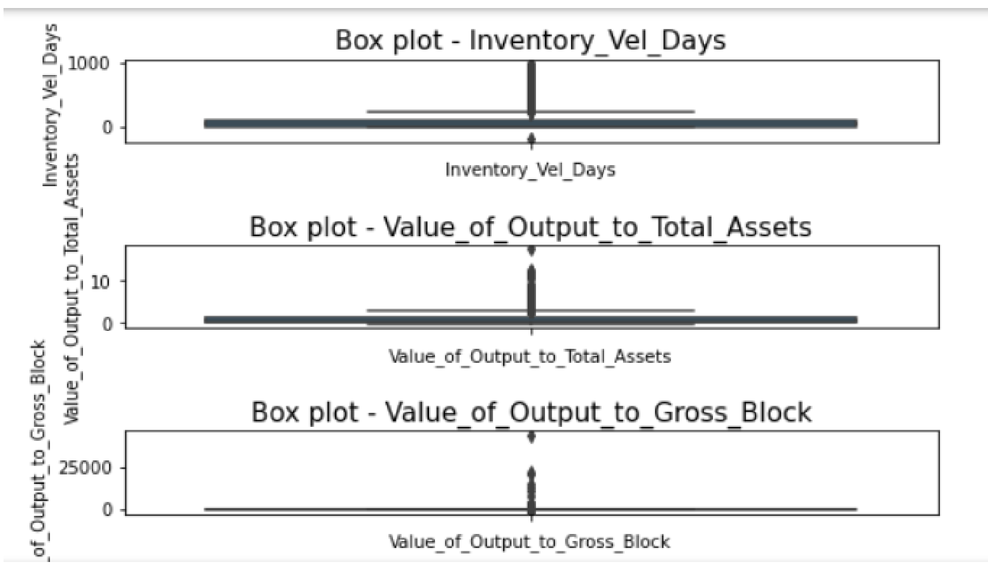

Box plot - ROG_Rev_exp_in_forex_perc
500000
0
ROG_Rev_exp_in_forex_perc
Box plot - ROG_Market_Capitalisation_perc
ROG_Market_Capitalisation_perc
50000
0
ROG_Market_Capitalisation_perc
Box plot - Curr_Ratio_Latest
Curr_Ratio_Latest
5000
0
Curr_Ratio_Latest


Box plot - Fixed_Assets_Ratio_Latest
20000
0
Fixed_Assets_Ratio_Latest
Box plot - Inventory_Ratio_Latest
10000
0
Inventory_Ratio_Latest
Box plot - Debtors_Ratio_Latest
20000
0
Debtors_Ratio_Latest


Box plot - Total_Asset_Turnover_Ratio_Latest
50
0
Total_Asset_Turnover_Ratio_Latest
Box plot - Interest_Cover_Ratio_Latest
10000
0
Interest_Cover_Ratio_Latest
Box plot - PBIDTM_perc_Latest
0
−50000
PBIDTM_perc_Latest


Box plot - PBITM_perc_Latest
0
−100000
PBITM_perc_Latest
Box plot - PBDTM_perc_Latest
0
−500000
PBDTM_perc_Latest
Box plot - CPM_perc_Latest
0
−500000
CPM_perc_Latest





There are outliers in the dataset, let’s use capping method to treat them

Let’s check outliers (Lower and Upper whiskers) in these variables

- **`Networth_Next_Year`** - `(-175.74125, 303.52875000000006)`
- **`Gross_Sales`** - `(-359.76875, 603.4612500000001)`
- **`Net_Sales`** - `(-348.06000000000006, 583.94)`
- **`PBT`** - `(-11.28375, 18.64625)`

1.2 Missing Value Treatment

There are missing values in the following variables of the dataset:

```
1. Book_Value_Adj_Unit_Curr              4
2. Curr_Ratio_Latest                     1
3. Fixed_Assets_Ratio_Latest             1
4. Inventory_Ratio_Latest                1
5. Debtors_Ratio_Latest                  1
6. Total_Asset_Turnover_Ratio_Latest     1
7. Interest_Cover_Ratio_Latest           1
8. PBIDTM_perc_Latest                    1
9. PBITM_perc_Latest                     1
10.  PBDTM_perc_Latest                    1
11.  CPM_perc_Latest                      1
12.  APATM_perc_Latest                    1
13.  Inventory_Vel_Days                 103
```

Let's treat these missing values with median (replacement with median elimin ates impact of outliers in the treatment)

1.3 Transform Target variable into 0 and 1

	default	Networth_Next_Year
0	1	-17.445
1	1	-17.445
2	1	-17.445
3	1	-17.445
4	1	-17.445
5	1	-17.445
6	1	-17.445
7	1	-17.445
8	1	-17.445
9	1	-17.445

	default	Networth_Next_Year
3576	0	1978.8225
3577	0	1978.8225
3578	0	1978.8225
3579	0	1978.8225
3580	0	1978.8225
3581	0	1978.8225
3582	0	1978.8225
3583	0	1978.8225
3584	0	1978.8225
3585	0	1978.8225

```
0     3198
1      388
```

- **Checking the proportion of default:**

```
388/(3198+388) = 0.10819854991634133
```

10.8% Companies in the total dataset are prone to default

- **Checking summary statistics of default variable**

```
count    3586.000000
mean        0.108199
std         0.310674
min         0.000000
25%         0.000000
50%         0.000000
75%         0.000000
max         1.000000
Name: default, dtype: float64
```

Average default rate matches with overall default rate of 10.8%

- **Let us check significance of variables 'PBT' in predicting Networth_Next_Year(default) before proceeding to model development.**

- **Checking Descriptive statistics of the variable 'PBT'**

```
count    3586.000000
mean       -8.115020
std       197.883559
min      -513.947500
25%       -41.235000
50%         0.025000
75%        61.957500
max       372.377500
Name: ROG_PBT_perc, dtype: float64
```

- **Checking Descriptive statistics of the variable 'PBT' for non-defaulters.**

```
count    3198.000000
mean       -1.161241
std       193.633271
min      -513.947500
25%       -37.457500
50%         2.175000
75%        63.140000
max       372.377500
Name: ROG_PBT_perc, dtype: float64
```

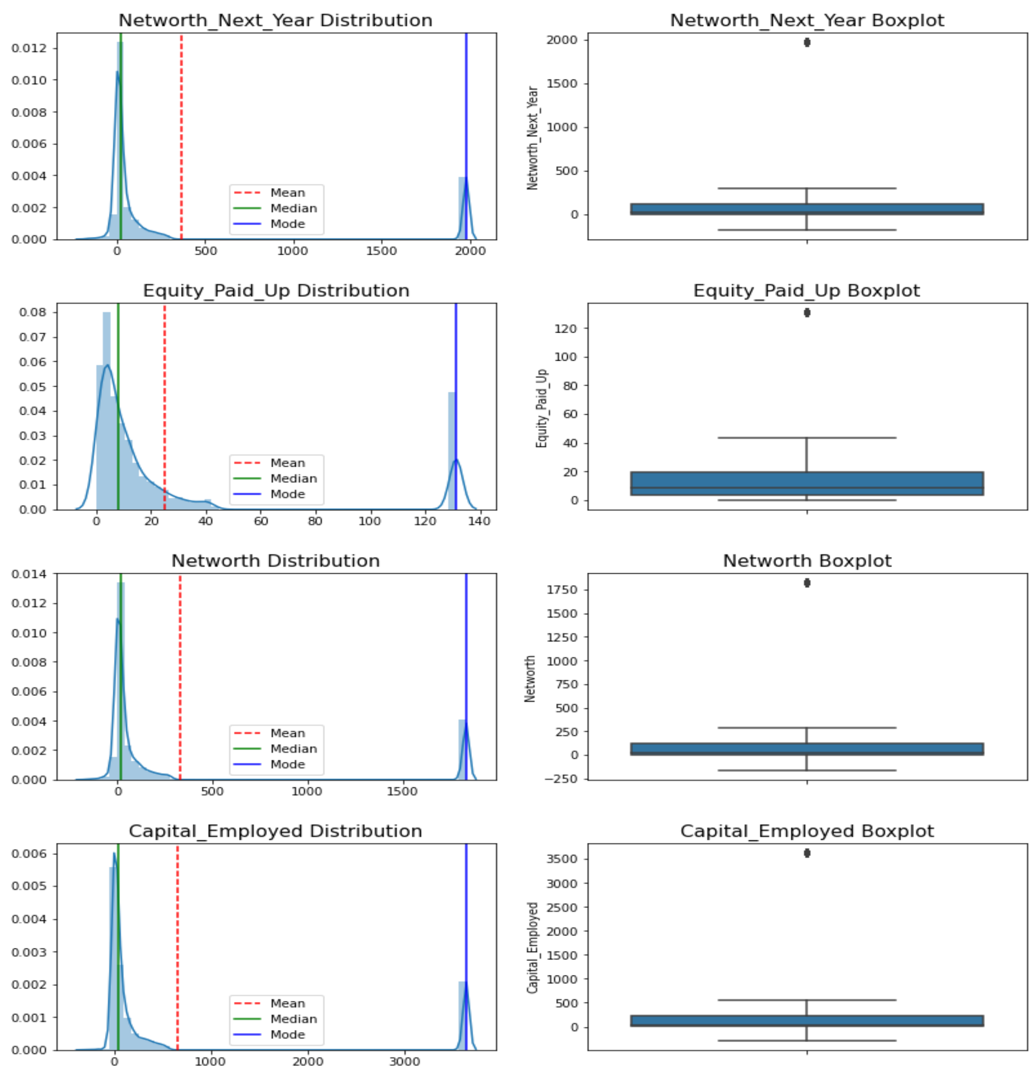
For companies whose have not defaulted, median 'Profit before tax is about 2.1'

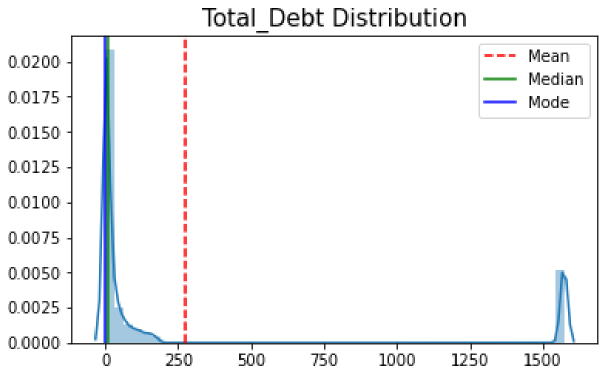
- **Checking Descriptive statistics of the variable 'PBT' for defaulters.**

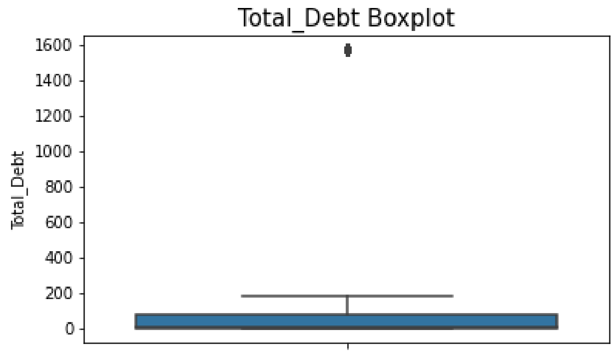
```
count     388.000000
mean      -65.429936
std       222.064802
min      -513.947500
25%      -100.055000
50%         0.000000
75%        50.000000
max       372.377500
Name: ROG_PBT_perc, dtype: float64
```

