Financial Risk Analytics

Credit risk and Market risk

Project for Great Learning

Aditya Rishi 1/16/22 PGP-DSBA Feb 21

Problem set

Tables

Figures

Problem description: Executive summary

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

Usage

Default

Format

A dataframe with 3587 observations on 68 variables

Introduction

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.

Dataset

'Credit Default Data Dictionary.xlsx

Data file: Company_Data2015-1.xlsx

Source:-

Great Learning

Data dictionary

	#	Field Name	Description	New Field Name
0	1	Co_Code	Company Code	Co_Code
1	2	Co_Name	Company Name	Co_Name
2	3	Networth Next Year	Value of a company as on 2016 - Next Year(difference between the value of total assets and total liabilities)	Networth_Next_Year
3	4	Equity Paid Up	Amount that has been received by the company through the issue of shares to the shareholders	Equity_Paid_Up
4	5	Networth	Value of a company as on 2015 - Current Year	Networth
5	6	Capital Employed	Total amount of capital used for the acquisition of profits by a company	Capital_Employed
6	7	Total Debt	The sum of money borrowed by the company and is due to be paid	Total_Debt
7	8	Gross Block	Total value of all of the assets that a company owns	Gross_Block
8	9	Net Working Capital	The difference between a company's current assets (cash, accounts receivable, inventories of raw materials and finished goods) and its current liabilities (accounts payable).	Net_Working_Capital
9	10	Current Assets	All the assets of a company that are expected to be sold or used as a result of standard business operations over the next year.	Curr_Assets
10	11	Current Liabilities and Provisions	Short-term financial obligations that are due within one year (includes amount that is set aside cover a future liability)	Curr_Liab_and_Prov
11	12	Total Assets/Liabilities	Ratio of total assets to liabailities of the company	Total_Assets_to_Liab
12	13	Gross Sales	The grand total of sale transactions within the accounting period	Gross_Sales
13	14	Net Sales	Gross sales minus returns, allowances, and discounts	Net_Sales
14	15	Other Income	Income realized from non-business activities (e.g. sale of long term asset)	Other_Income
15	16	Value Of Output	Product of physical output of goods and services produced by company and its market price	Value_Of_Output
16	17	Cost of Production	Costs incurred by a business from manufacturing a product or providing a service	Cost_of_Prod
17	18	Selling Cost	Costs which are made to create the demand for the product (advertising expenditures, packaging and styling, salaries, commissions and travelling expenses of sales personnel, and the cost of shops ...	Selling_Cost
18	19	PBIDT	Profit Before Interest, Depreciation & Taxes	PBIDT
19	20	PBDT	Profit Before Depreciation and Tax	PBDT
20	21	PBIT	Profit before interest and taxes	PBIT
21	22	PBT	Profit before tax	PBT
22	23	PAT	Profit After Tax	PAT
23	24	Adjusted PAT	Adjusted profit is the best estimate of the true profit	Adjusted_PAT
24	26	CP	Commercial paper , a short-term debt instrument to meet short-term liabilities.	CP
25	27	Revenue earnings in forex	Revenue earned in foreign currency	Rev_earn_in_forex
26	28	Revenue expenses in forex	Expenses due to foreign currency transactions	Rev_exp_in_forex
27	29	Capital expenses in forex	Long term investment in forex	Capital_exp_in_forex
28	30	Book Value (Unit Curr)	Net asset value	Book_Value_Unit_Curr
29	31	Book Value (Adj.) (Unit Curr)	Book value adjusted to reflect asset's true fair market value	Book_Value_Adj_Unit_Curr
30	32	Market Capitalisation	Product of the total number of a company's outstanding shares and the current market price of one share	Market_Capitalisation
31	33	CEPS (annualised) (Unit Curr)	Cash Earnings per Share, profitability ratio that measures the financial performance of a company by calculating cash flows on a per share basis	CEPS_annualised_Unit_Curr
32	34	Cash Flow From Operating Activities	Use of cash from ongoing regular business activities	Cash_Flow_From_Opr
33	35	Cash Flow From Investing Activities	Cash used in the purchase of non-current assets–or long-term assets– that will deliver value in the future	Cash_Flow_From_Inv
34	36	Cash Flow From Financing Activities	Net flows of cash that are used to fund the company (transactions involving debt, equity, and dividends)	Cash_Flow_From_Fin
35	37	ROG-Net Worth (%)	Rate of Growth - Networth	ROG_Net_Worth_perc
36	38	ROG-Capital Employed (%)	Rate of Growth - Capital Employed	ROG_Capital_Employed_perc
37	39	ROG-Gross Block (%)	Rate of Growth - Gross Block	ROG_Gross_Block_perc
38	40	ROG-Gross Sales (%)	Rate of Growth - Gross Sales	ROG_Gross_Sales_perc

Tab 1

Data dictionary

39	41	ROG-Net Sales (%)	Rate of Growth - Net Sales	ROG_Net_Sales_perc
40	42	ROG-Cost of Production (%)	Rate of Growth - Cost of Production	ROG_Cost_of_Prod_perc
41	43	ROG-Total Assets (%)	Rate of Growth - Total Assets	ROG_Total_Assets_perc
42	44	ROG-PBIDT (%)	Rate of Growth- PBIDT	ROG_PBIDT_perc
43	45	ROG-PBDT (%)	Rate of Growth- PBDT	ROG_PBDT_perc
44	46	ROG-PBIT (%)	Rate of Growth- PBIT	ROG_PBIT_perc
45	47	ROG-PBT (%)	Rate of Growth- PBT	ROG_PBT_perc
46	48	ROG-PAT (%)	Rate of Growth- PAT	ROG_PAT_perc
47	49	ROG-CP (%)	Rate of Growth- CP	ROG_CP_perc
48	50	ROG-Revenue earnings in forex (%)	Rate of Growth - Revenue earnings in forex	ROG_Rev_earn_in_forex_perc
49	51	ROG-Revenue expenses in forex (%)	Rate of Growth - Revenue expenses in forex	ROG_Rev_exp_in_forex_perc

50	52	ROG-Market Capitalisation (%)	Rate of Growth - Market Capitalisation	ROG_Market_Capitalisation_perc
51	53	Current Ratio[Latest]	Liquidity ratio, company's ability to pay short-term obligations or those due within one year	Curr_Ratio_Latest
52	54	Fixed Assets Ratio[Latest]	Solvency ratio, the capacity of a company to discharge its obligations towards long-term lenders indicating	Fixed_Assets_Ratio_Latest
53	55	Inventory Ratio[Latest]	Activity ratio, specifies the number of times the stock or inventory has been replaced and sold by the company	Inventory_Ratio_Latest
54	56	Debtors Ratio[Latest]	Measures how quickly cash debtors are paying back to the company	Debtors_Ratio_Latest
55	57	Total Asset Turnover Ratio[Latest]	The value of a company's revenues relative to the value of its assets	Total_Asset_Turnover_Ratio_Latest
56	58	Interest Cover Ratio[Latest]	Determines how easily a company can pay interest on its outstanding debt	Interest_Cover_Ratio_Latest
57	59	PBIDTM (%)[Latest]	Profit before Interest Depreciation and Tax Margin	PBIDTM_perc_Latest
58	60	PBITM (%)[Latest]	Profit Before Interest Tax Margin	PBITM_perc_Latest

59	61	PBDTM (%)[Latest]	Profit Before Depreciation Tax Margin	PBDTM_perc_Latest
60	62	CPM (%)[Latest]	Cost per thousand (advertising cost)	CPM_perc_Latest
61	63	APATM (%)[Latest]	After tax profit margin	APATM_perc_Latest
62	64	Debtors Velocity (Days)	Average days required for receiving the payments	Debtors_Vel_Days
63	65	Creditors Velocity (Days)	Average number of days company takes to pay suppliers	Creditors_Vel_Days
64	66	Inventory Velocity (Days)	Average number of days the company needs to turn its inventory into sales	Inventory_Vel_Days
65	67	Value of Output/Total Assets	Ratio of Value of Output (market value) to Total Assets	Value_of_Output_to_Total_Assets
66	68	Value of Output/Gross Block	Ratio of Value of Output (market value) to Gross Block	Value_of_Output_to_Gross_Block

Tab 2

Dataset

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

...

Tab 3

Data description

	Co_Code	Networth Next Year	Equity Paid Up	Networth	Capital Employed	Total Debt	Gross Block	Net Working Capital	Current Assets	Current Liabilities and Provisions	...
count	3586.000000	3586.000000	3586.000000	3586.000000	3586.000000	3586.000000	3586.000000	3586.000000	3586.000000	3586.000000	...
mean	16065.388734	725.045251	62.966584	649.746299	2799.611054	1994.823779	594.178829	410.809665	1960.349172	391.992078	...
std	19776.817379	4769.681004	778.761744	4091.988792	26975.135385	23652.842746	4871.547802	6301.218546	22577.570829	2675.001631	...
min	4.000000	-8021.600000	0.000000	-7027.480000	-1824.750000	-0.720000	-41.190000	-13162.420000	-0.910000	-0.230000	...
25%	3029.250000	3.985000	3.750000	3.892500	7.602500	0.030000	0.570000	0.942500	4.000000	0.732500	...
50%	6077.500000	19.015000	8.290000	18.580000	39.090000	7.490000	15.870000	10.145000	24.540000	9.225000	...
75%	24269.500000	123.802500	19.517500	117.297500	226.605000	72.350000	131.895000	61.175000	135.277500	65.650000	...
max	72493.000000	111729.100000	42263.460000	81657.350000	714001.250000	652823.810000	128477.590000	223257.560000	721166.000000	83232.980000	...

Tab 4

Dataset with cleaned column names

	Co_Code	Co_Name	Networth_Next_Year	Equity_Paid_Up	Networth	Capital_Employed	Total_Debt	Gross_Block	Net_Working_Capital	Curr_Assets	...
0	16974	Hind.Cables	-8021.60	419.36	-7027.48	-1007.24	5936.03	474.30	-1076.34	40.50	...
1	21214	Tata Tele. Mah.	-3986.19	1954.93	-2968.08	4458.20	7410.18	9070.86	-1098.88	486.86	...
2	14852	ABG Shipyard	-3192.58	53.84	506.86	7714.68	6944.54	1281.54	4496.25	9097.64	...
3	2439	GTL	-3054.51	157.30	-623.49	2353.88	2326.05	1033.69	-2612.42	1034.12	...
4	23505	Bharati Defence	-2967.36	50.30	-1070.83	4675.33	5740.90	1084.20	1836.23	4685.81	...