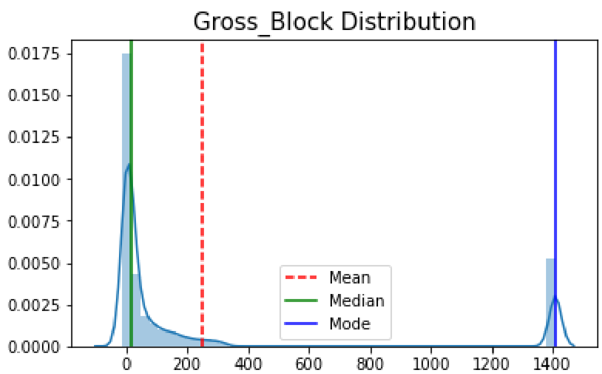
For companies whose have defaulted, median 'Profit before tax is about 0'
In conclusion what it means is, typical good companies make a profit of about 2.1 un its per 100 units of income And a typical defaulted companies loses about 0 units per 100 units of income

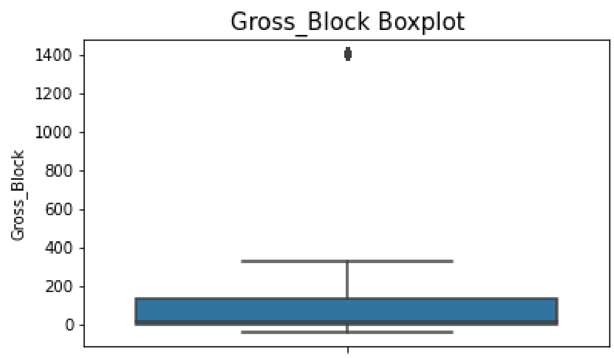
1.4 Univariate analysis:

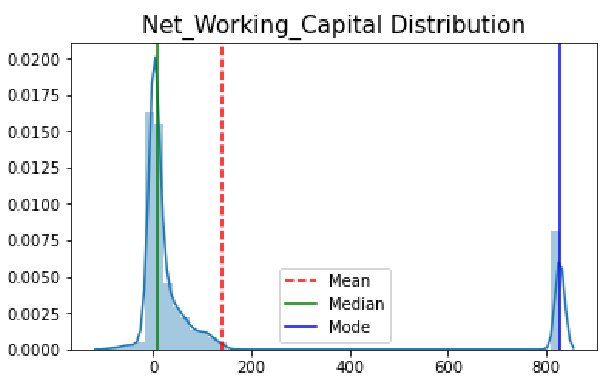


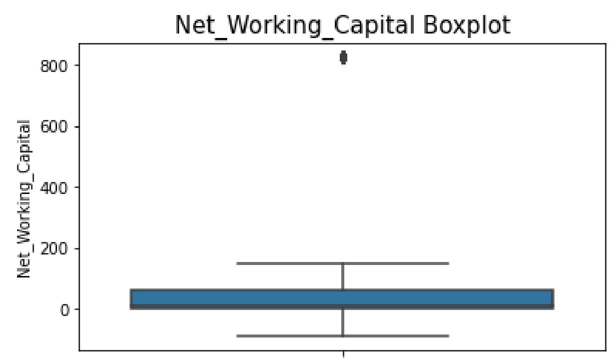

Total_Debt Distribution
Mean
Median
Mode
0.0000
0.0025
0.0050
0.0075
0.0100
0.0125
0.0150
0.0175
0.0200
0
250
500
750
1000
1250
1500

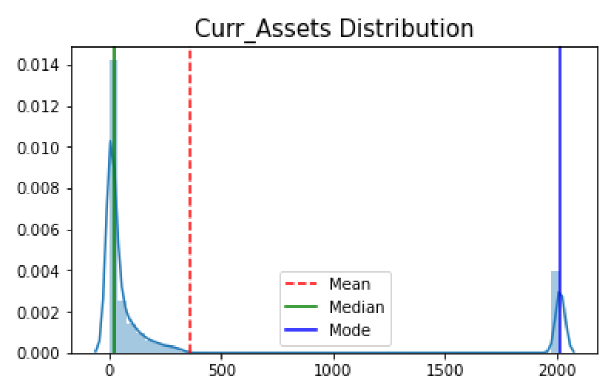

Total_Debt Boxplot
Total_Debt
0
200
400
600
800
1000
1200
1400
1600

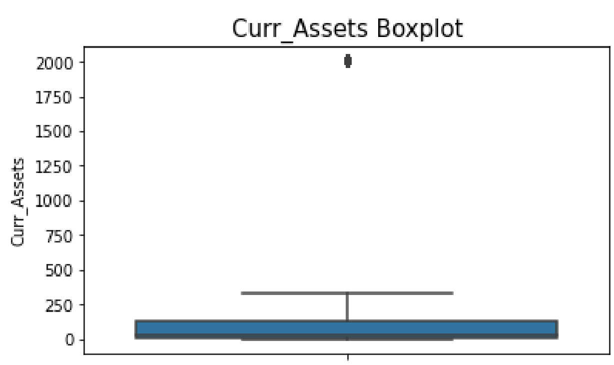

Gross_Block Distribution
Mean
Median
Mode
0.0000
0.0025
0.0050
0.0075
0.0100
0.0125
0.0150
0.0175
0
200
400
600
800
1000
1200
1400

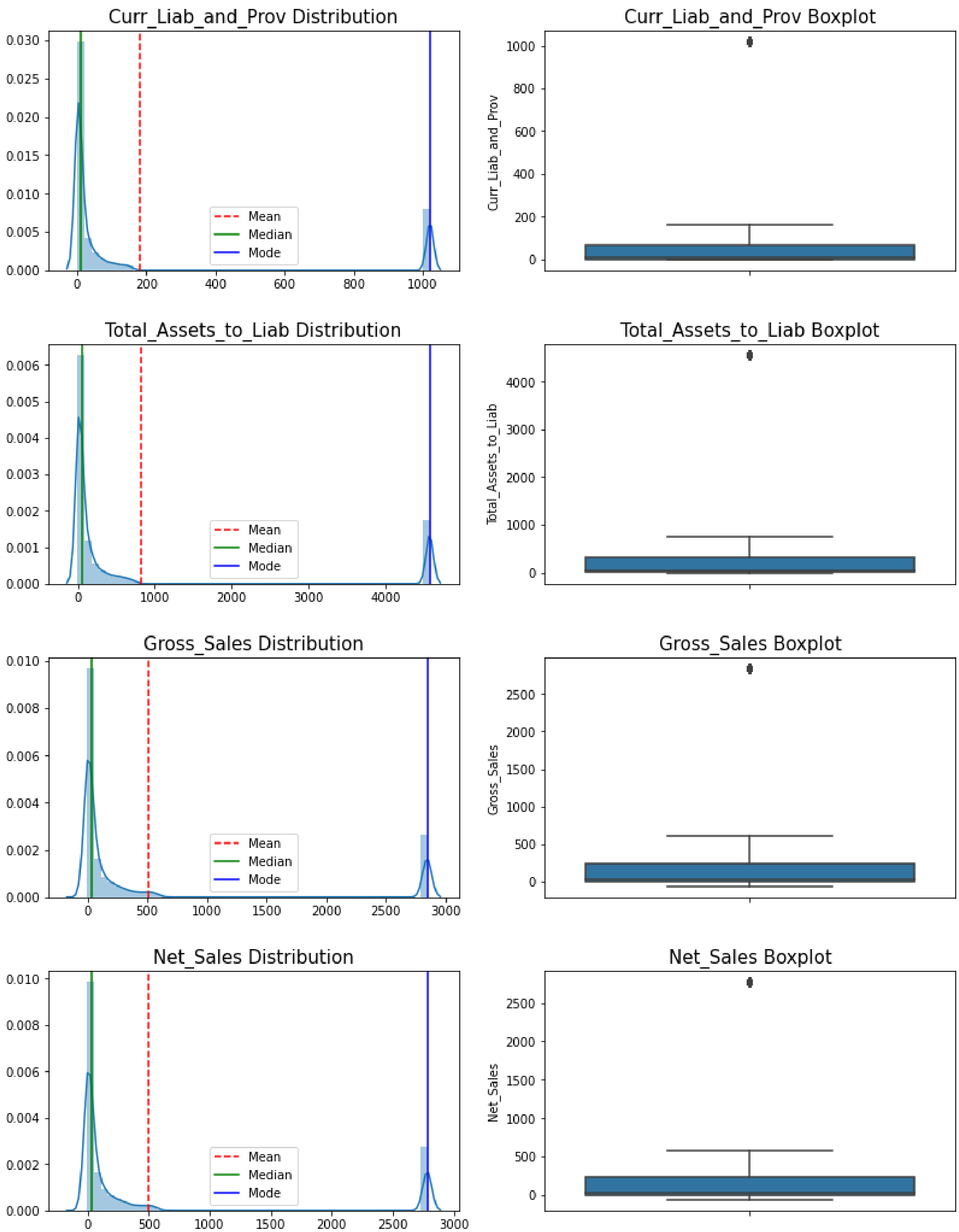

Gross_Block Boxplot
Gross_Block
0
200
400
600
800
1000
1200
1400


Net_Working_Capital Distribution
Mean
Median
Mode
0.0000
0.0025
0.0050
0.0075
0.0100
0.0125
0.0150
0.0175
0.0200
0
200
400
600
800

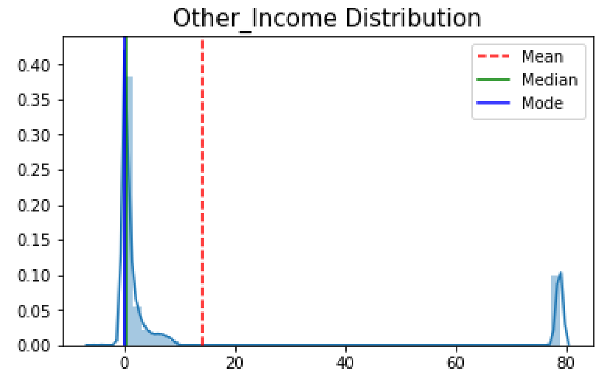

Net_Working_Capital Boxplot
Net_Working_Capital
0
200
400
600
800

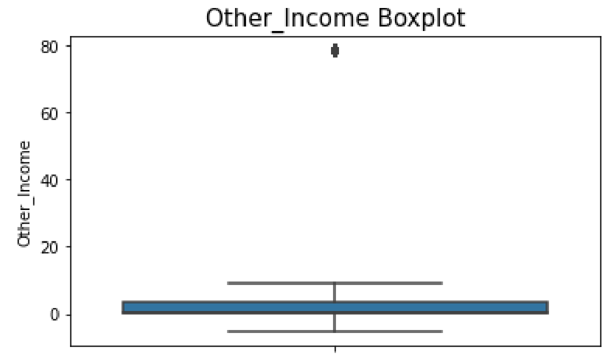

Curr_Assets Distribution
Mean
Median
Mode
0.000
0.002
0.004
0.006
0.008
0.010
0.012
0.014
0
500
1000
1500
2000

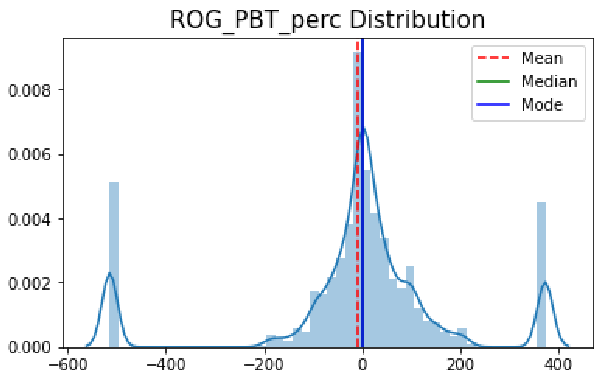

Curr_Assets Boxplot
Curr_Assets
0
250
500
750
1000
1250
1500
1750
2000

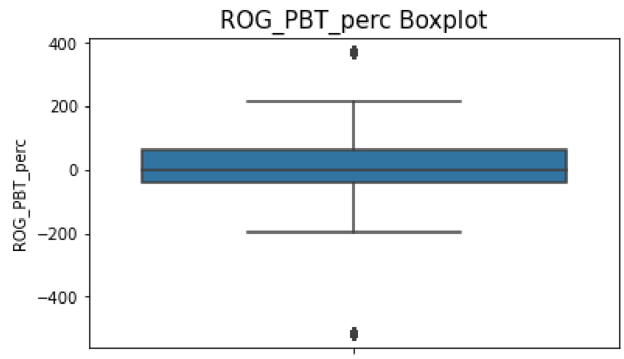


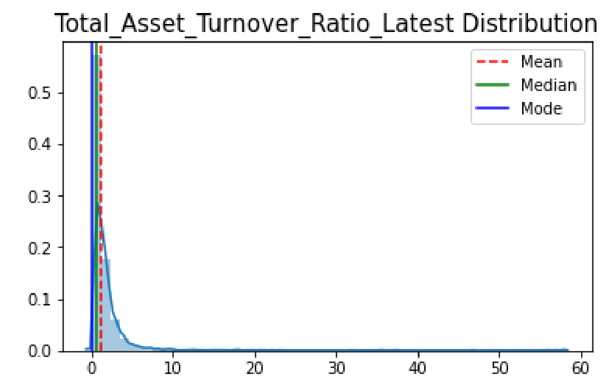
Curr_Liab_and_Prov Distribution
Mean
Median
Mode
Curr_Liab_and_Prov Boxplot
Curr_Liab_and_Prov
Total_Assets_to_Liab Distribution
Mean
Median
Mode
Total_Assets_to_Liab Boxplot
Total_Assets_to_Liab
Gross_Sales Distribution
Mean
Median
Mode
Gross_Sales Boxplot
Gross_Sales
Net_Sales Distribution
Mean
Median
Mode
Net_Sales Boxplot
Net_Sales

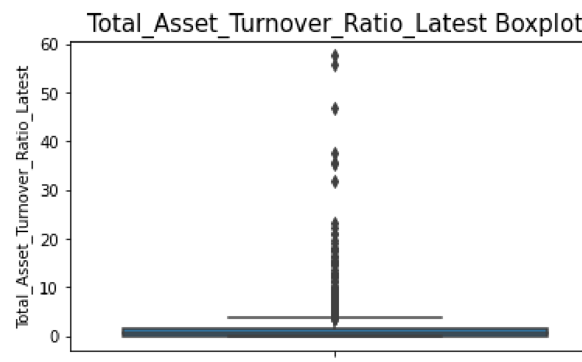

Other_Income Distribution
Mean
Median
Mode

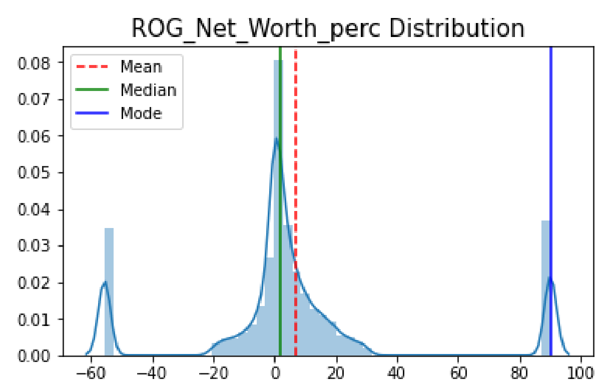

Other_Income Boxplot
Other_Income

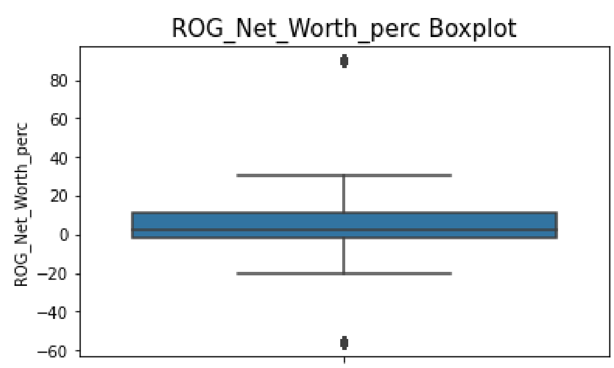

ROG_PBT_perc Distribution
Mean
Median
Mode


ROG_PBT_perc Boxplot
ROG_PBT_perc


Total_Asset_Turnover_Ratio_Latest Distribution
Mean
Median
Mode


Total_Asset_Turnover_Ratio_Latest Boxplot
Total_Asset_Turnover_Ratio_Latest


ROG_Net_Worth_perc Distribution
Mean
Median
Mode

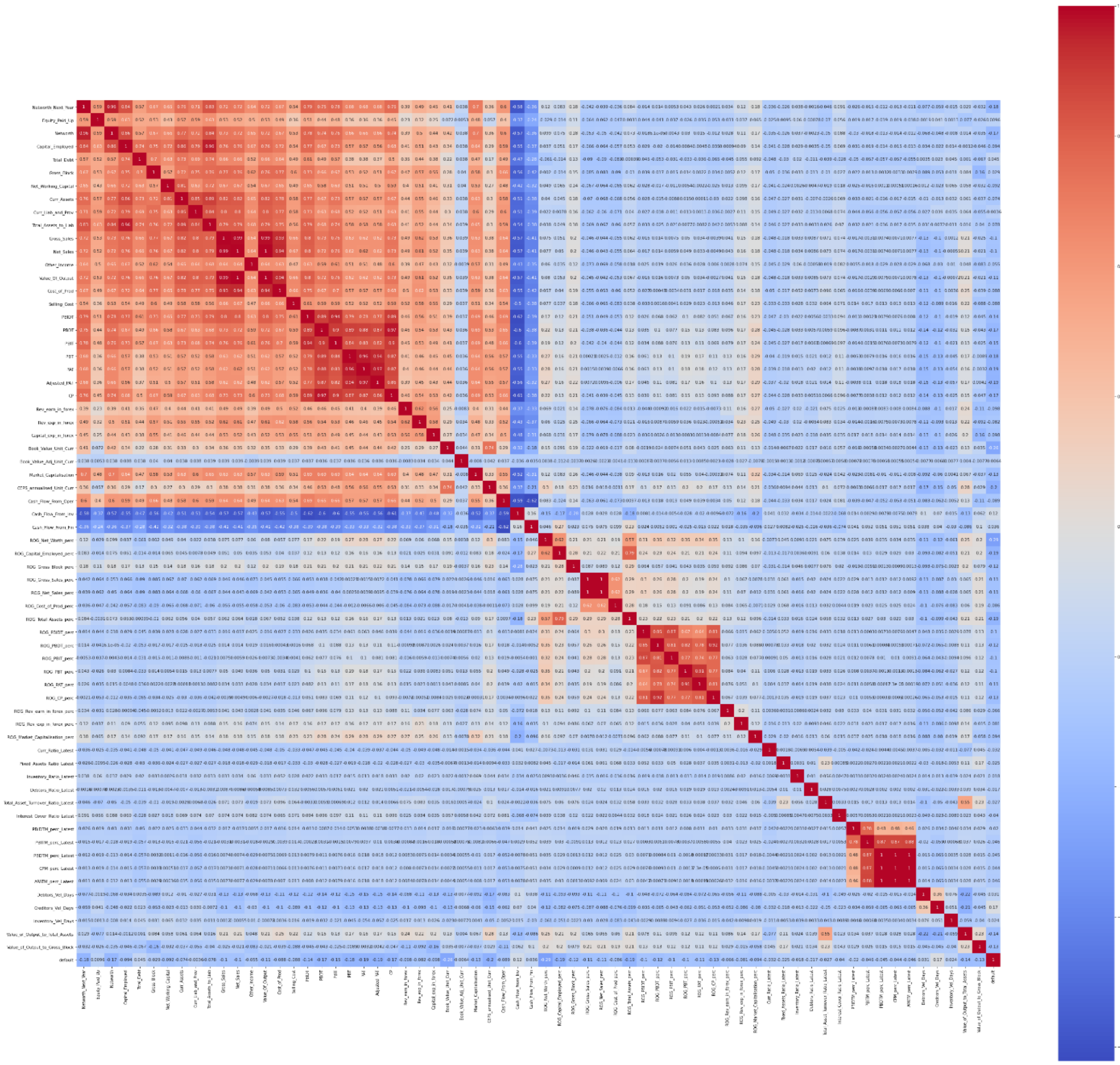

ROG_Net_Worth_perc Boxplot
ROG_Net_Worth_perc

To measure the skeweness of every attribute:

```
Networth_Next_Year                      1.751175
Equity_Paid_Up                          2.086800
Networth                                1.799108
Capital_Employed                        1.783536
Total_Debt                              1.815459
Gross_Block                             1.888628
Net_Working_Capital                     1.924737
Curr_Assets                             1.818667
Curr_Liab_and_Prov                      1.808656
Total_Assets_to_Liab                    1.825442
Gross_Sales                             1.858550
Net_Sales                               1.855065
Other_Income                            1.764381
Value_Of_Output                         1.848036
Cost_of_Prod                            1.831873
Selling_Cost                            1.753899
PBIDT                                   1.754702
PBDT                                    1.668444
PBIT                                    1.759498
PBT                                     1.604391
PAT                                     1.573827
Adjusted_PAT                            1.590750
CP                                      1.658778
Rev_earn_in_forex                       1.454781
Rev_exp_in_forex                        1.552273
Capital_exp_in_forex                    1.552139
Book_Value_Unit_Curr                    1.872894
Book_Value_Adj_Unit_Curr               59.877217
Market_Capitalisation                   1.678750
CEPS_annualised_Unit_Curr               1.834748
Cash_Flow_From_Oper                     1.708939
Cash_Flow_From_Inv                     -1.657783
Cash_Flow_From_Fin                     -1.305180
ROG_Net_Worth_perc                      0.865557
ROG_Capital_Employed_perc               1.341156
ROG_Gross_Block_perc                    0.818541
ROG_Gross_Sales_perc                    1.848110
ROG_Net_Sales_perc                      1.852963
ROG_Cost_of_Prod_perc                   1.818736
ROG_Total_Assets_perc                   1.343172
ROG_PBIDT_perc                          1.089836
ROG_PBDT_perc                           0.155558
ROG_PBIT_perc                           0.810952
ROG_PBT_perc                           -0.910237
ROG_PAT_perc                           -0.535738
ROG_CP_perc                             0.076314
ROG_Rev_earn_in_forex_perc             -0.100233
ROG_Rev_exp_in_forex_perc               0.532183
ROG_Market_Capitalisation_perc          1.657618
Curr_Ratio_Latest                      31.255716
Fixed_Assets_Ratio_Latest              24.126480
Inventory_Ratio_Latest                 27.006606
Debtors_Ratio_Latest                   35.261410
```

```
Total_Asset_Turnover_Ratio_Latest     10.360259
Interest_Cover_Ratio_Latest           40.829653
PBIDTM_perc_Latest                   -30.935896
PBITM_perc_Latest                    -36.002893
PBDTM_perc_Latest                    -47.756985
CPM_perc_Latest                      -47.018189
APATM_perc_Latest                    -49.284357
Debtors_Vel_Days                       2.216580
Creditors_Vel_Days                     2.253606
Inventory_Vel_Days                     3.556921
Value_of_Output_to_Total_Assets        0.982197
Value_of_Output_to_Gross_Block         2.094863
default                                2.523672
dtype: float64
```