Tab 5

Checking data types

```
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_Opr                  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
```

Tab 6

Missing values

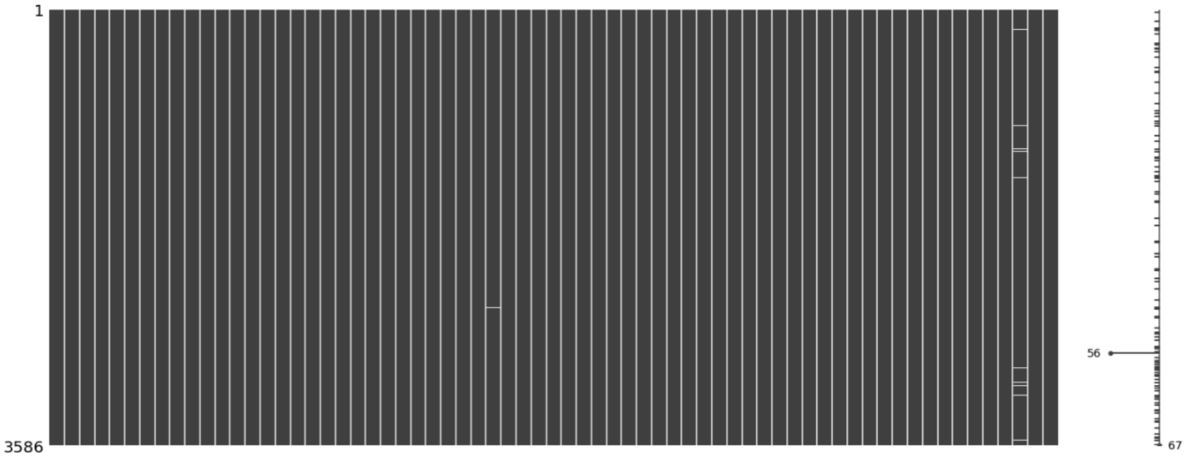


Fig 1

- Treated missing values in Inventory_Vel_Days variable
- Dropped rows with missing values after fixing Inventory_Vel_Days variable

Shape of resultant dataset after missing value treatment

```
(3581, 67)
```

Checking for duplicate data

```
The credit risk dataset has 0 duplicate values
```

Dropping unrequired columns

Dropped columns 'Co_Code' and 'Co_Name'

	Networth_Next_Year	Equity_Paid_Up	Networth	Capital_Employed	Total_Debt	Gross_Block	Net_Working_Capital
0	-8021.60	419.36	-7027.48	-1007.24	5936.03	474.30	-1076.34
1	-3986.19	1954.93	-2968.08	4458.20	7410.18	9070.86	-1098.88
2	-3192.58	53.84	506.86	7714.68	6944.54	1281.54	4496.25
3	-3054.51	157.30	-623.49	2353.88	2326.05	1033.69	-2612.42
4	-2967.36	50.30	-1070.83	4675.33	5740.90	1084.20	1836.23

5 rows × 65 columns

Tab 7

Making the dependent variable

Creditors_Vel_Days	Inventory_Vel_Days	Value_of_Output_to_Total_Assets	Value_of_Output_to_Gross_Block	Default
0	45.0	0.00	0.00	1
101	2.0	0.31	0.24	1
558	0.0	-0.03	-0.26	1
63	2.0	0.24	1.90	1
346	0.0	0.01	0.05	1

Tab 8

Converted to X and y

Where y is the 'Default' column, while X is rest of the dataset

Feature Selection

Since there are too many columns, we need to determine the columns which are related and eliminate them if possible. We will use VIF to determine the collinearity and eliminate using a threshold of 5.

```
dropping 'PBIDT' at index: 16
dropping 'PBDT' at index: 16
dropping 'APATM_perc_Latest' at index: 57
dropping 'ROG_Gross_Sales_perc' at index: 34
dropping 'Net_Sales' at index: 11
dropping 'PBDTM_perc_Latest' at index: 53
dropping 'Value_Of_Output' at index: 12
dropping 'ROG_Total_Assets_perc' at index: 34
dropping 'Capital_Employed' at index: 3
dropping 'Gross_Sales' at index: 9
dropping 'Total_Debt' at index: 3
dropping 'ROG_PBDT_perc' at index: 32
dropping 'ROG_PBIT_perc' at index: 32
dropping 'CP' at index: 15
dropping 'PAT' at index: 13
dropping 'ROG_PBIDT_perc' at index: 29
dropping 'Total_Assets_to_Liab' at index: 7
dropping 'ROG_PBT_perc' at index: 28
dropping 'PBT' at index: 11
dropping 'PBIT' at index: 10
dropping 'Cost_of_Prod' at index: 8
dropping 'Networth' at index: 2
dropping 'PBITM_perc_Latest' at index: 36
dropping 'ROG_CP_perc' at index: 25
dropping 'Cash_Flow_From_Fin' at index: 18
dropping 'ROG_Net_Worth_perc' at index: 18
dropping 'Gross_Block' at index: 2
```

```
dropping 'Fixed_Assets_Ratio_Latest' at index: 26
dropping 'Curr_Liab_and_Prov' at index: 4
dropping 'Networth_Next_Year' at index: 0
dropping 'Adjusted_PAT' at index: 5
Remaining variables:
Index(['Equity_Paid_Up', 'Net_Working_Capital', 'Curr_Assets', 'Other_Income',
       'Selling_Cost', 'Rev_earn_in_forex', 'Rev_exp_in_forex',
       'Capital_exp_in_forex', 'Book_Value_Unit_Curr',
       'Book_Value_Adj_Unit_Curr', 'Market_Capitalisation',
       'CEPS_annualised_Unit_Curr', 'Cash_Flow_From_Opr', 'Cash_Flow_From_Inv',
       'ROG_Capital_Employed_perc', 'ROG_Gross_Block_perc',
       'ROG_Net_Sales_perc', 'ROG_Cost_of_Prod_perc', 'ROG_PAT_perc',
       'ROG_Rev_earn_in_forex_perc', 'ROG_Rev_exp_in_forex_perc',
       'ROG_Market_Capitalisation_perc', 'Curr_Ratio_Latest',
       'Inventory_Ratio_Latest', 'Debtors_Ratio_Latest',
       'Total_Asset_Turnover_Ratio_Latest', 'Interest_Cover_Ratio_Latest',
       'PBIDTM_perc_Latest', 'CPM_perc_Latest', 'Debtors_Vel_Days',
       'Creditors_Vel_Days', 'Inventory_Vel_Days',
       'Value_of_Output_to_Total_Assets', 'Value_of_Output_to_Gross_Block'],
      dtype='object')
```

Tab 9

Dataset X after feature selection

	Equity_Paid_Up	Net_Working_Capital	Curr_Assets	Other_Income	Selling_Cost	Rev_earn_in_forex	Rev_exp_in_forex
0	419.36	-1076.34	40.50	7.60	0.00	0.00	0.00
1	1954.93	-1098.88	486.86	46.27	40.51	6.35	143.42
2	53.84	4496.25	9097.64	9.55	54.83	0.00	86.36
3	157.30	-2612.42	1034.12	223.85	3.34	0.89	28.88
4	50.30	1836.23	4685.81	9.82	1.97	0.00	15.62
...	...	...	...	...	...	...	...
3581	501.30	0.00	444633.50	8996.35	187.47	0.00	0.00
3582	296.50	2503.86	11554.45	2008.86	249.20	14429.18	19525.06
3583	2427.95	6376.84	89609.82	5815.66	686.53	16009.99	193979.73
3584	8245.46	11449.79	42353.59	2399.39	71.22	3.41	962.27
3585	1998.70	-12145.30	11947.10	5193.00	1555.50	3727.40	3017.20

...

Creditors_Vel_Days	Inventory_Vel_Days	Value_of_Output_to_Total_Assets	Value_of_Output_to_Gross_Block
0	45.000000	0.00	0.00
101	2.000000	0.31	0.24
558	0.000000	-0.03	-0.26
63	2.000000	0.24	1.90
346	0.000000	0.01	0.05
...	...	...	...
0	79.644559	0.60	7.76
53	77.000000	0.29	1.00
30	48.000000	1.42	3.24
69	42.000000	0.36	0.68
74	0.000000	0.42	0.49

Tab 10

Dataset after concatenating X and y

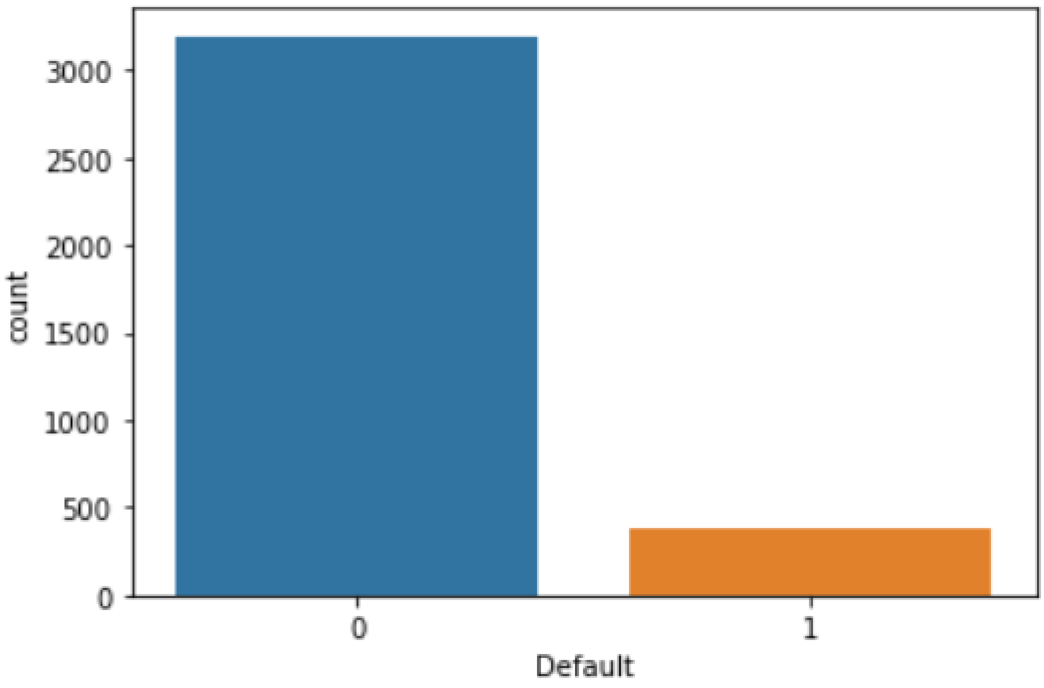
	Equity_Paid_Up	Net_Working_Capital	Curr_Assets	Other_Income	Selling_Cost	Rev_earn_in_forex
0	419.36	-1076.34	40.50	7.60	0.00	0.00
1	1954.93	-1098.88	486.86	46.27	40.51	6.35
2	53.84	4496.25	9097.64	9.55	54.83	0.00
3	157.30	-2612.42	1034.12	223.85	3.34	0.89
4	50.30	1836.23	4685.81	9.82	1.97	0.00
...	...	...	...	...	...	...
3581	501.30	0.00	444633.50	8996.35	187.47	0.00
3582	296.50	2503.86	11554.45	2008.86	249.20	14429.18
3583	2427.95	6376.84	89609.82	5815.66	686.53	16009.99
3584	8245.46	11449.79	42353.59	2399.39	71.22	3.41
3585	1998.70	-12145.30	11947.10	5193.00	1555.50	3727.40

...