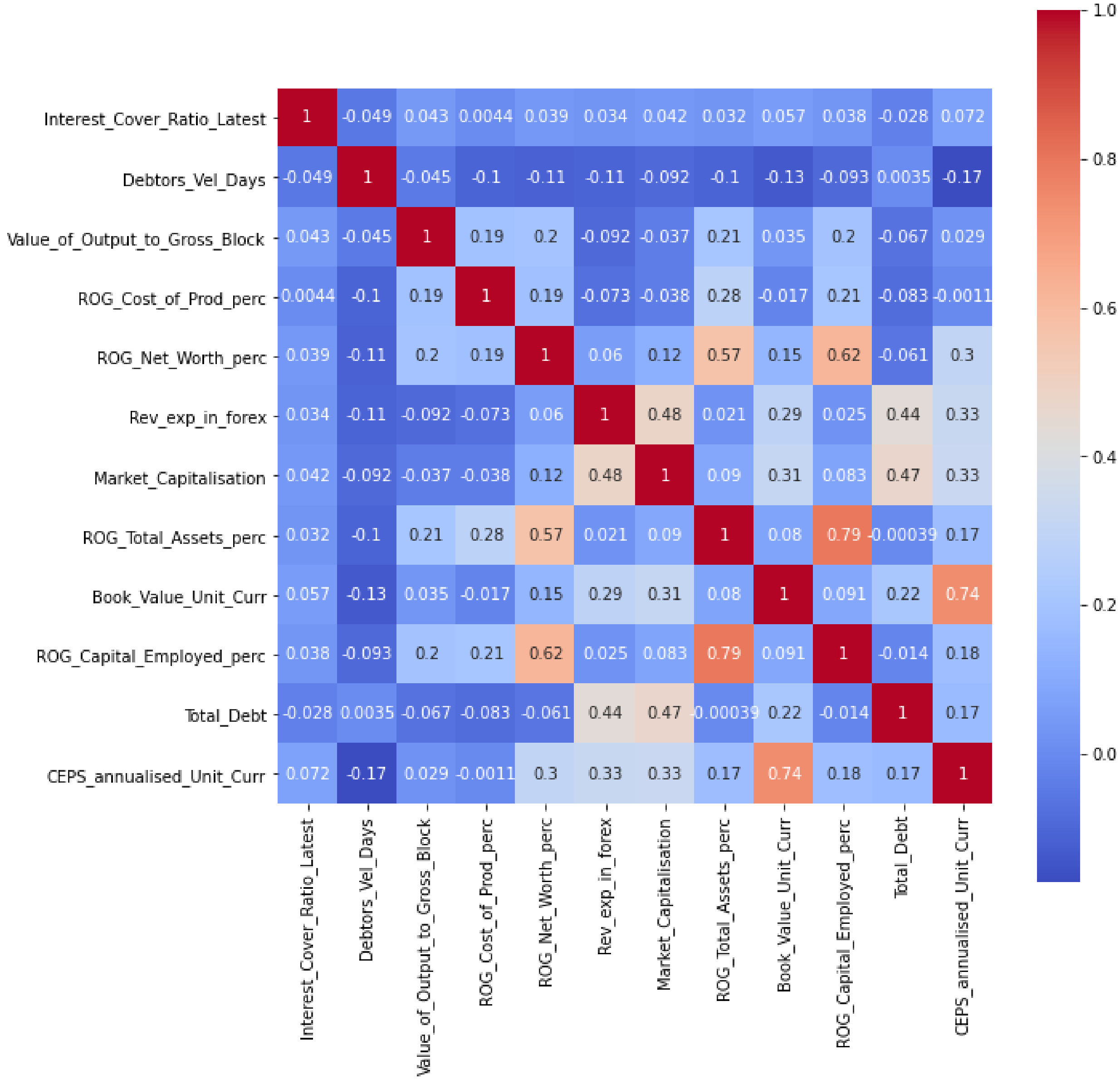
Bi-Variate Analysis:



Inference:

- First few variables (approx. 29 variables) are highly correlated to each other which shows in red - orange color.

Selected variables without multicollinearity



1.5 Train Test Split

Splitting arrays or matrices into random train and test subsets. Model will be fitted on train set and predictions will be made on the test set

```
X = Company.drop(['default','Networth_Next_Year'], axis=1)
y = Company['default']
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test =
train_test_split(X,y,test_size=0.33,random_state=42,stratify=Company['default'])
Company_train = pd.concat([X_train,y_train], axis=1)
Company_test = pd.concat([X_test,y_test], axis=1)
Company_train.to_csv('Company_train.csv',index=False)
Company_test.to_csv('Company_test.csv',index=False)
```

1.6 Build Logistic Regression Model on most important variables on Train Dataset

Before starting model building, let's look at the problem of multicollinearity. Multicollinearity occurs when two or more independent variables are highly correlated with one another in a regression model.

	variables	VIF
51	Debtors_Ratio_Latest	1.021243e+00
26	Book_Value_Adj_Unit_Curr	1.022753e+00
48	Curr_Ratio_Latest	1.023580e+00
50	Inventory_Ratio_Latest	1.039941e+00
53	Interest_Cover_Ratio_Latest	1.079519e+00
49	Fixed_Assets_Ratio_Latest	1.091491e+00
45	ROG_Rev_earn_in_forex_perc	1.118596e+00
46	ROG_Rev_exp_in_forex_perc	1.194357e+00
34	ROG_Gross_Block_perc	1.278542e+00
61	Inventory_Vel_Days	1.284497e+00
47	ROG_Market_Capitalisation_perc	1.507138e+00
60	Creditors_Vel_Days	1.547699e+00
59	Debtors_Vel_Days	1.561576e+00
63	Value_of_Output_to_Gross_Block	1.623351e+00
52	Total_Asset_Turnover_Ratio_Latest	1.745511e+00
37	ROG_Cost_of_Prod_perc	1.905652e+00
32	ROG_Net_Worth_perc	2.151541e+00
22	Rev_earn_in_forex	2.418991e+00
31	Cash_Flow_From_Fin	2.425077e+00

```
0                      Equity_Paid_Up  2.488868e+00
24                Capital_exp_in_forex  2.650147e+00
23                    Rev_exp_in_forex  2.855170e+00
11                        Other_Income  2.876768e+00
14                        Selling_Cost  3.003806e+00
30                  Cash_Flow_From_Inv  3.004314e+00
62     Value_of_Output_to_Total_Assets  3.037694e+00
27               Market_Capitalisation  3.181710e+00
38               ROG_Total_Assets_perc  3.214704e+00
25                Book_Value_Unit_Curr  3.441361e+00
33          ROG_Capital_Employed_perc  3.465365e+00
3                           Total_Debt  3.869653e+00
28          CEPS_annualised_Unit_Curr  4.021658e+00
5                  Net_Working_Capital  4.454152e+00
29                 Cash_Flow_From_Oper  4.477459e+00
4                          Gross_Block  5.229333e+00
41                       ROG_PBIT_perc  6.533250e+00
7                   Curr_Liab_and_Prov  6.570884e+00
39                      ROG_PBIDT_perc  7.503174e+00
1                             Networth  7.526605e+00
43                        ROG_PAT_perc  7.631506e+00
42                        ROG_PBT_perc  8.752629e+00
44                         ROG_CP_perc  9.122133e+00
6                          Curr_Assets  1.102961e+01
40                       ROG_PBDT_perc  1.150587e+01
13                        Cost_of_Prod  1.286867e+01
17                                PBIT  1.376020e+01
15                               PBIDT  1.498045e+01
2                     Capital_Employed  1.731066e+01
18                                 PBT  1.856289e+01
8                Total_Assets_to_Liab  2.134722e+01
20                        Adjusted_PAT  2.278784e+01
21                                  CP  2.448345e+01
16                                PBDT  2.595878e+01
19                                 PAT  3.390117e+01
9                          Gross_Sales  5.752165e+01
36                  ROG_Net_Sales_perc  1.199063e+02
35                ROG_Gross_Sales_perc  1.205352e+02
12                     Value_Of_Output  1.527914e+02
10                           Net_Sales  2.046160e+02
56                  PBDTM_perc_Latest  2.940952e+02
54                 PBIDTM_perc_Latest  3.300852e+10
55                  PBITM_perc_Latest  4.105939e+10
57                    CPM_perc_Latest  5.172449e+10
58                  APATM_perc_Latest  6.449423e+10
```

Here, we see that the value of VIF is high for many variables. Here, we may drop variables with VIF more than 5 (very high correlation) & build our model

Model 1 – Logistic Regression

Logit Regression Results

Dep. Variable:	default	**No. Observations:**	2402
Model:	Logit	**Df Residuals:**	2367
Method:	MLE	**Df Model:**	34
Date:	Sun, 20 Dec 2020	**Pseudo R-squ.:**	0.6113
Time:	09:47:45	**Log-Likelihood:**	-320.07
converged:	True	**LL-Null:**	-823.47
Covariance Type:	nonrobust	**LLR p-value:**	2.004e-189

	coef	**std err**	**z**	**P>|z|**	**[0.025**	**0.975]**
Intercept	-1.0642	0.190	-5.597	0.000	-1.437	-0.692
Debtors_Ratio_Latest	-0.0015	0.003	-0.525	0.600	-0.007	0.004
Book_Value_Adj_Unit_Curr	-3.811e-05	0.000	-0.164	0.870	-0.000	0.000
Curr_Ratio_Latest	-0.0038	0.005	-0.724	0.469	-0.014	0.006
Inventory_Ratio_Latest	-0.0028	0.002	-1.267	0.205	-0.007	0.002
Interest_Cover_Ratio_Latest	-0.0019	0.001	-2.019	0.044	-0.004	-5.4e-05
Fixed_Assets_Ratio_Latest	-0.0009	0.001	-0.639	0.523	-0.003	0.002
ROG_Rev_earn_in_forex_perc	-0.0012	0.004	-0.266	0.790	-0.010	0.008
ROG_Rev_exp_in_forex_perc	-0.0027	0.002	-1.145	0.252	-0.007	0.002
ROG_Gross_Block_perc	-0.0016	0.007	-0.247	0.805	-0.014	0.011
Inventory_Vel_Days	0.0005	0.001	0.702	0.483	-0.001	0.002
ROG_Market_Capitalisation_perc	-0.0002	0.002	-0.138	0.890	-0.003	0.003
Creditors_Vel_Days	0.0006	0.000	1.309	0.191	-0.000	0.002
Debtors_Vel_Days	-0.0013	0.000	-2.735	0.006	-0.002	-0.000
Value_of_Output_to_Gross_Block	-0.0190	0.010	-1.926	0.054	-0.038	0.000

Total_Asset_Turnover_Ratio_Latest	0.0280	0.036	0.775	0.438	-0.043	0.099
ROG_Cost_of_Prod_perc	-0.0027	0.001	-1.943	0.052	-0.005	2.35e-05
ROG_Net_Worth_perc	-0.0123	0.004	-3.007	0.003	-0.020	-0.004
Rev_earn_in_forex	0.0011	0.002	0.672	0.502	-0.002	0.004
Cash_Flow_From_Fin	0.0010	0.005	0.204	0.839	-0.009	0.011
Equity_Paid_Up	-0.0013	0.004	-0.372	0.710	-0.008	0.006
Capital_exp_in_forex	-0.0928	0.065	-1.433	0.152	-0.220	0.034
Rev_exp_in_forex	0.0041	0.002	2.125	0.034	0.000	0.008
Other_Income	0.0003	0.007	0.040	0.968	-0.014	0.015
Selling_Cost	-0.0113	0.011	-1.005	0.315	-0.033	0.011
Cash_Flow_From_Inv	-0.0024	0.005	-0.475	0.635	-0.012	0.007
Value_of_Output_to_Total_Assets	0.0934	0.190	0.493	0.622	-0.278	0.465
Market_Capitalisation	-0.0006	0.000	-2.618	0.009	-0.001	-0.000
ROG_Total_Assets_perc	-0.0151	0.007	-2.035	0.042	-0.030	-0.001
Book_Value_Unit_Curr	-0.1489	0.012	-12.512	0.000	-0.172	-0.126
ROG_Capital_Employed_perc	0.0115	0.006	1.834	0.067	-0.001	0.024
Total_Debt	0.0013	0.001	2.598	0.009	0.000	0.002
CEPS_annualised_Unit_Curr	-0.1013	0.039	-2.626	0.009	-0.177	-0.026
Net_Working_Capital	-0.0003	0.001	-0.259	0.796	-0.002	0.002
Cash_Flow_From_Oper	0.0006	0.004	0.157	0.875	-0.007	0.009

We can see that few variables are insignificant & may not be useful to discriminate cases of deault

Let us look at the adjusted pseudo R-square value

```
The adjusted pseudo R-square value is 0.570022460723017
```

Adjusted pseudo R-square seems to be lower than Pseudo R-square value which means there are insignificant variables present in the model. Let’s try & remove variables whose p value is greater than 0.05 & rebuild our model

Model 2 - Logistic Regression (Statsmodel)

Logit Regression Results

Dep. Variable:	default	**No. Observations:**	2402
Model:	Logit	**Df Residuals:**	2389
Method:	MLE	**Df Model:**	12
Date:	Sun, 20 Dec 2020	**Pseudo R-squ.:**	0.6031
Time:	09:47:46	**Log-Likelihood:**	-326.87
converged:	True	**LL-Null:**	-823.47
Covariance Type:	nonrobust	**LLR p-value:**	5.426e-205

	coef	**std err**	**z**	**P>|z|**	**[0.025**	**0.975]**
Intercept	-0.9810	0.136	-7.196	0.000	-1.248	-0.714
Interest_Cover_Ratio_Latest	-0.0019	0.001	-2.185	0.029	-0.004	-0.000
Debtors_Vel_Days	-0.0010	0.000	-2.290	0.022	-0.002	-0.000
Value_of_Output_to_Gross_Block	-0.0215	0.009	-2.387	0.017	-0.039	-0.004
ROG_Cost_of_Prod_perc	-0.0028	0.001	-2.066	0.039	-0.005	-0.000
ROG_Net_Worth_perc	-0.0132	0.004	-3.260	0.001	-0.021	-0.005
Rev_exp_in_forex	0.0031	0.002	2.001	0.045	6.37e-05	0.006
Market_Capitalisation	-0.0006	0.000	-2.933	0.003	-0.001	-0.000
ROG_Total_Assets_perc	-0.0139	0.007	-1.924	0.054	-0.028	0.000
Book_Value_Unit_Curr	-0.1523	0.011	-13.325	0.000	-0.175	-0.130
ROG_Capital_Employed_perc	0.0127	0.006	2.111	0.035	0.001	0.025
Total_Debt	0.0010	0.000	3.068	0.002	0.000	0.002
CEPS_annualised_Unit_Curr	-0.0940	0.036	-2.641	0.008	-0.164	-0.024

We can see that all variables are significant & may be useful to discriminate cases of deault.

Let us also check the multicollinearity of the model using Variance Inflation Factor (VIF) for the predictor variables.

	variables	VIF
0	Interest_Cover_Ratio_Latest	1.048556
1	Debtors_Vel_Days	1.104037
3	ROG_Cost_of_Prod_perc	1.208945
2	Value_of_Output_to_Gross_Block	1.265036
10	Total_Debt	1.708669
5	Rev_exp_in_forex	1.743560
6	Market_Capitalisation	1.852071
4	ROG_Net_Worth_perc	1.889759
7	ROG_Total_Assets_perc	3.016602
8	Book_Value_Unit_Curr	3.033404
11	CEPS_annualised_Unit_Curr	3.171242
9	ROG_Capital_Employed_perc	3.203777

We can see that multicollinearity still exists but let's not drop them as VIFs are not very high.

```
The adjusted pseudo R-square value is 0.5884863653102299
```

Inference:

- We see that adjusted R sq is now close to Rsq, thus suggesting lesser insignificant variables in the model
- We also notice that current model has no insignificant variables and can be used for prediction purposes.

Checking the coefficients

```
Intercept                           -0.980968
Interest_Cover_Ratio_Latest         -0.001881
Debtors_Vel_Days                    -0.001010
Value_of_Output_to_Gross_Block      -0.021505
ROG_Cost_of_Prod_perc               -0.002791
ROG_Net_Worth_perc                  -0.013181
Rev_exp_in_forex                     0.003073
Market_Capitalisation               -0.000633
ROG_Total_Assets_perc               -0.013893
Book_Value_Unit_Curr                -0.152321
ROG_Capital_Employed_perc            0.012736
Total_Debt                           0.000971
CEPS_annualised_Unit_Curr           -0.093962
dtype: float64
```

- Positive coefficient values means, higer that particular variable, more the chance of default.
- Similarly, less profitable and more chance of default.

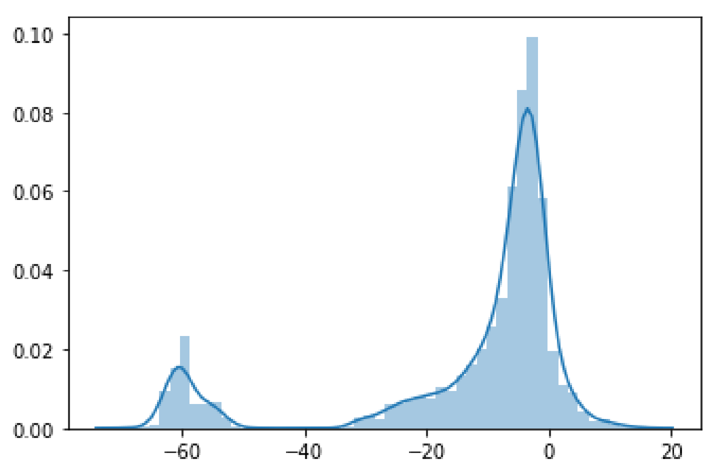
Checking the descriptive statistics of predicted probabilities

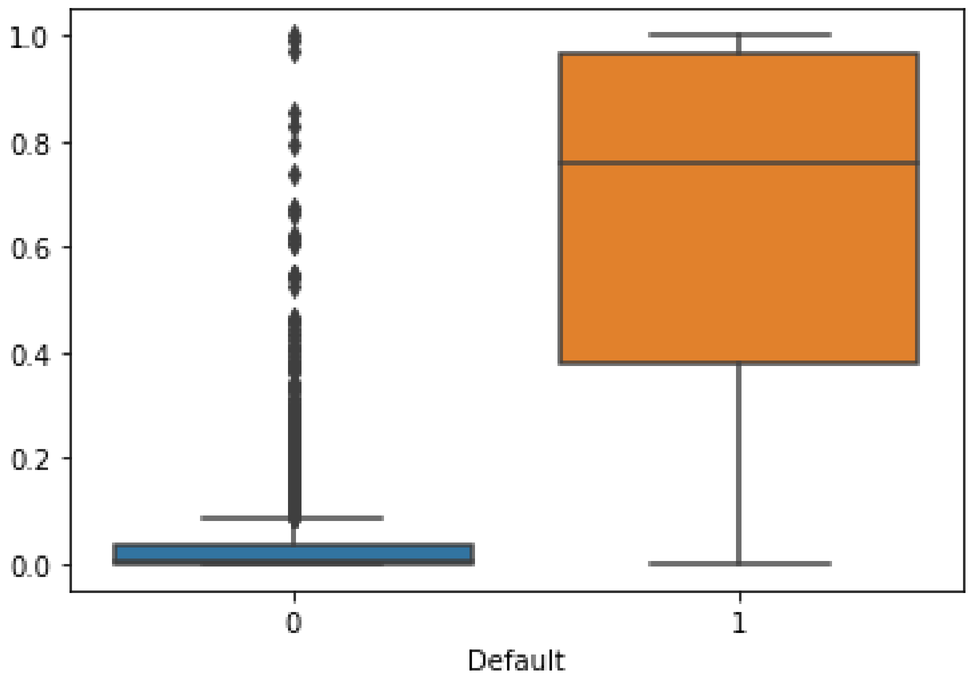
```
array([1.46583616e-30, 6.14263635e-25, 1.50882736e-06, 5.0543842
6e-03, 6.95997211e-02, 9.99999654e-01])
```

1.7 Validate the Model on Test Dataset and state the performance matrices

Prediction on the Data

Let us first check the distribution plot of the logit function values

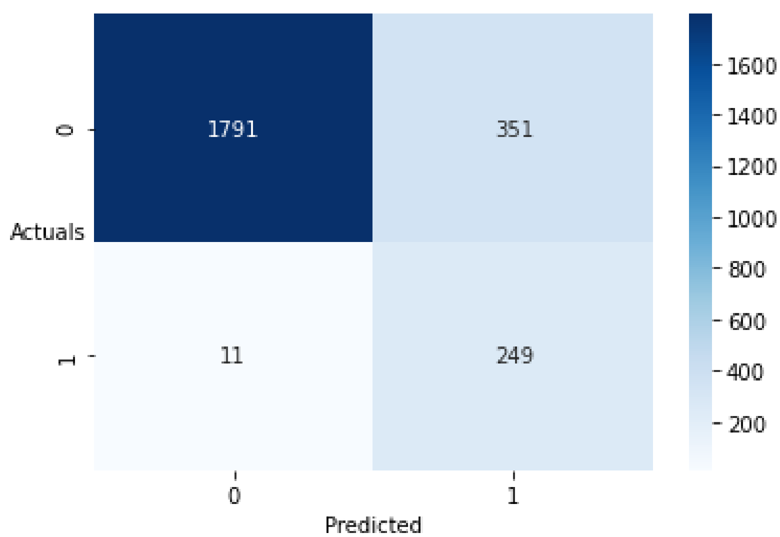




Inference:

From the above boxplot, we need to decide on one such value of a cut-off which will give us the most reasonable descriptive power of the model. Let us take a cut-off of 0.07 and check.