Value_of_Output_to_Total_Assets	Value_of_Output_to_Gross_Block	Default
0.00	0.00	1
0.31	0.24	1
-0.03	-0.26	1
0.24	1.90	1
0.01	0.05	1
...	...	...
0.60	7.76	0
0.29	1.00	0
1.42	3.24	0
0.36	0.68	0
0.42	0.49	0

Tab 11

Split of data based on Default Variable




```
0    3195
1     386
Name: Default, dtype: int64
```

Fig 2

Grouped by default

Default	Equity_Paid_Up	Net_Working_Capital	Curr_Assets	Other_Income	Selling_Cost	Rev_earn_in_forex	Rev_exp_in_forex
0	214218.10	1462147.61	6968115.21	171778.53	89962.39	464874.78	912428.20
1	11418.37	10954.66	61538.31	2960.36	1662.93	5483.88	6760.43

Tab 12

Since the data set is very imbalanced. We will apply SMOTE and then we will evaluate all models on both SMOTE as well as NON - SMOTE dataset and compare

```
Before OverSampling the shape of X: (3581, 34)
Before OverSampling the shape of y: (3581,)
Before OverSampling, counts of label '1': 386
Before OverSampling, counts of label '0': 3195
```

```
After OverSampling the shape of X: (6390, 34)
After OverSampling the shape of y: (6390,)
After OverSampling, counts of label '1': 3195
After OverSampling, counts of label '0': 3195
```

Concatenating X and y, so they can be used with Statsmodels library

Train

	Equity_Paid_Up	Net_Working_Capital	Curr_Assets	Other_Income	Selling_Cost	...	Value_of_Output_to_Total_Assets	Value_of_Output_to_Gross_Block	Default
2218	3.06	40.94	66.67	0.25	7.65		0.85	0.70	0
498	5.10	0.39	10.15	1.08	0.01		1.13	2.55	0
1154	5.26	8.87	12.57	0.04	0.00		3.28	18.12	0
1348	6.25	7.51	12.64	0.02	2.14		0.27	4.03	0
3540	289.37	-892.08	3862.90	582.42	576.87		1.22	5.07	0
...	...	...	...	...	...		...	...	...
463	5.25	74.68	117.52	0.12	0.21		0.47	2.44	0
143	42.56	35.49	174.83	5.29	23.35		1.81	1.87	1
1522	15.50	6.56	7.30	0.00	0.12		0.15	0.08	0
14	67.27	-197.77	131.31	8.45	0.21		0.52	0.24	1
3139	17.47	501.78	1390.93	62.52	18.91		0.48	0.62	0

2399 rows × 35 columns

Test

	Equity_Paid_Up	Net_Working_Capital	Curr_Assets	Other_Income	...	Value_of_Output_to_Total_Assets	Value_of_Output_to_Gross_Block	Default
1682	11.75	3.01	3.38	0.62		0.24	0.23	0
3260	25.98	478.87	855.19	28.00		0.68	1.16	0
2156	9.34	7.40	59.65	1.48		1.18	2.39	0
3295	7.35	424.45	1960.66	42.31		1.04	3.63	0
1311	5.05	2.82	4.30	0.23		1.20	1.56	0
...	...	...	...	...		...	...	...
2539	8.90	112.20	130.61	0.30		3.18	4.76	0
451	0.35	0.02	0.06	0.00		0.20	0.00	0
1058	3.69	2.52	2.72	0.23		0.00	0.00	0
2378	5.29	49.48	153.78	3.82		2.71	10.72	0
762	0.27	0.09	0.14	0.00		0.00	0.00	0

1182 rows × 35 columns

Smote train

	Equity_Paid_Up	Net_Working_Capital	Curr_Assets	Other_Income	Selling_Cost	...	Value_of_Output_to_Total_Assets	Value_of_Output_to_Gross_Block	Default
4223	7.347672	7.020814	15.989639	5.816834	0.560601		0.973436	0.745392	1
3493	58.010000	3321.460000	5247.240000	123.720000	376.330000		0.580000	1.040000	0
2844	11.560000	135.650000	267.950000	3.250000	11.670000		2.120000	3.420000	0
4221	8.509298	0.610611	1.269695	10.223334	0.010000		0.359677	0.000000	1
5583	3.960926	-12.222521	1.994483	0.000000	0.001584		0.000000	0.000000	1
...	...	...	...	...	...		...	...	...
2037	5.590000	19.890000	28.630000	2.420000	1.310000		0.950000	2.530000	0
1231	0.500000	5.660000	16.670000	0.020000	1.000000		4.020000	12.010000	0
4528	38.063388	70.952532	109.997387	11.169226	2.001923		0.841460	4.561193	1
3629	3.371029	-0.013994	0.029000	0.084003	0.001999		0.000000	0.000000	1
1428	9.660000	30.590000	30.810000	0.000000	0.010000		0.190000	86.420000	0

Tab 13

Smote test

	Equity_Paid_Up	Net_Working_Capital	Curr_Assets	Other_Income	...	Value_of_Output_to_Total_Assets	Value_of_Output_to_Gross_Block	Default
2835	22.380000	78.410000	103.390000	8.640000		0.780000	1.480000	0
397	3.480000	1.590000	1.590000	0.000000		0.000000	0.000000	0
6079	17.176523	-10.459328	21.625558	6.387596		0.838552	0.367936	1
1238	3.360000	7.660000	8.350000	0.450000		0.650000	0.880000	0
5033	8.221044	-0.404253	0.508410	0.248232		0.000000	0.000000	1
...	...	...	...	...		...	...	...
1262	15.910000	5.100000	15.710000	0.070000		0.240000	0.280000	0
5567	5.449801	0.012240	0.082071	0.000317		0.002537	0.001982	1
705	3.000000	1.770000	1.890000	0.000000		0.050000	0.000000	0
2666	5.240000	53.720000	74.820000	1.640000		0.450000	1.110000	0
5008	5.119697	-3.756400	0.268286	0.000000		0.000000	0.000000	1

2109 rows × 35 columns

Tab 14

Question 1.8

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

Two Random Forest models were created, one with SMOTE data and the other without. The data for Random Forest wasn't scaled as tree models are not distance-based and can handle a wide range of features.

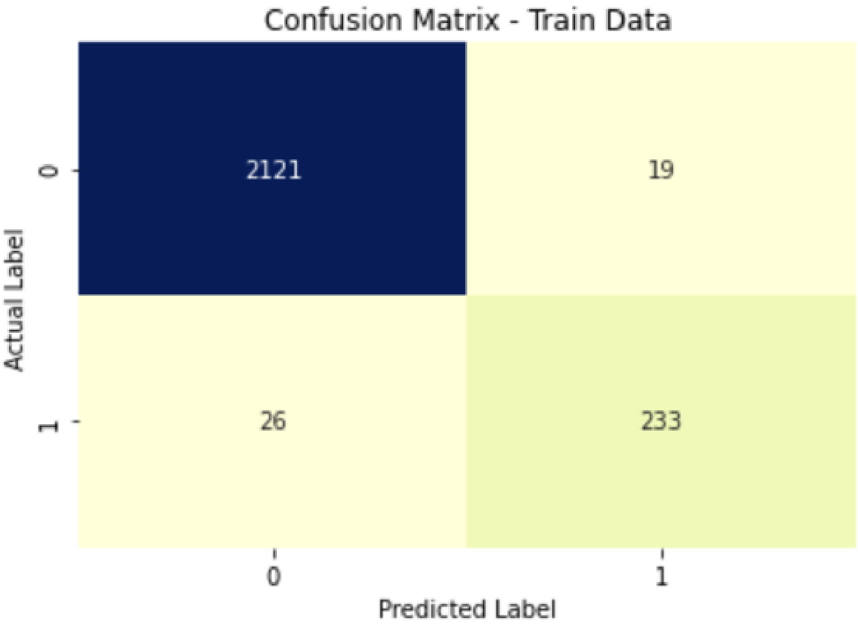
GridSearchCV helped in hyperparameter tuning. For the sake of uniformity, same grid was used for both models (with and without SMOTE). Custom-defined function apply_evl was created to make different type of models and return performance metrics as the output. The same function is used in building and evaluating the various models under the project.

Arguments that the function apply_evl takes: -
1. **Name:** Of the model
2. **Model:** Object created and passed to the function
3. **param_grid:** Grid as a dictionary object that can also be passed as none in case grid search is not needed.
4. **X_Train, X_test,y_train,y_test:** Train-test split of the dataset in a ratio of 67:33 and using random_state = 42.

Random Forest Without SMOTE - Train Data

```
RandomForest
--------------------Best Parameters--------------------
{'max_depth': 10, 'max_features': 7, 'min_samples_leaf': 75, 'min_samples_split': 100, 'n_estimators': 150}


--------------------Best Model Params--------------------
RandomForestClassifier(max_depth=10, max_features=7, min_samples_leaf=75,
                       min_samples_split=100, n_estimators=150)


Train Accuracy Score for model RandomForestClassifier() is 0.9812421842434348
```

```
-------------------Classification Report - Train Data-------------------
              precision    recall  f1-score   support

           0       0.99      0.99      0.99      2140
           1       0.92      0.90      0.91       259

    accuracy                           0.98      2399
   macro avg       0.96      0.95      0.95      2399
weighted avg       0.98      0.98      0.98      2399

-----------------------------------------------------------
```




```
---------------------------------------------------
AUC:  0.993
```

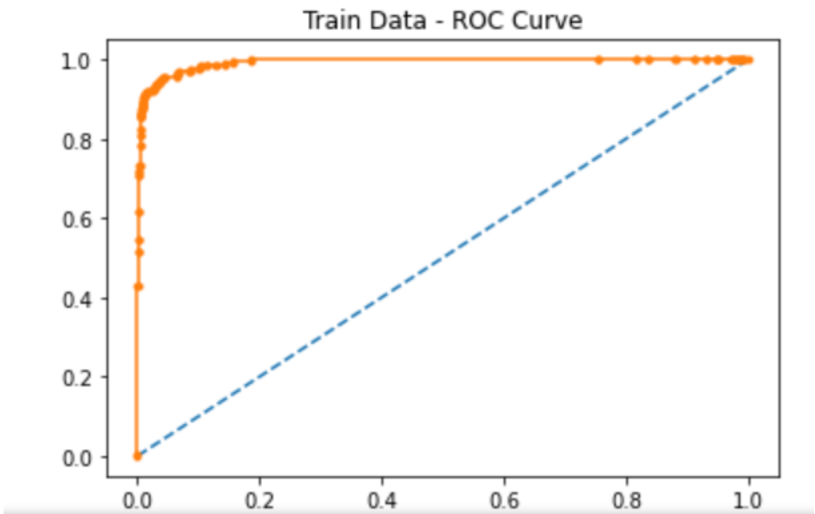
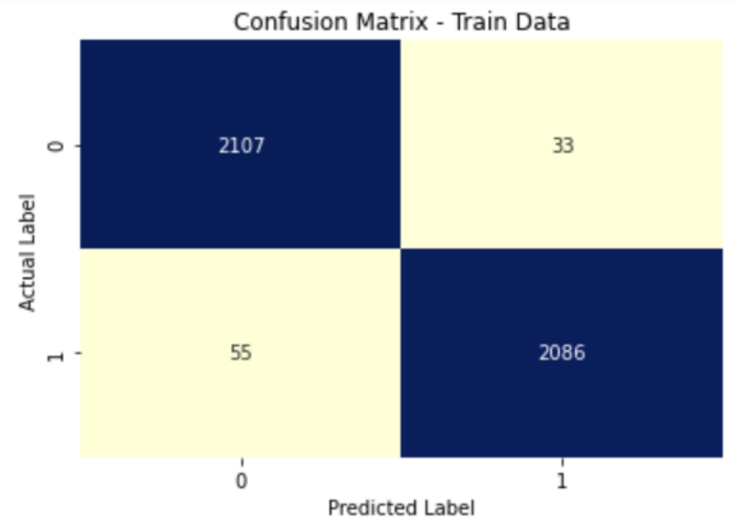