Checking the accuracy of the model using confusion matrix for training set



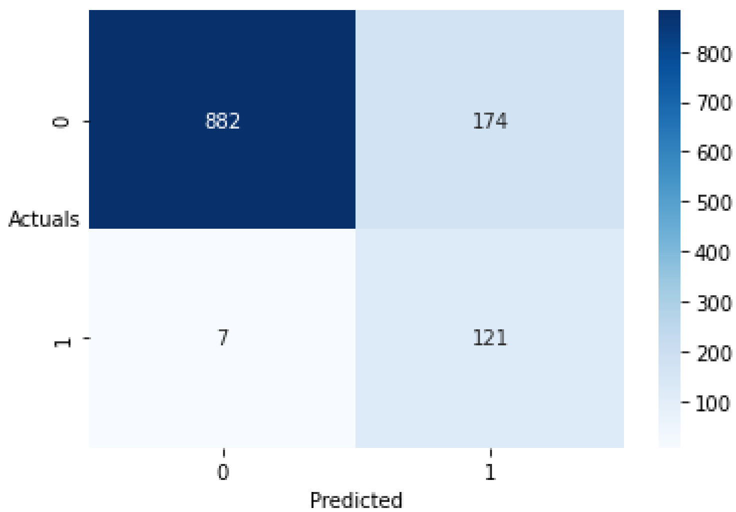

```
True Negative: 1791
False Positives: 351
False Negatives: 11
True Positives: 249
```

	precision	recall	f1-score	support
0	0.994	0.836	0.908	2142
1	0.415	0.958	0.579	260
accuracy			0.849	2402
macro avg	0.704	0.897	0.744	2402
weighted avg	0.931	0.849	0.873	2402

Inference:

- As observed above, accuracy of the model i.e. %overall correct predictions is 84%
- Sensitivity of the model is 95% i.e. 95% of those defaulted were correctly identified as defaulters by the model

Checking the accuracy of the model using confusion matrix for test set




```
True Negative: 882
False Positives: 174
False Negatives: 7
True Positives: 121
```

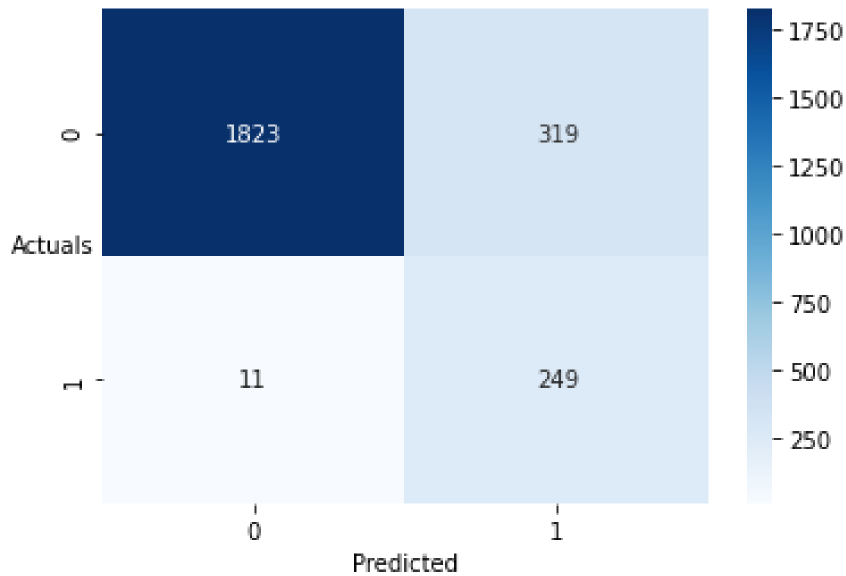
	precision	recall	f1-score	support
0	0.992	0.835	0.907	1056
1	0.410	0.945	0.572	128
accuracy			0.847	1184
macro avg	0.701	0.890	0.740	1184
weighted avg	0.929	0.847	0.871	1184

Inference:

- As observed above, accuracy of the model i.e. %overall correct predictions is 84%
- Sensitivity of the model is 94% i.e. 94% of those defaulted were correctly identified as defaulters by the model

Let us take a cut-off of 0.08 and check if our predictions have improved

Checking the accuracy of the model using confusion matrix for training set



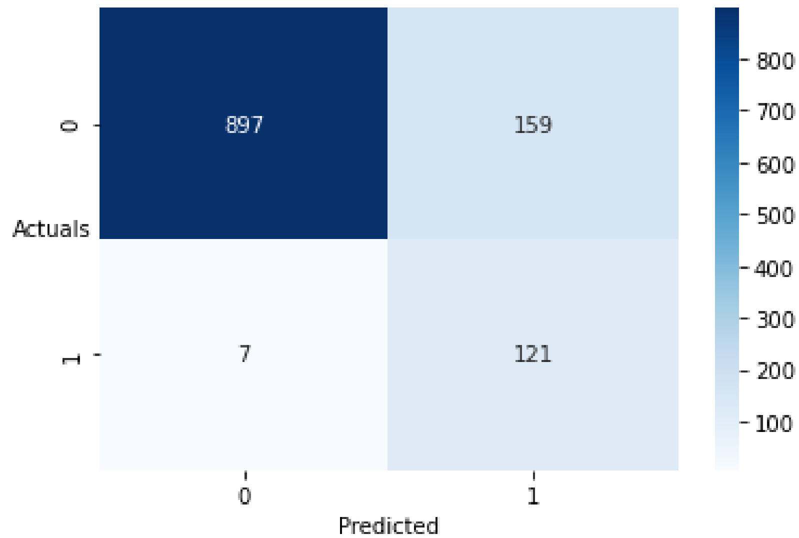

```
True Negative: 1823
False Positives: 319
False Negatives: 11
True Positives: 249
```

	precision	recall	f1-score	support
0	0.994	0.851	0.917	2142
1	0.438	0.958	0.601	260
accuracy			0.863	2402
macro avg	0.716	0.904	0.759	2402
weighted avg	0.934	0.863	0.883	2402

Inference:

- Accuracy of the model i.e. %overall correct predictions has increased from 84% to 86% and the sensitivity of the model is same as 95%.

Checking the accuracy of the model using confusion matrix for test set




```
True Negative: 897
False Positives: 159
False Negatives: 7
True Positives: 121
```

	precision	recall	f1-score	support
0	0.992	0.849	0.915	1056
1	0.432	0.945	0.593	128
accuracy			0.860	1184
macro avg	0.712	0.897	0.754	1184
weighted avg	0.932	0.860	0.880	1184

Inference:

- Accuracy of the model i.e. %overall correct predictions is 86% & sensitivity of the model stands at 94%
- We may choose cutoff of 0.08 as it gave higher model sensitivity & overall accuracy of the model in test dataset

Model 3 - Logistic Regression (Sklearn)

Classification report for Train:

```
Confusion Matrix
 [[2107   35]
 [  74  186]]


              precision    recall  f1-score   support

           0      0.966     0.984     0.975      2142
           1      0.842     0.715     0.773       260

    accuracy                          0.955      2402
   macro avg      0.904     0.850     0.874      2402
weighted avg      0.953     0.955     0.953      2402
```

Classification report for Test:

```
Confusion Matrix
 [[1031   25]
 [  35   93]]


              precision    recall  f1-score   support

           0      0.967     0.976     0.972      1056
           1      0.788     0.727     0.756       128

    accuracy                          0.949      1184
   macro avg      0.878     0.851     0.864      1184
weighted avg      0.948     0.949     0.948      1184
```

- Accuracy Score for Train set is **0.9546211490424646**
- Accuracy Score for Test set is **0.9493243243243243**

Inference:

- Accuracy of the Logistic Model i.e. %overall correct predictions is 94% & sensitivity of the model stands at 73%
- 73% of those defaulted were correctly identified as defaulters by the model

1.8 Build a Random Forest Model on Train Dataset

Model 4 - Random Forest

Classification report for Train:

```
Confusion Matrix
 [[2128   14]
 [  31  229]]
              precision    recall  f1-score   support

           0      0.986     0.993     0.990      2142
           1      0.942     0.881     0.911       260

    accuracy                          0.981      2402
   macro avg      0.964     0.937     0.950      2402
weighted avg      0.981     0.981     0.981      2402
```

1.9 Validate the Random Forest Model on test Dataset and state the performance matrices

Classification report for Test:

```
Confusion Matrix
 [[1053    3]
 [  20  108]]
              precision    recall  f1-score   support

           0      0.981     0.997     0.989      1056
           1      0.973     0.844     0.904       128

    accuracy                          0.981      1184
   macro avg      0.977     0.920     0.946      1184
weighted avg      0.980     0.981     0.980      1184
```

- Accuracy Score for Train set is **0.9812656119900083**
- Accuracy Score for Test set is **0.9805743243243243**

Inference:

- Accuracy of the Random Forest Model i.e. %overall correct predictions is 98% & sensitivity of the model stands at 84%
- 84% of those defaulted were correctly identified as defaulters by the model.