Fig 3

Random Forest With SMOTE – Train Data

```
RF_With_Smote
--------------------Best Parameters-------------------
{'max_depth': 10, 'max_features': 4, 'min_samples_leaf': 25, 'min_samples_split': 25, 'n_estimators': 150}


--------------------Best Model Params-------------------
RandomForestClassifier(max_depth=10, max_features=4, min_samples_leaf=25,
                       min_samples_split=25, n_estimators=150)


Train Accuracy Score for model RandomForestClassifier() is 0.9794440551273067
```

```
        --------------------Classification Report - Train Data--------------------
              precision    recall  f1-score   support

           0       0.97      0.98      0.98      2140
           1       0.98      0.97      0.98      2141

    accuracy                           0.98      4281
   macro avg       0.98      0.98      0.98      4281
weighted avg       0.98      0.98      0.98      4281

        ------------------------------------------------------------
```




```
--------------------------------------------------
AUC: 0.998
```

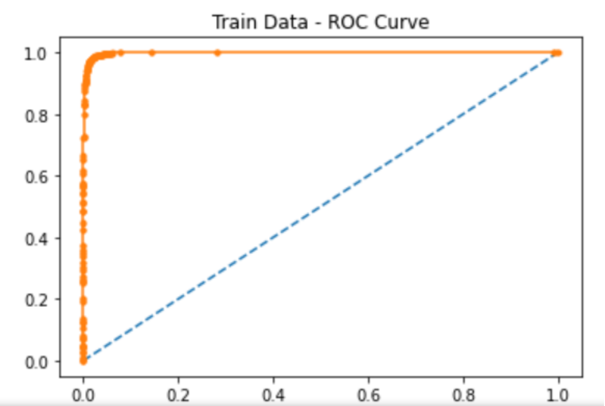


Fig 4

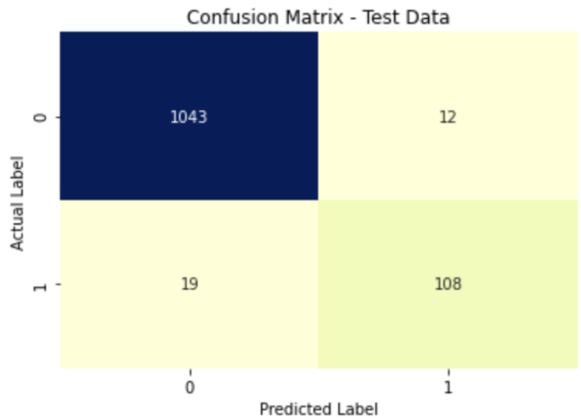
Question 1.9

Validate the Random Forest Model on test dataset and state the performance matrices. Also state interpretation from the model

Both random forest models were evaluated on the test data. Below are those results for models with and without SMOTE data.

Random Forest Without SMOTE - Test Data

```
Test Accuracy Score for model RandomForestClassifier() is 0.9737732656514383


-------------------Classification Report - Test Data-------------------
              precision    recall  f1-score   support

           0       0.98      0.99      0.99      1055
           1       0.90      0.85      0.87       127

    accuracy                           0.97      1182
   macro avg       0.94      0.92      0.93      1182
weighted avg       0.97      0.97      0.97      1182

------------------------------------------------------------
```




```
-----------------------------------------------------------
AUC:  0.985
```

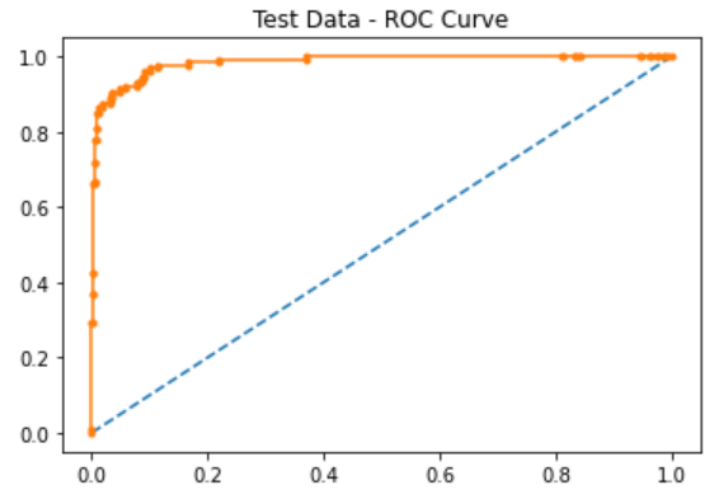
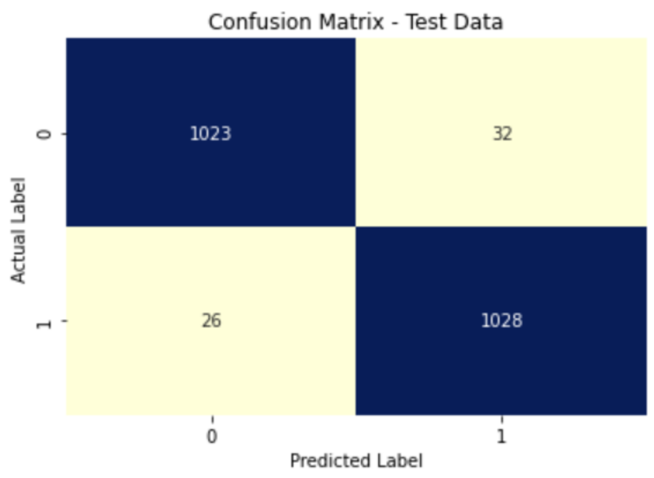


Fig 5

Random Forest With SMOTE - Test Data

```
Test Accuracy Score for model RandomForestClassifier() is 0.9724988146040777


--------------------Classification Report - Test Data--------------------
              precision    recall  f1-score   support

           0       0.98      0.97      0.97      1055
           1       0.97      0.98      0.97      1054

    accuracy                           0.97      2109
   macro avg       0.97      0.97      0.97      2109
weighted avg       0.97      0.97      0.97      2109

------------------------------------------------------------
```




```
------------------------------------------------------------
AUC: 0.996
```

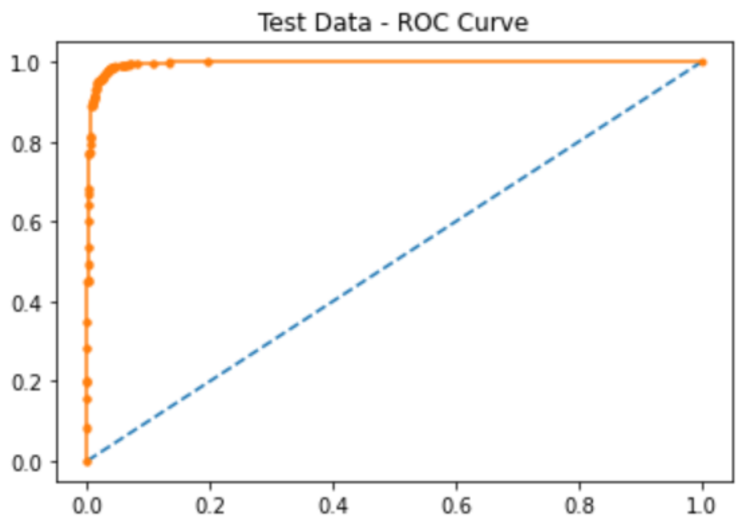


Fig 6

Comparison of various evaluation metrics for both RF models, with and without SMOTE: -

	Accuracy	Precision	Recall	F1	AUC
RandomForest_Train	0.981242	0.924603	0.899614	0.911937	0.992758
RandomForest_Test	0.973773	0.9	0.850394	0.874494	0.98467

	Accuracy	Precision	Recall	F1	AUC
RF_With_Smote_Train	0.979444	0.984427	0.974311	0.979343	0.998394
RF_With_Smote_Test	0.972499	0.969811	0.975332	0.972564	0.996065

Even though Random Forest without SMOTE has higher accuracy, Random Forest with SMOTE surpasses the other model significantly in terms of Precision, Recall, F1 score as well as AUC score.

There is also no significant difference in the accuracies of both models.

Random Forest with SMOTE data is the better of the two models with 97.24% accuracy, 96.98% precision, 97.53% recall, an F1 score of 0.9725, and an AUC score of 0.9960.

Looking at the confusion matrix, the RF model with smote calls default or non-default accurately 97.24% of the times overall but when it calls default, it is right 97.53% of the times, so this prediction is even more reliable.

Question 1.10

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

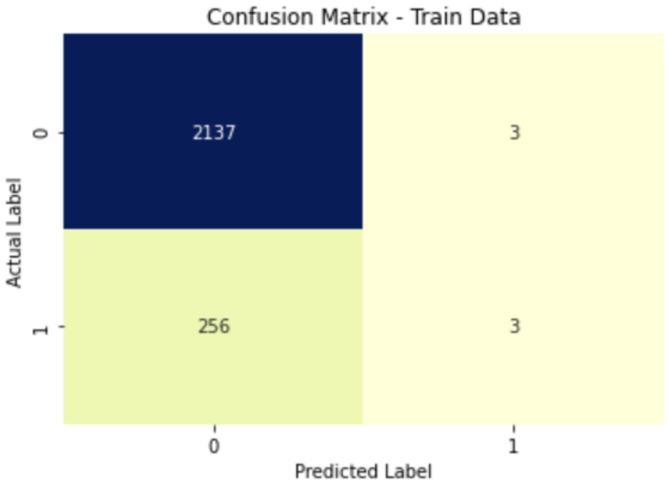
Two Linear Discriminant Analysis (LDA) models were created, one with SMOTE data and the other without. The data for LDA was not scaled, as LDA finds its coefficients using the variation between the classes, which is why scaling doesn't matter.

GridSearchCV helped in hyperparameter tuning. For the sake of uniformity, same grid was used for both models (with and without SMOTE). Custom-defined function apply_evl was created to make different type of models and return performance metrics as the output. The same function is used in building and evaluating the various models under the project.

Arguments that the function apply_evl takes: -
1. **Name:** Of the model
2. **Model:** Object created and passed to the function
3. **param_grid:** Grid as a dictionary object that can also be passed as none in case grid search is not needed.
4. **X_Train, X_test,y_train,y_test:** Train-test split of the dataset in a ratio of 67:33 and using random_state = 42.

LDA Without SMOTE - Train Data

```
LDA
--------------------Best Parameters--------------------
{'shrinkage': 'auto', 'solver': 'lsqr', 'tol': 1e-06}


--------------------Best Model Params--------------------
LinearDiscriminantAnalysis(shrinkage='auto', solver='lsqr', tol=1e-06)


Train Accuracy Score for model LinearDiscriminantAnalysis() is 0.8920383493122134


  --------------------Classification Report - Train Data--------------------
               precision    recall  f1-score   support

           0       0.89      1.00      0.94      2140
           1       0.50      0.01      0.02       259

    accuracy                           0.89      2399
   macro avg       0.70      0.51      0.48      2399
weighted avg       0.85      0.89      0.84      2399

  ------------------------------------------------------------
```




```
--------------------------------------------------
AUC: 0.740
```

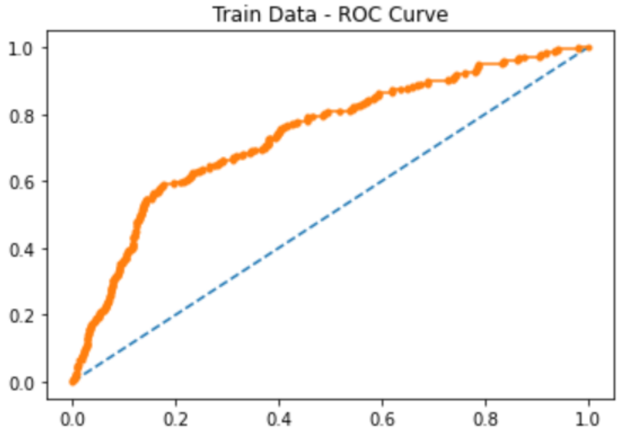
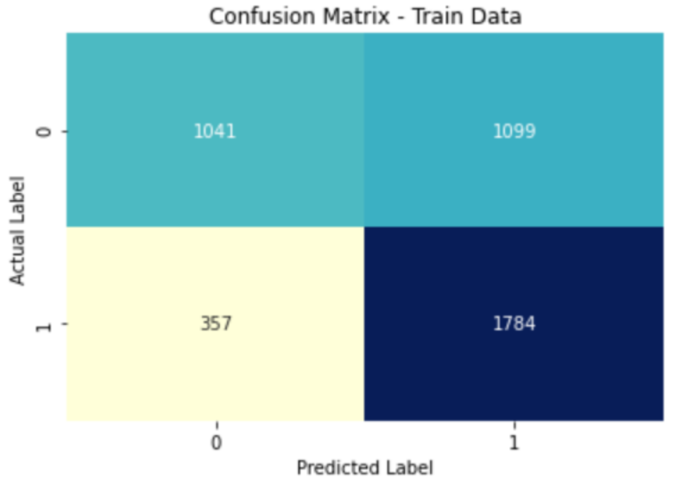


Fig 7

LDA With SMOTE - Train Data

```
LDA_With_Smote
--------------------Best Parameters--------------------
{'shrinkage': 'auto', 'solver': 'lsqr', 'tol': 1e-06}


--------------------Best Model Params--------------------
LinearDiscriminantAnalysis(shrinkage='auto', solver='lsqr', tol=1e-06)


Train Accuracy Score for model LinearDiscriminantAnalysis() is 0.6598925484699837
```

```
--------------------Classification Report - Train Data-------
              precision    recall  f1-score   support

           0       0.74      0.49      0.59      2140
           1       0.62      0.83      0.71      2141

    accuracy                           0.66      4281
   macro avg       0.68      0.66      0.65      4281
weighted avg       0.68      0.66      0.65      4281

------------------------------------------------------------
```




```
------------------------------------------------
AUC: 0.725
```

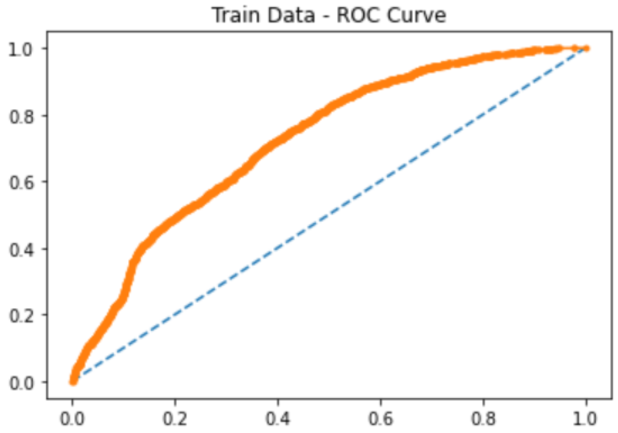


Fig 8

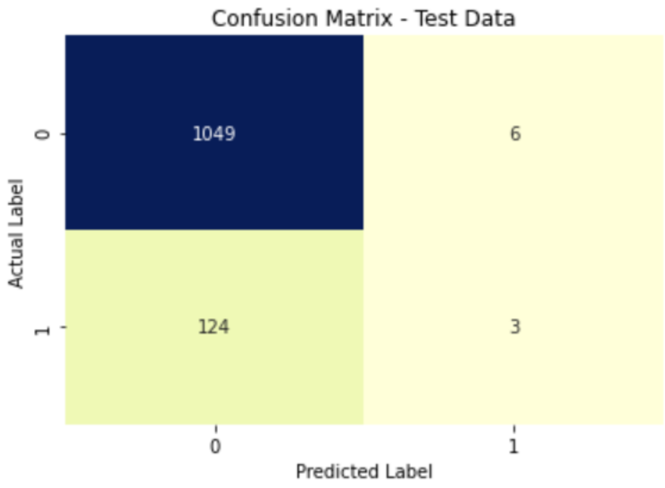
Question 1.11

Validate the LDA model on test dataset and state the performance matrices. Also state interpretation from the model

Both LDA models were evaluated on the test data. Below are those results.

LDA Without SMOTE - Test Data

```
Test Accuracy Score for model LinearDiscriminantAnalysis() is 0.8900169204737732

--------------------Classification Report - Test Data--------------------
              precision    recall  f1-score   support

           0       0.89      0.99      0.94      1055
           1       0.33      0.02      0.04       127

    accuracy                           0.89      1182
   macro avg       0.61      0.51      0.49      1182
weighted avg       0.83      0.89      0.85      1182

------------------------------------------------------------
```




```
--------------------------------------------------
AUC:  0.683
```

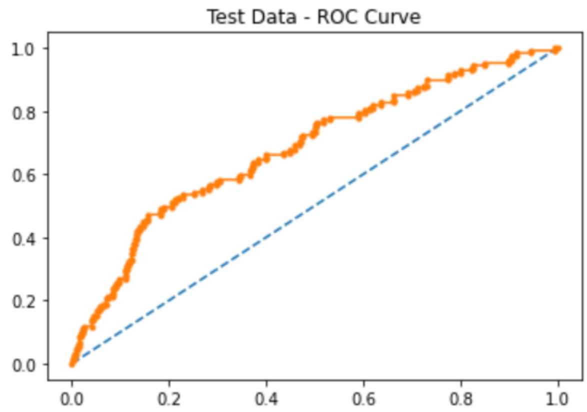
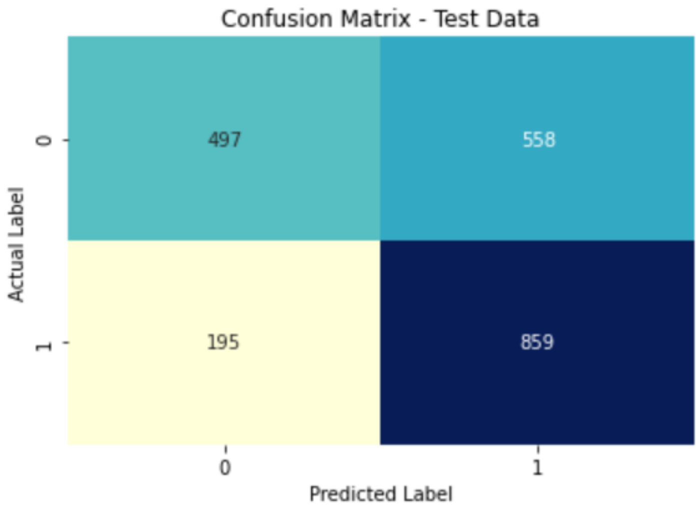


Fig 9

LDA With SMOTE - Test Data

```
Test Accuracy Score for model LinearDiscriminantAnalysis() is 0.6429587482219061


--------------------Classification Report - Test Data--------------------
              precision    recall  f1-score   support

           0       0.72      0.47      0.57      1055
           1       0.61      0.81      0.70      1054

    accuracy                           0.64      2109
   macro avg       0.66      0.64      0.63      2109
weighted avg       0.66      0.64      0.63      2109

------------------------------------------------------------
```




```
----------------------------------------------------
AUC:  0.703
```

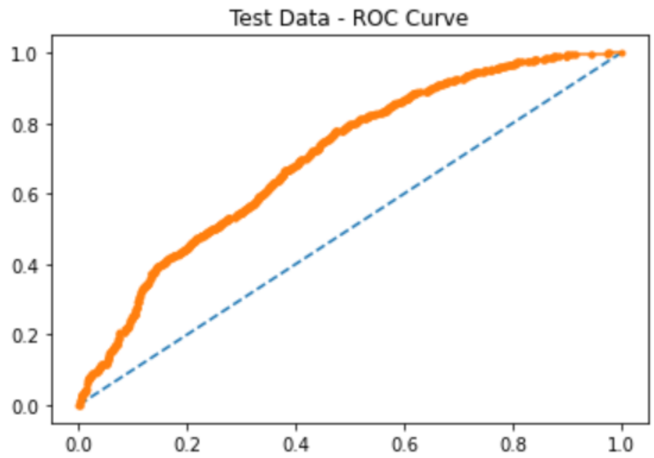


Fig 10

Comparison of various evaluation metrics for both RF models, with and without SMOTE: -

	Accuracy	Precision	Recall	F1	AUC
LDA_Train	0.892038	0.5	0.011583	0.0226415	0.740082
LDA_Test	0.890017	0.333333	0.023622	0.0441176	0.683203

	Accuracy	Precision	Recall	F1	AUC
LDA_With_Smote_Train	0.659893	0.6188	0.833255	0.710191	0.724862
LDA_With_Smote_Test	0.642959	0.60621	0.814991	0.695265	0.702944

The LDA model without SMOTE data has higher accuracy on test data. However, it has really bad precision, recall and F1 scores when compared with the other model. That's where it fails.

Going by just the accuracy perspective, the model without smote data seems to performs better.

However, in the real-world scenario, the model with SMOTE will perform better, given its better recall and precision.

LDA with SMOTE gives a precision of 60.62%, a recall of 81.49%, an F1 score of 0.6952 and an AUC score of 0.7029

Looking at the confusion matrix, the LDA model with SMOTE calls default or non-default accurately just 64.29% of the times overall but when it calls default, it is right 81.49% of the times

Question 1.12

Compare the performances of Logistics, Radom Forest and LDA models (include ROC Curve)

We made following models as part of this project. For the Random Forest and LDA models, scaling was not done, since these models are not impacted by varying magnitudes of the variables.

1) Logistic regression using statsmodels without SMOTE
2) Logistic regression using statsmodels with SMOTE
3) Logistic Regression using sklearn With Unscaled Variables Without SMOTE
4) Logistic Regression using sklearn With Scaled Variables Without SMOTE
5) Logistic Regression using sklearn With Unscaled Variables With SMOTE
6) Logistic Regression using sklearn With Scaled Variables With SMOTE
7) LDA using sklearn without SMOTE
8) LDA using sklearn with SMOTE
9) Random Forest using sklearn without SMOTE
10) Random Forest using sklearn with SMOTE

Logistic Regression for 'probability at default'

The equation of the Logistic Regression by which we predict the corresponding probabilities and then proceed to predict the discrete target variable is given below: -

$$y = \frac{1}{1+e^{-z}}$$

$$z = \beta_0 + \sum_{i=1}^{n} (\beta_i X_1)$$

Fig 11

Statsmodels Logit Modelling with backward elimination: -

After each model is built, the variable that has a p-value of >0.05 will be dropped as their coefficients are unreliable.