1.10 Build an LDA Model on Train Dataset

Model 5 - Linear Discriminant Analysis (LDA)

```
from sklearn.discriminant_analysis import LinearDiscriminantAnalysis
from sklearn.metrics import
confusion_matrix,classification_report,roc_auc_score,roc_curve,accuracy_score

lda = LinearDiscriminantAnalysis()
model_lda = lda.fit(X_train,y_train)

lda_ypred_train = model_lda.predict(X_train)
lda_ypred_test = model_lda.predict(X_test)
```

1.11 Validate the LDA Model on test Dataset and state the performance matrices

Model evaluation on test data set

Classification report for Test:

```
Confusion Matrix
 [[1041   15]
 [  98   30]]

              precision    recall  f1-score   support

           0       0.91      0.99      0.95      1056
           1       0.67      0.23      0.35       128

    accuracy                           0.90      1184
   macro avg       0.79      0.61      0.65      1184
weighted avg       0.89      0.90      0.88      1184
```

- Accuracy Score for Train set is **0.906744379683597**
- Accuracy Score for Test set is **0.9045608108108109**

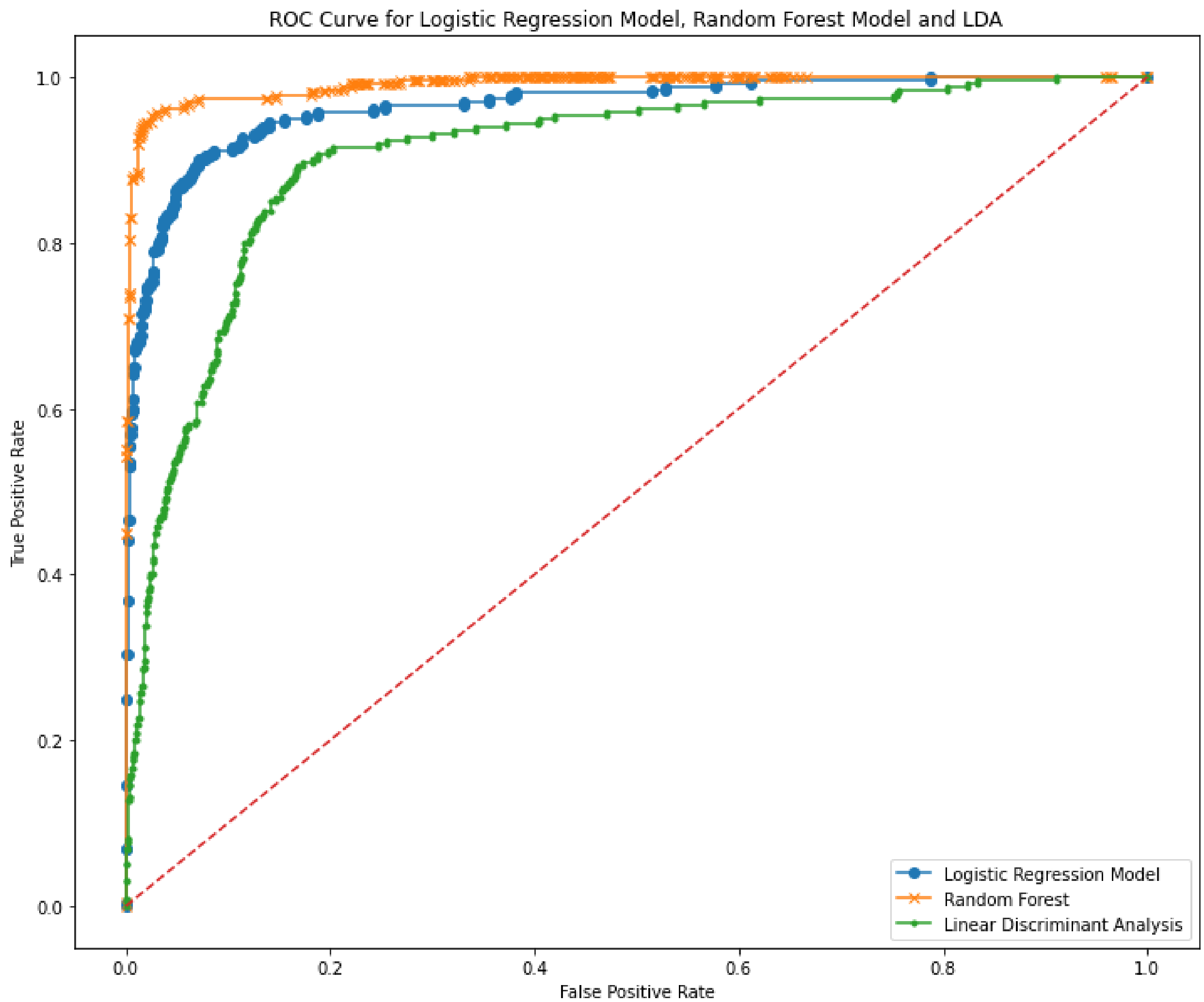
Inference:

- Accuracy of the Random Forest Model i.e. %overall correct predictions is 90% & sensitivity of the model stands at 23%
- 23% of those defaulted were correctly identified as defaulters by the model which is very poor prediction.

1.12 Compare the performances of all the three models (include ROC Curve)

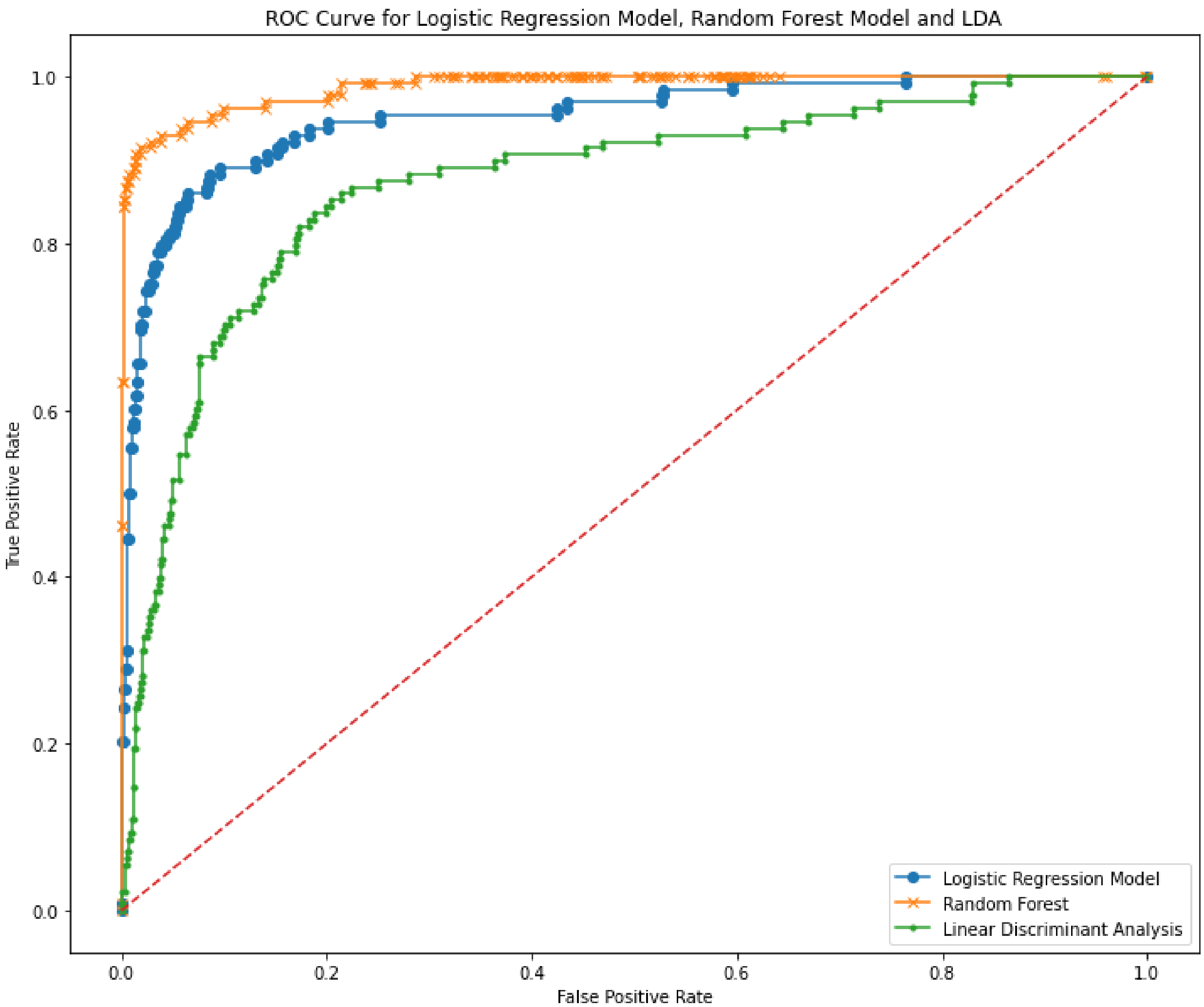
Train:

```
AUC for Logistic Regression Train Model_2 is 0.9644185879479996
AUC for Random Forest Train Model is 0.9908460820225525
AUC for Linear Discriminant Analysis Train Model is 0.90553939524
52775
```



Test:

AUC for Logistic Regression Test Model_2 is **0.9505282315340909**
AUC for Random Forest Test Model is **0.9876302083333333**
AUC for Linear Discriminant Analysis Test Model is **0.8722034801136364**



1.13 Recommendation:

- It appears that all models performed well for the defaulters, with precision, recall metrics all above 0.7.
- Both models' performance is almost the same for the non-defaulters and arguably the "more important" classification of whether a company is going to default or not.
- AUC for Random Forest Analysis Test Model is 98%
- AUC for Logistic Regression Model Test Model is 95%
- Accuracy of the Random Forest Model i.e. %overall correct predictions is 98% & sensitivity of the model stands at 84%
- Recall: 84% of those defaulted were correctly identified as defaulters by the model.
- Accuracy of the Logistic Model i.e. %overall correct predictions is 94% & sensitivity of the model stands at 73%
- Recall: 73% of those defaulted were correctly identified as defaulters by the model.

Part 2 Market Risk:

The dataset contains 6 years of information (weekly stock information) on the stock prices of 10 different Indian Stocks. Calculate the mean and standard deviation on the stock returns and share insights.

Data set for the Problem: Market+Risk+Dataset.csv

Importing the dataset

Data set:

	Date	Infosys	Indian Hotel	Mahindra & Mahindra	Axis Bank	SAIL	Shree Cement	Sun Pharma	Jindal Steel	Idea Vodafone	Jet Airways
0	31-03-2014	264	69	455	263	68	5543	555	298	83	278
1	07-04-2014	257	68	458	276	70	5728	610	279	84	303
2	14-04-2014	254	68	454	270	68	5649	607	279	83	280
3	21-04-2014	253	68	488	283	68	5692	604	274	83	282
4	28-04-2014	256	65	482	282	63	5582	611	238	79	243

Exploratory Data Analysis:

The number of rows (observations) is 314

The number of columns (variables) is 11

Data types of all variables

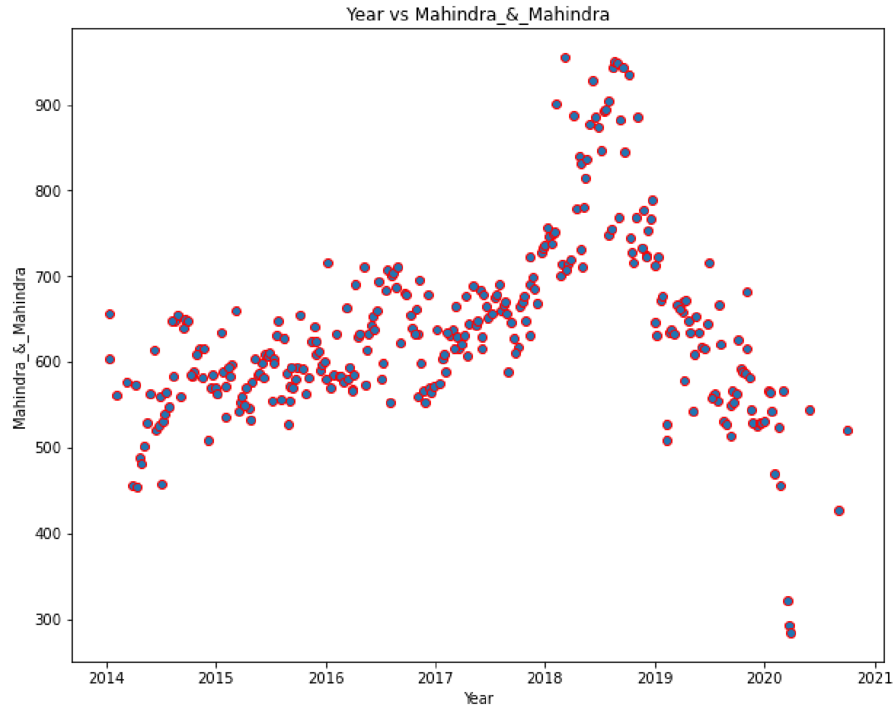
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 314 entries, 0 to 313