Columns used

```
['Equity_Paid_Up',
 'Net_Working_Capital',
 'Curr_Assets',
 'Other_Income',
 'Selling_Cost',
 'Rev_earn_in_forex',
 'Rev_exp_in_forex',
 'Capital_exp_in_forex',
 'Book_Value_Unit_Curr',
 'Book_Value_Adj_Unit_Curr',
 'Market_Capitalisation',
 'CEPS_annualised_Unit_Curr',
 'Cash_Flow_From_Opr',
 'Cash_Flow_From_Inv',
 'ROG_Capital_Employed_perc',
 'ROG_Gross_Block_perc',
 'ROG_Net_Sales_perc',
 'ROG_Cost_of_Prod_perc',
 'ROG_PAT_perc',
 'ROG_Rev_earn_in_forex_perc',
 'ROG_Rev_exp_in_forex_perc',
 'ROG_Market_Capitalisation_perc',
 'Curr_Ratio_Latest',
 'Inventory_Ratio_Latest',
 'Debtors_Ratio_Latest',
 'Total_Asset_Turnover_Ratio_Latest'
 'Interest_Cover_Ratio_Latest',
 'PBIDTM_perc_Latest',
 'CPM_perc_Latest',
 'Debtors_Vel_Days',
 'Creditors_Vel_Days',
 'Inventory_Vel_Days',
 'Value_of_Output_to_Total_Assets',
 'Value_of_Output_to_Gross_Block']
```

Comparison chart for all the models, sorted by accuracy

	Accuracy	Precision	Recall	F1	AUC
RandomForest_Train	0.981242	0.924603	0.899614	0.911937	0.992758
RF_With_Smote_Train	0.979444	0.984427	0.974311	0.979343	0.998394
RandomForest_Test	0.973773	0.9	0.850394	0.874494	0.98467
RF_With_Smote_Test	0.972499	0.969811	0.975332	0.972564	0.996065
LogisticRegression_Unscaled_Train	0.971238	0.927928	0.795367	0.856549	0.966781
LogisticRegression_Scaled_Train	0.962484	0.942408	0.694981	0.8	0.976515
LogisticRegression_Unscaled_Test	0.961929	0.87963	0.748031	0.808511	0.944762
Logit_SM_Train	0.958316	0.760656	0.895753	0.822695	0.975275
Logit_SM_Test	0.952623	0.726115	0.897638	0.802817	0.956256
Logit_SM_Train_SMOTE	0.950245	0.979125	0.920131	0.948712	0.981177
LogisticRegression_Scaled_Test	0.950085	0.84	0.661417	0.740088	0.944233
Logit_SM_Test_SMOTE	0.945472	0.962562	0.926945	0.944418	0.974799
LogisticRegression_With_Smote_Train	0.9395	0.965381	0.911723	0.937785	0.972872
LogisticRegression_Scaled_With_Smote_Train	0.936463	0.917001	0.959832	0.937928	0.982399
LogisticRegression_With_Smote_Test	0.93504	0.943853	0.925047	0.934356	0.968673
LogisticRegression_Scaled_With_Smote_Test	0.927454	0.900444	0.961101	0.929784	0.975394
LDA_Train	0.892038	0.5	0.011583	0.0226415	0.740082
LDA_Test	0.890017	0.333333	0.023622	0.0441176	0.683203
LDA_With_Smote_Train	0.659893	0.6188	0.833255	0.710191	0.724862
LDA_With_Smote_Test	0.642959	0.60621	0.814991	0.695265	0.702944

Comparison chart for all the models, sorted by AUC score

	Accuracy	Precision	Recall	F1	AUC
RF_With_Smote_Train	0.979444	0.984427	0.974311	0.979343	0.998394
RF_With_Smote_Test	0.972499	0.969811	0.975332	0.972564	0.996065
RandomForest_Train	0.981242	0.924603	0.899614	0.911937	0.992758
RandomForest_Test	0.973773	0.9	0.850394	0.874494	0.98467
LogisticRegression_Scaled_With_Smote_Train	0.936463	0.917001	0.959832	0.937928	0.982399
Logit_SM_Train_SMOTE	0.950245	0.979125	0.920131	0.948712	0.981177
LogisticRegression_Scaled_Train	0.962484	0.942408	0.694981	0.8	0.976515
LogisticRegression_Scaled_With_Smote_Test	0.927454	0.900444	0.961101	0.929784	0.975394
Logit_SM_Train	0.958316	0.760656	0.895753	0.822695	0.975275
Logit_SM_Test_SMOTE	0.945472	0.962562	0.926945	0.944418	0.974799
LogisticRegression_With_Smote_Train	0.9395	0.965381	0.911723	0.937785	0.972872
LogisticRegression_With_Smote_Test	0.93504	0.943853	0.925047	0.934356	0.968673
LogisticRegression_Unscaled_Train	0.971238	0.927928	0.795367	0.856549	0.966781
Logit_SM_Test	0.952623	0.726115	0.897638	0.802817	0.956256
LogisticRegression_Unscaled_Test	0.961929	0.87963	0.748031	0.808511	0.944762
LogisticRegression_Scaled_Test	0.950085	0.84	0.661417	0.740088	0.944233
LDA_Train	0.892038	0.5	0.011583	0.0226415	0.740082
LDA_With_Smote_Train	0.659893	0.6188	0.833255	0.710191	0.724862
LDA_With_Smote_Test	0.642959	0.60621	0.814991	0.695265	0.702944
LDA_Test	0.890017	0.333333	0.023622	0.0441176	0.683203

Tab 15

Best Model on Train Data

	Accuracy	Precision	Recall	F1	AUC
RandomForest_Train	0.981242	0.924603	0.899614	0.911937	0.992758
RF_With_Smote_Train	0.979444	0.984427	0.974311	0.979343	0.998394
LogisticRegression_Unscaled_Train	0.971238	0.927928	0.795367	0.856549	0.966781
LogisticRegression_Scaled_Train	0.962484	0.942408	0.694981	0.8	0.976515
Logit_SM_Train	0.958316	0.760656	0.895753	0.822695	0.975275
Logit_SM_Train_SMOTE	0.950245	0.979125	0.920131	0.948712	0.981177
LogisticRegression_With_Smote_Train	0.9395	0.965381	0.911723	0.937785	0.972872
LogisticRegression_Scaled_With_Smote_Train	0.936463	0.917001	0.959832	0.937928	0.982399
LDA_Train	0.892038	0.5	0.011583	0.0226415	0.740082
LDA_With_Smote_Train	0.659893	0.6188	0.833255	0.710191	0.724862

Best Model on Test Data

	Accuracy	Precision	Recall	F1	AUC
RandomForest_Test	0.973773	0.9	0.850394	0.874494	0.98467
RF_With_Smote_Test	0.972499	0.969811	0.975332	0.972564	0.996065
LogisticRegression_Unscaled_Test	0.961929	0.87963	0.748031	0.808511	0.944762
Logit_SM_Test	0.952623	0.726115	0.897638	0.802817	0.956256
LogisticRegression_Scaled_Test	0.950085	0.84	0.661417	0.740088	0.944233
Logit_SM_Test_SMOTE	0.945472	0.962562	0.926945	0.944418	0.974799
LogisticRegression_With_Smote_Test	0.93504	0.943853	0.925047	0.934356	0.968673
LogisticRegression_Scaled_With_Smote_Test	0.927454	0.900444	0.961101	0.929784	0.975394
LDA_Test	0.890017	0.333333	0.023622	0.0441176	0.683203
LDA_With_Smote_Test	0.642959	0.60621	0.814991	0.695265	0.702944

Tab 16

Random Forest model has surpassed all other models in terms of accuracy. Both Random Forest with and without smote feature in the top 2 models.

While RF was the best performing model, LDA models were the worst. LDA with or without SMOTE was bad in terms of accuracy, LDA with SMOTE being the worst

Comparison of different evaluation metrics for all the models on the training data set:-

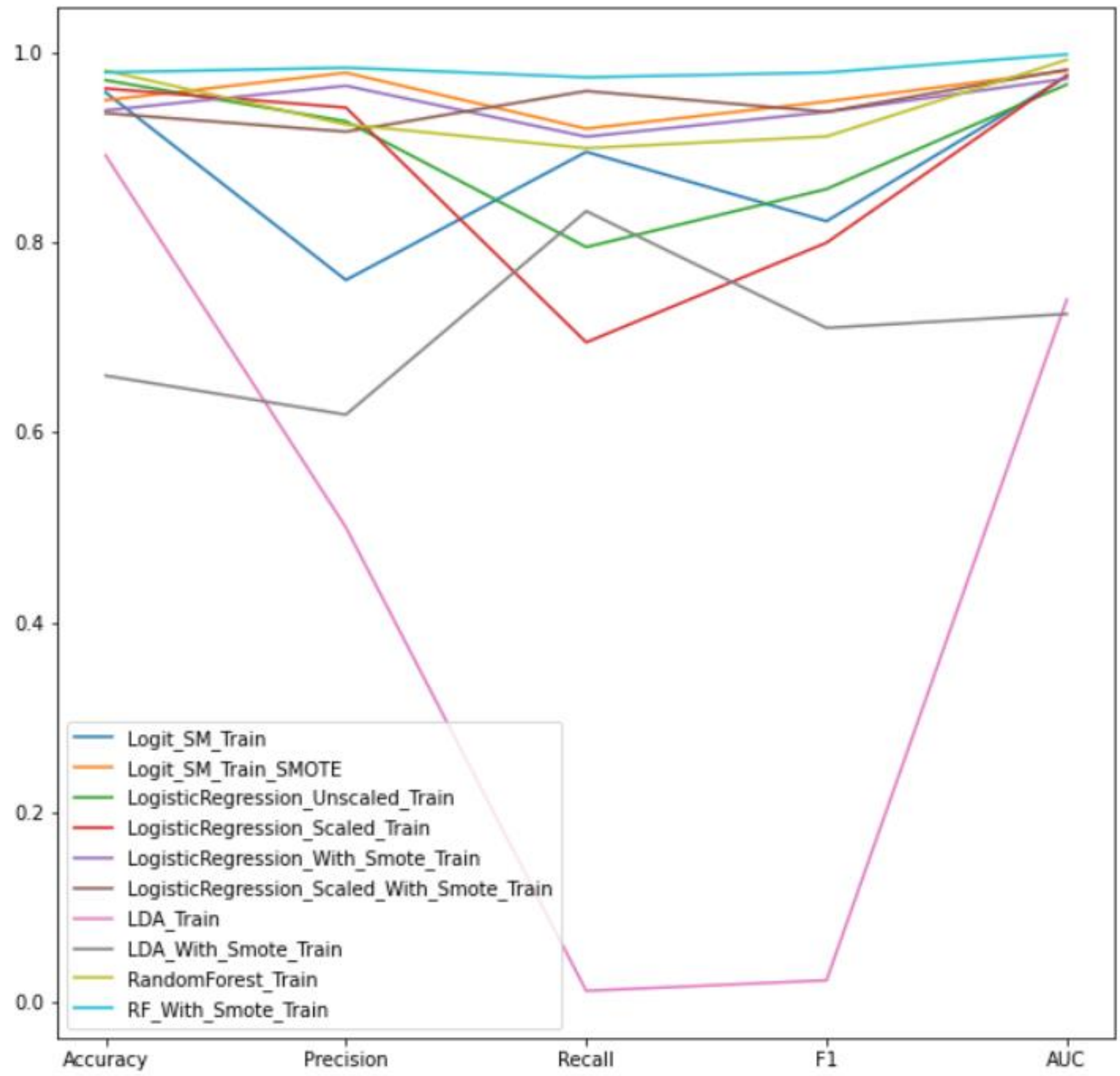


Fig 11.1

From the chart, it is clear that the blue line which indicates 'Random Forest with SMOTE' is on top of all the other models, and consistently for all the evaluation parameters. While the pink line for LDA without SMOTE is slightly higher than LDA with SMOTE in terms of accuracy and AUC score, it slips down a great deal on all other parameters, making it the worst performing model.

Comparison of different evaluation metrics for all the models on the test data set:-

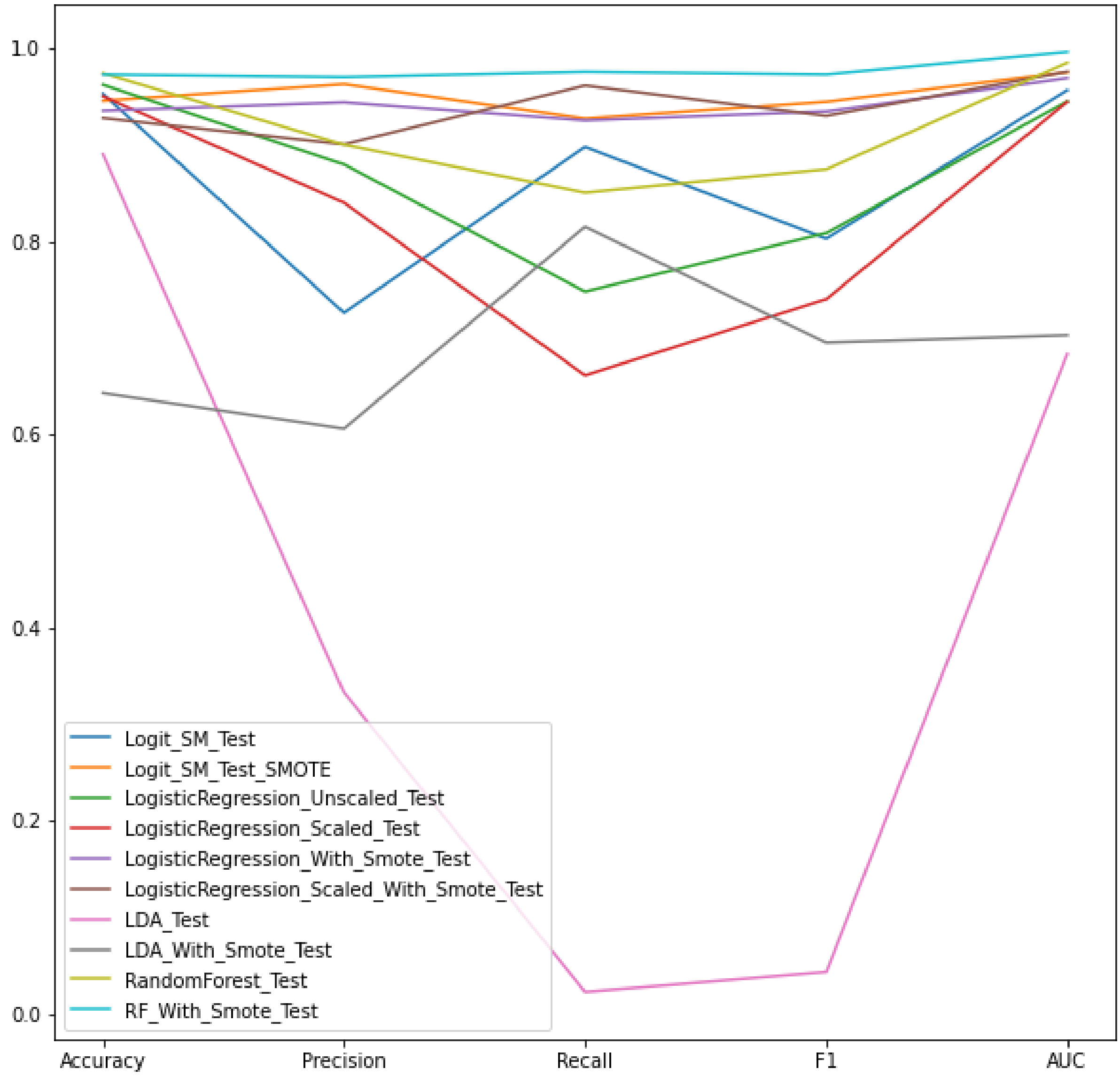


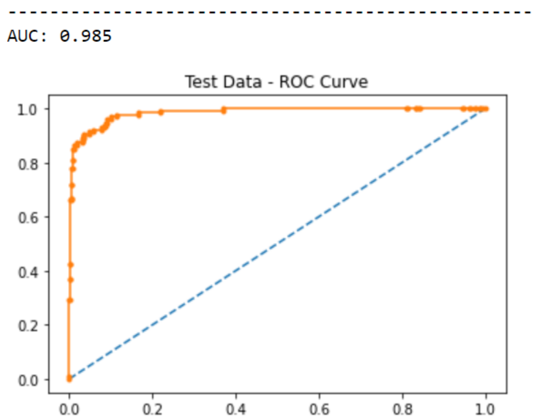
Fig 12

Test data is the deciding factor. There, we see Random Forest with SMOTE indicated by blue line performing the best on almost all the parameters. The blue line dip only very slightly on accuracy compared with Random Forest without SMOTE, but on all other fronts, the blue line stays on the top and is the clear winner among all the various models.

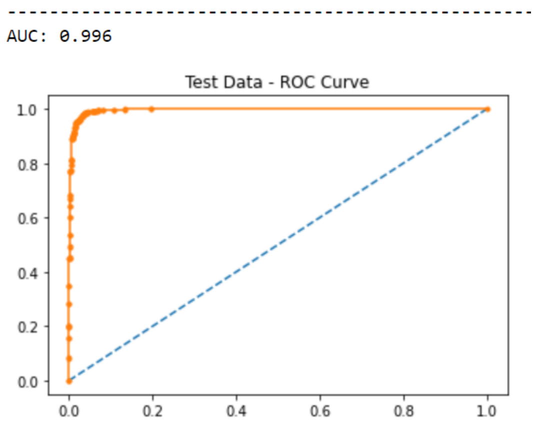
Random Forest model with SMOTE is the best model of all, with 97.24% accuracy, 96.98% precision, 97.53% recall, an F1 score of 0.9725, and an AUC score of 0.9960.

ROC curves

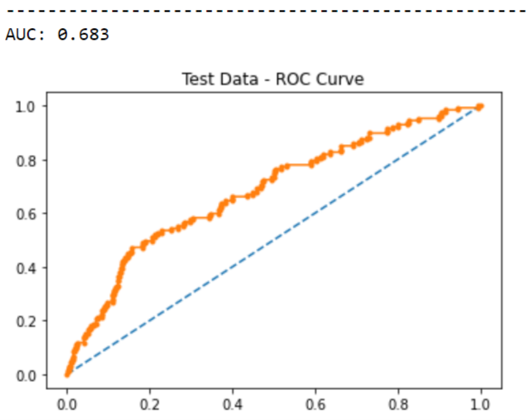
ROC curves for Logit, LDA and random forest models have been plotted below. The ROC curves for the best performing models on the test dataset have also been plotted for comparison. Comparing the AUC scores, we see random forest with SMOTE to be clear winner among all the models.



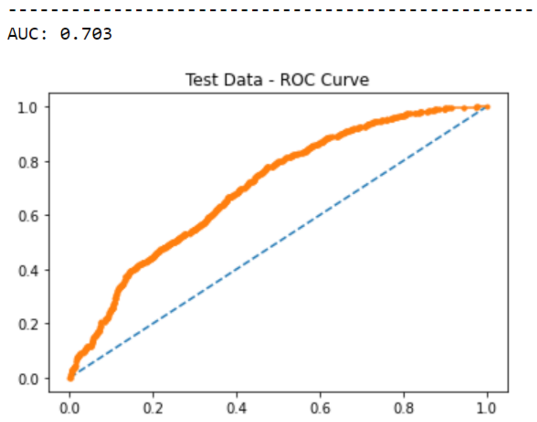
RF without SMOTE



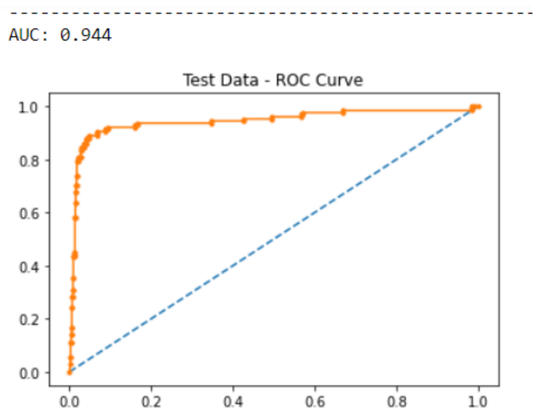
RF with SMOTE



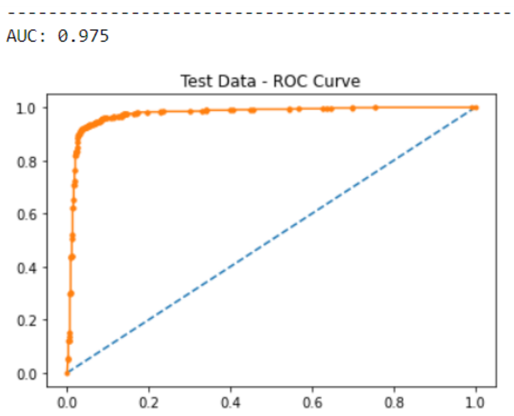
LDA without SMOTE



LDA with SMOTE



Logit without SMOTE (scaled)



Logit with SMOTE (scaled)

Fig 13

Question 1.13

State Recommendations from the above models

Random Forest with SMOTE is found to be the best model. The coefficients derived from the best logistic regression model built using Statsmodels library also help us derive some useful insights.

	coef	std err	z	P>|z|	[0.025	0.975]
Intercept	-0.3898	0.148	-2.641	0.008	-0.679	-0.100
Book_Value_Unit_Curr	-0.1514	0.012	-12.174	0.000	-0.176	-0.127
CEPS_annualised_Unit_Curr	-0.0972	0.013	-7.649	0.000	-0.122	-0.072
Curr_Ratio_Latest	-0.4580	0.097	-4.733	0.000	-0.648	-0.268
Interest_Cover_Ratio_Latest	-0.0024	0.001	-2.999	0.003	-0.004	-0.001

Tab 17

Best model summary shows us that the following factors should be kept in mind while investing in the given set of companies.

1) Lower the Book_value_unit_curr i.e. Net assets, higher is the chance of a default, which means the net worth next year for this company is expected to be negative.

2) Lower the CEPS_annualised_Unit_Curr i.e. Cash earning per share, higher is the change of a default.