Data columns (total 11 columns):

 #   Column              Non-Null Count  Dtype
---  ------              --------------  -----
 0   Date                314 non-null    object
 1   Infosys             314 non-null    int64
 2   Indian_Hotel        314 non-null    int64
 3   Mahindra_&_Mahindra 314 non-null    int64
 4   Axis_Bank           314 non-null    int64
 5   SAIL                314 non-null    int64
 6   Shree_Cement        314 non-null    int64
 7   Sun_Pharma          314 non-null    int64
 8   Jindal_Steel        314 non-null    int64
 9   Idea_Vodafone       314 non-null    int64
 10  Jet_Airways         314 non-null    int64

dtypes: int64(10), object(1)
memory usage: 27.1+ KB
```

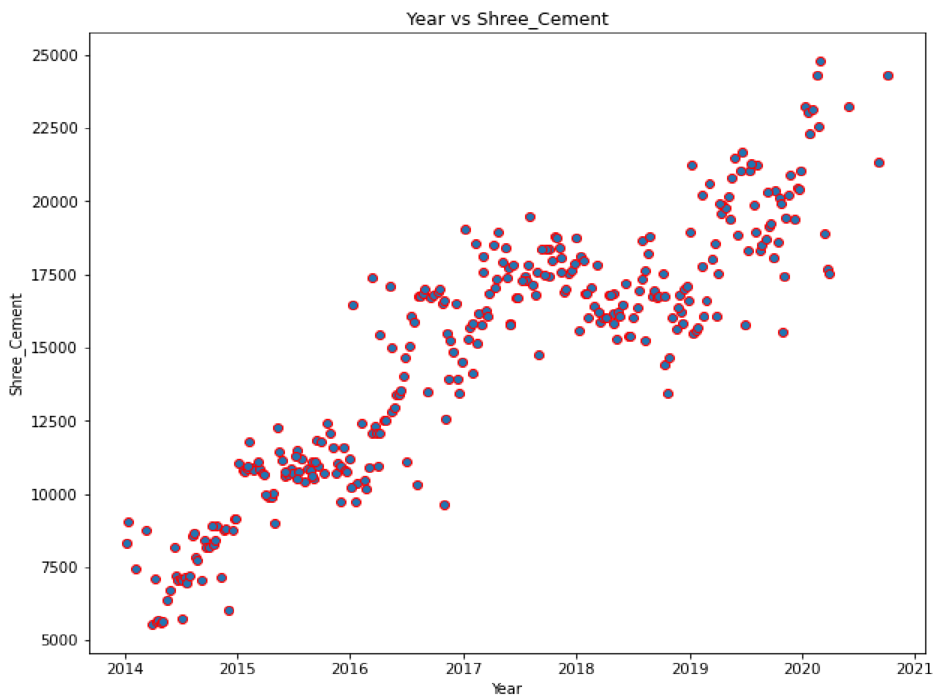
Now, let us check the basic measures of descriptive statistics for the continuous variables

5 Point summary:

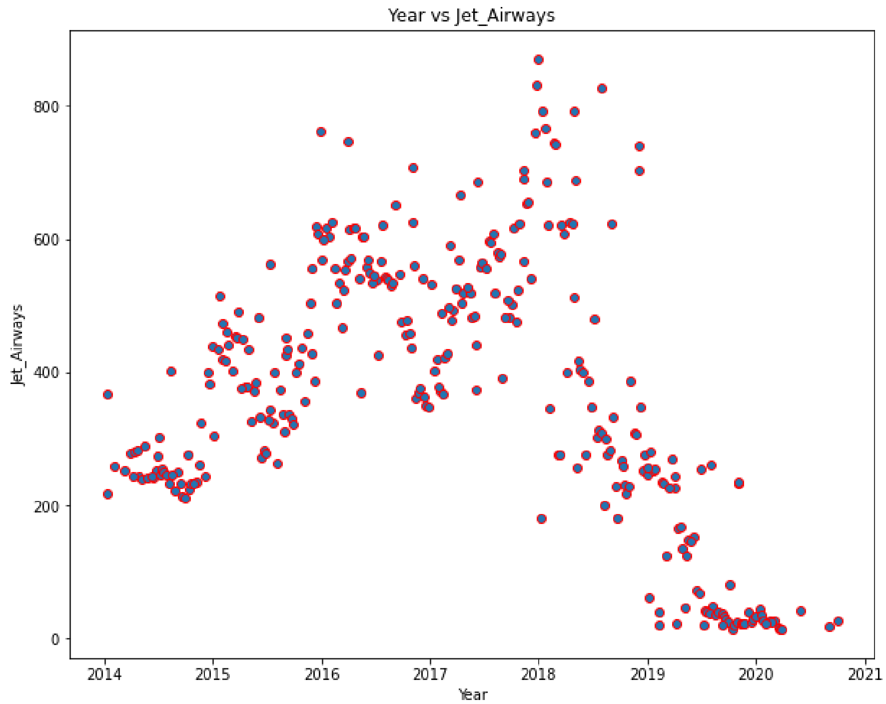
	count	mean	std	min	25%	50%	75%	max
Infosys	314.0	511.340764	135.952051	234.0	424.00	466.5	630.75	810.0
Indian_Hotel	314.0	114.560510	22.509732	64.0	96.00	115.0	134.00	157.0
hindra_&_Mahindra	314.0	636.678344	102.879975	284.0	572.00	625.0	678.00	956.0
Axis_Bank	314.0	540.742038	115.835569	263.0	470.50	528.0	605.25	808.0
SAIL	314.0	59.095541	15.810493	21.0	47.00	57.0	71.75	104.0
Shree_Cement	314.0	14806.410828	4288.275085	5543.0	10952.25	16018.5	17773.25	24806.0
Sun_Pharma	314.0	633.468153	171.855893	338.0	478.50	614.0	785.00	1089.0
Jindal_Steel	314.0	147.627389	65.879195	53.0	88.25	142.5	182.75	338.0
Idea_Vodafone	314.0	53.713376	31.248985	3.0	25.25	53.0	82.00	117.0
Jet_Airways	314.0	372.659236	202.262668	14.0	243.25	376.0	534.00	871.0

2.1 Draw Stock Price Chart for any 2 variables





The above stock is on increasing trend since it does not have much concentration on one particular area.



2.2 Calculate Returns

Steps for calculating returns from prices:

Take logarithms Take differences

Checking the rows & columns of dataset

```
(314, 10)
```

Checking top 5 rows:

	Infosys	Indian_Hotel	Mahindra_&_Mahindra	Axis_Bank	SAIL	Shree_Cement
0	NaN	NaN	NaN	NaN	NaN	NaN
1	-0.026873	-0.014599	0.006572	0.048247	0.028988	0.032831
2	-0.011742	0.000000	-0.008772	-0.021979	-0.028988	-0.013888
3	-0.003945	0.000000	0.072218	0.047025	0.000000	0.007583
4	0.011788	-0.045120	-0.012371	-0.003540	-0.076373	-0.019515

2.3 Calculate Stock Means and Standard Deviation

- **Stock Means: Average returns that the stock is making on a week to week basis**
- **Stock Standard Deviation : It is a measure of volatility meaning the more a stock's returns vary from the stock's average return, the more volatile the stock**

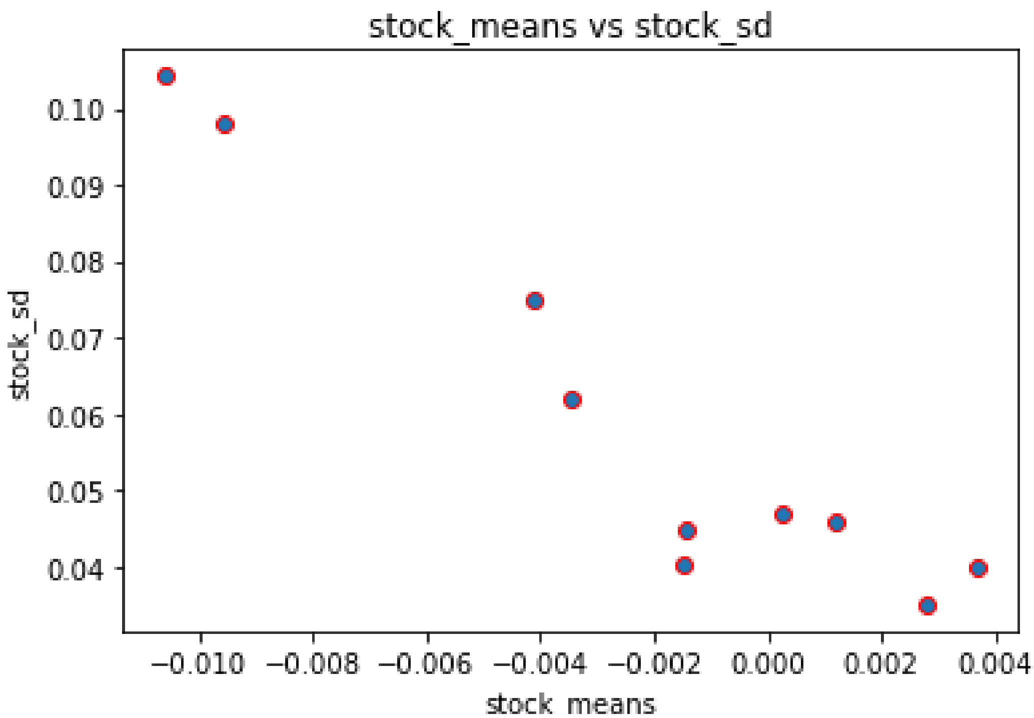
Calculating stock means

```
Infosys                 0.002794
Indian_Hotel            0.000266
Mahindra_&_Mahindra    -0.001506
Axis_Bank               0.001167
SAIL                   -0.003463
Shree_Cement            0.003681
Sun_Pharma             -0.001455
Jindal_Steel           -0.004123
Idea_Vodafone          -0.010608
Jet_Airways            -0.009548

dtype: float64
```

Calculating stock standard deviation

```
Infosys                 0.035070
Indian_Hotel            0.047131
Mahindra_&_Mahindra     0.040169
Axis_Bank               0.045828
SAIL                    0.062188
Shree_Cement            0.039917
Sun_Pharma              0.045033
Jindal_Steel            0.075108
Idea_Vodafone           0.104315
Jet_Airways             0.097972

dtype: float64
```

2.4 Draw a plot of Stock Means vs Standard Deviation and share insights



Insights:

Stock with a lower mean & higher standard deviation do not play a role in a portfolio that has competing stock with more returns & less risk. Thus for the data we have here, we are only left few stocks:

- **One with highest return and lowest risk &**
- **One with lowest risk and highest return**

Therefore from pure ***Returns*** perspective, ***Shree_Cement*** looks good in this dataset & from pure ***Risk*** perspective (as measured by standard deviation), ***Infosys*** followed by ***Shree_Cement*** & ***Mahindra_&_Mahindra*** looks good in this datase

The End