3) Higher the Curr_Ratio_Latest, i.e. the companies' ability to pay short-term dues, lower are their chances of defaulting or having a negative net worth in the next year.

4) Higher the Interest_Cover_Ratio_Latest, lower the chances of default. Which means easier the company is able to pay the interest on its outstanding debt, lower are its chances to default.

Curr_Ratio_Latest is most important criteria amongst the above parameters, while Interest_Cover_Ratio_Latest is the least important when considering only these 4 parameters. However, all these 4 parameters are more important than the rest of the variables.

Problem 2: Executive summary

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

Usage

Market risk analysis

Format

A dataframe with 314 rows and 11 columns

Introduction

Stock price evolution is very difficult to forecast but the spread between two correlated stocks is easier to predict. The spread can also be traded separately. If two stocks are highly correlated, it doesn't mean that they will remain highly correlated in the long term. Their correlation may weaken in the long term.
Source: GL literature

Dataset

Market Risk Dataset: Market+Risk+Dataset.csv

Source

Great Learning

Dataset

	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

Dataset after fixing messy names

	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

Tab 18

The dataset has a date variable, followed by stock prices of 10 Indian companies

Data types

```
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
```

Descriptive stats

	Infosys	Indian_Hotel	Mahindra_&_Mahindra	Axis_Bank	SAIL	Shree_Cement	Sun_Pharma	Jindal_Steel	Idea_Vodafone	Jet_Airways
count	314.000000	314.000000	314.000000	314.000000	314.000000	314.000000	314.000000	314.000000	314.000000	314.000000
mean	511.340764	114.560510	636.678344	540.742038	59.095541	14806.410828	633.468153	147.627389	53.713376	372.659236
std	135.952051	22.509732	102.879975	115.835569	15.810493	4288.275085	171.855893	65.879195	31.248985	202.262668
min	234.000000	64.000000	284.000000	263.000000	21.000000	5543.000000	338.000000	53.000000	3.000000	14.000000
25%	424.000000	96.000000	572.000000	470.500000	47.000000	10952.250000	478.500000	88.250000	25.250000	243.250000
50%	466.500000	115.000000	625.000000	528.000000	57.000000	16018.500000	614.000000	142.500000	53.000000	376.000000
75%	630.750000	134.000000	678.000000	605.250000	71.750000	17773.250000	785.000000	182.750000	82.000000	534.000000
max	810.000000	157.000000	956.000000	808.000000	104.000000	24806.000000	1089.000000	338.000000	117.000000	871.000000

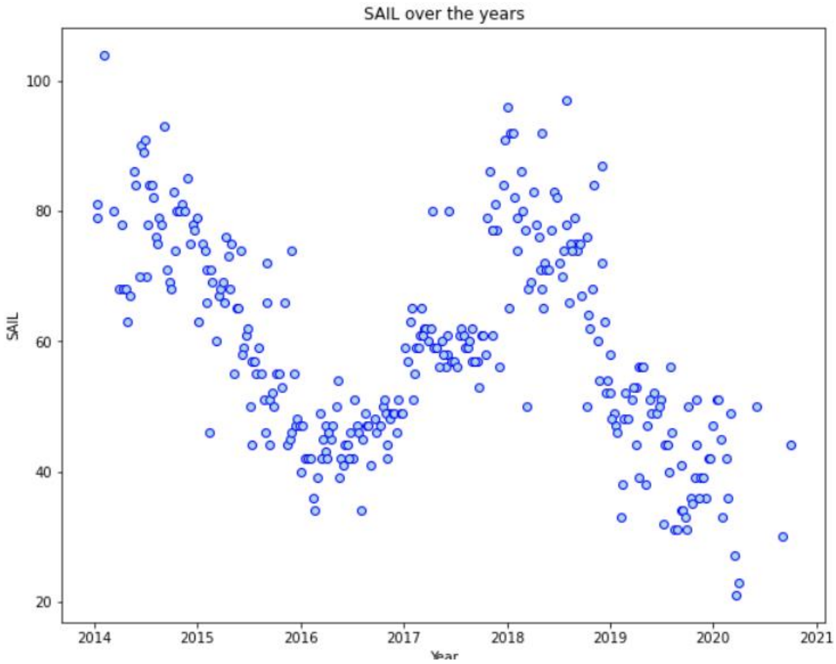
When it comes to stocks, we are more interested in returns than prices

Tab 19

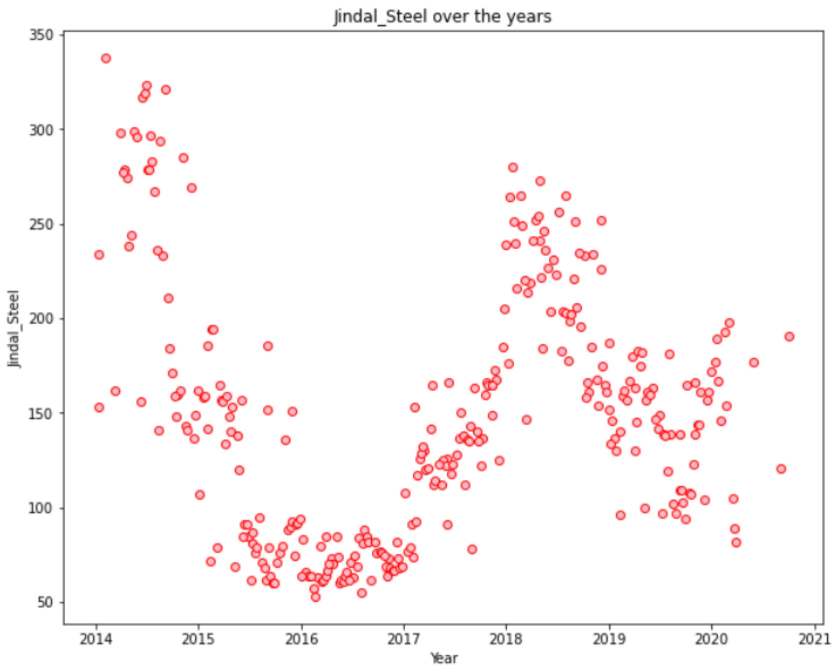
Question 2.1

Draw stock price graph (stock price vs time) for any 2 given stocks with inference

We pick up two steel-sector companies: SAIL or Steel Authority of India Limited and Jindal Steel



SAIL



Jindal Steel

Fig 14

A similar story in there. Both stocks have a downward movement overall, with a period of fall in 2016, followed by recovery in 2018. Then they went down again and are trying to bounce back.

Question 2.2

Calculate the returns for all stocks with inference

Analyzing returns

Steps for calculating returns from prices:

- Take logarithms
- Take differences

Shape of returns dataset

```
(314, 10)
```

Top 5 rows for week-over-week returns for all the stocks are given below: -

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

Tab 20

Inference

The first row contains NaNs, because this observation didn’t have a previous value, to be converted into a return. Thereafter, every value has been converted into a proper logarithmic return. Log is the difference between the two consecutive days’ prices.

Stock Return r_t at time t is measured as $r_t = \ln(P_t/P_{t-1})$

Where P_t & P_{t-1} are the stock prices at times t and t-1 respectively.

Question 2.3

Calculate stock means and standard deviation for all stocks with inference

Stock means

Average returns that the stock is making on a week-to-week basis

```
Shree_Cement            0.003681
Infosys                 0.002794
Axis_Bank               0.001167
Indian_Hotel            0.000266
Sun_Pharma             -0.001455
Mahindra_&_Mahindra    -0.001506
SAIL                   -0.003463
Jindal_Steel           -0.004123
Jet_Airways            -0.009548
Idea_Vodafone          -0.010608
dtype: float64
```

Tab 21

Idea Vodafone has the lowest return, while Shree Cements has the highest.

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

```
Idea_Vodafone           0.104315
Jet_Airways             0.097972
Jindal_Steel            0.075108
SAIL                    0.062188
Indian_Hotel            0.047131
Axis_Bank               0.045828
Sun_Pharma              0.045033
Mahindra_&_Mahindra     0.040169
Shree_Cement            0.039917
Infosys                 0.035070
dtype: float64
```

Tab 22

Idea Vodafone has the highest risk factor, while Infosys is the least risky investment option.

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

Tab 23

Question 2.4

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

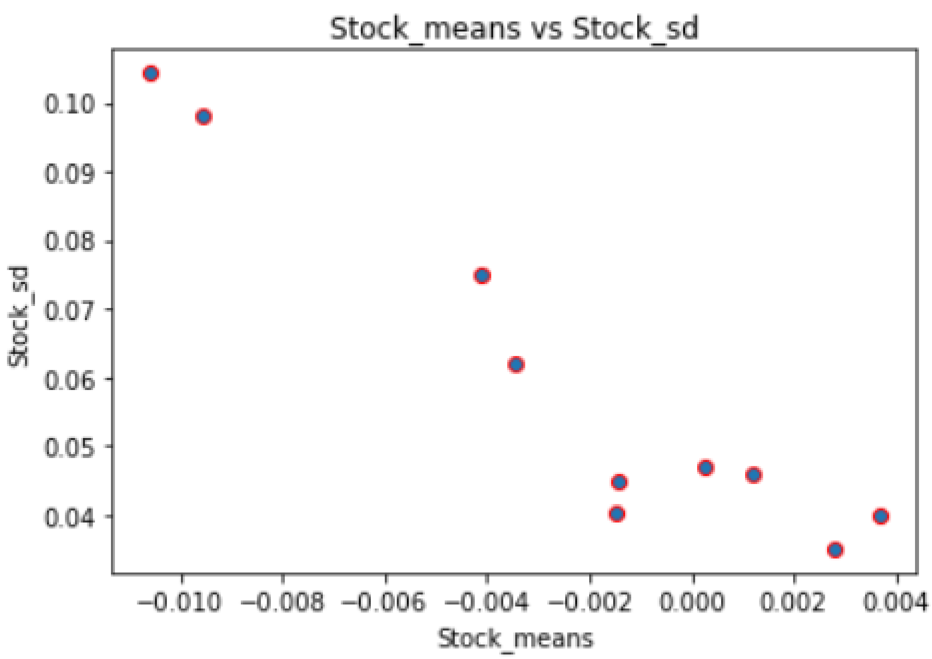


Fig 15

The stocks higher up on the far left have high volatility and low returns.

The stocks on the bottom right have low volatility and high returns.

This graph can be used to find a balance between risk and reward while investing in these different companies.

Question 2.5

Conclusion and Recommendations

1. Stocks with a lower mean and a higher standard deviation don't play a role in a portfolio that has competing stock with higher returns and lower risk.

2. For the data we have, we have only two kinds of stock, the first with highest returns and lowest risk, while the second with lowest risk and highest return.

3. From the point of view of returns, Shree Cement followed by Infosys and Axis Bank look good in this dataset.

4. From the risk perspective as measured by standard deviation, Infosys followed by Shree Cement and Mahindra&Mahindra look good in the dataset.

5. We recommend using the stock means versus the standard deviation plot to asses the risk to reward ratio. More volatile stock might give short-term gains but might not be a good investment in the long run.

6. A low volatile stock might not be a good investment in the short term, but might give a good return in the long run.

7. Based on the type of investment one desires, an investment decision should be made by looking at the above-mentioned plot.

Thank you