FINANCE AND RISK ANALYTICS

PROJECT REPORT

BY,

RAGAVEDHNI K R

1. STATEMENT

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. Find the Defaulters using the various attributes given in the data.

EXPLORATORY DATA ANALYSIS

- The data is read from 'Company_Data2015-1.xlsx' dataset, initial set of rows can be viewed as below.

	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]	Debtors Velocity (Days)
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	0
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	29
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	97
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	93
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	3887

- The data has 67 columns and 3586 rows.

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

- Checking the information on the data.

```
<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_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)
```

- The Descriptive Statistics is shown using the 5 point summary.
- The 5 point summary has Count, Mean, Std, Min, 25%, 50%, 75%, Max values calculated for all the columns.

	Co_Code	Networth_Next_Year	Equity_Paid_Up	Networth	Capital_Employed	Total_Debt	Gross_Block	Net_Working_Capital	Curr_Assets
count	3586.00	3586.00	3586.00	3586.00	3586.00	3586.00	3586.00	3586.00	3586.00
mean	16065.39	725.05	62.97	649.75	2799.61	1994.82	594.18	410.81	1960.35
std	19776.82	4769.68	778.76	4091.99	26975.14	23652.84	4871.55	6301.22	22577.57
min	4.00	-8021.60	0.00	-7027.48	-1824.75	-0.72	-41.19	-13162.42	-0.91
25%	3029.25	3.98	3.75	3.89	7.60	0.03	0.57	0.94	4.00
50%	6077.50	19.02	8.29	18.58	39.09	7.49	15.87	10.14	24.54
75%	24269.50	123.80	19.52	117.30	226.61	72.35	131.90	61.17	135.28
max	72493.00	111729.10	42263.46	81657.35	714001.25	652823.81	128477.59	223257.56	721166.00

- The data has 118 null values, i.e., most of the null values are from Inventory Velocity (Days) variable. We can either ignore them or use any missing value treatment.
- There are no duplicate values are preset in the data.

TRAIN TEST SPLIT

- We split the data into Train and Test dataset in a ratio of 67:33 and we use the random_state as 42.
- The Model Building is to be done on Train Dataset and Model Validation is to be done on Test Dataset.
- Checking the shape of the target column for the train and test split data respectively.

```
(2402, 65)
(1184, 65)
```

MODEL BUILDING – RANDOM FOREST MODEL

- Using the RandomForestClassifier module from the sklearn.Ensemble package, we generate the Random Forest Model.
- The parameters used in the model to help prune the forest and avoid overfitting of the model.
- The various parameters are n_estimators, max_features, oob_score.
- The max_features is used to decide on considering the number of features for the best split is used for the tree splitting.

```
RandomForestClassifier(bootstrap=True, ccp_alpha=0.0, class_weight=None,
                       criterion='gini', max_depth=None, max_features=10,
                       max_leaf_nodes=None, max_samples=None,
                       min_impurity_decrease=0.0, min_impurity_split=None,
                       min_samples_leaf=1, min_samples_split=2,
                       min_weight_fraction_leaf=0.0, n_estimators=301,
                       n_jobs=None, oob_score=True, random_state=1, verbose=0,
                       warm_start=False)
```

- The oob_score parameter is used to decide whether to use out-of-bag samples to estimate the generalization accuracy, pass it as true for the model to predict Out Of Bag (OOB) score.
- To know the Score of the training dataset obtained using an out-of-bag estimate.

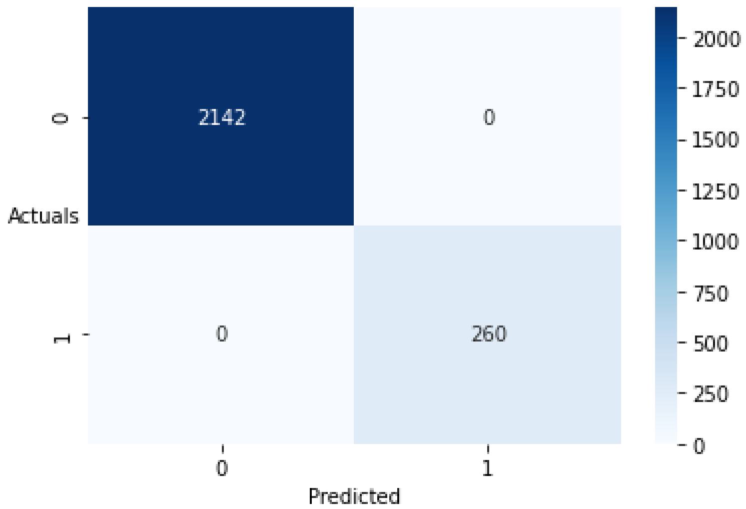
```
rf.oob_score_

0.97751873438801
```

- The RF model object can be used on train and test data to predict its performance.

<u>Validation on the Train data:</u>

- By default, we classify the predicted target values using 0.5 threshold.
- We get the confusion matrix and classification report as below.
- The Confusion matrix is created using the confusion_matrix module from sklearn.metrics package.



- The Classification Report is created using the classification_report module from sklearn.metrics package.

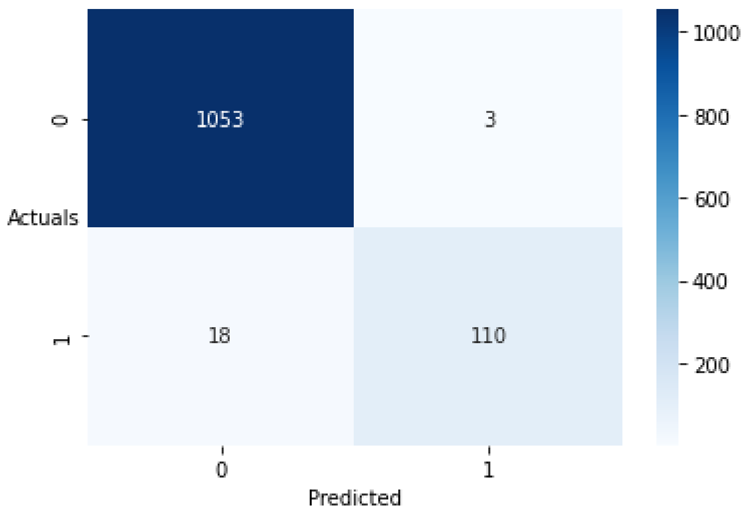
```
              precision    recall  f1-score   support

           0      1.000     1.000     1.000      2142
           1      1.000     1.000     1.000       260

    accuracy                          1.000      2402
   macro avg      1.000     1.000     1.000      2402
weighted avg      1.000     1.000     1.000      2402
```

<u>Validation on the Train data:</u>

- By default, we classify the predicted target values using 0.5 threshold.
- We get the confusion matrix and classification report for the test data as below.



```
              precision    recall  f1-score   support

           0      0.983     0.997     0.990      1056
           1      0.973     0.859     0.913       128

    accuracy                          0.982      1184
   macro avg      0.978     0.928     0.951      1184
weighted avg      0.982     0.982     0.982      1184
```

- The Random Forest model seems to be overfitting, hence we build Random Forest using Grid Search CV.

GRIDSEARCHCV WITH RANDOM FOREST:

- To get the best parameters for the model, use hyper parameters in the GridSearchCV module from sklearn. model_selection package.
- The various parameters are max_depth, max_features, min_samples_leaf, min_samples_split, n_estimators that are given in param_grid.
- The random forest model is fitted to the GridSearchCV with the param_grid and cross validation.

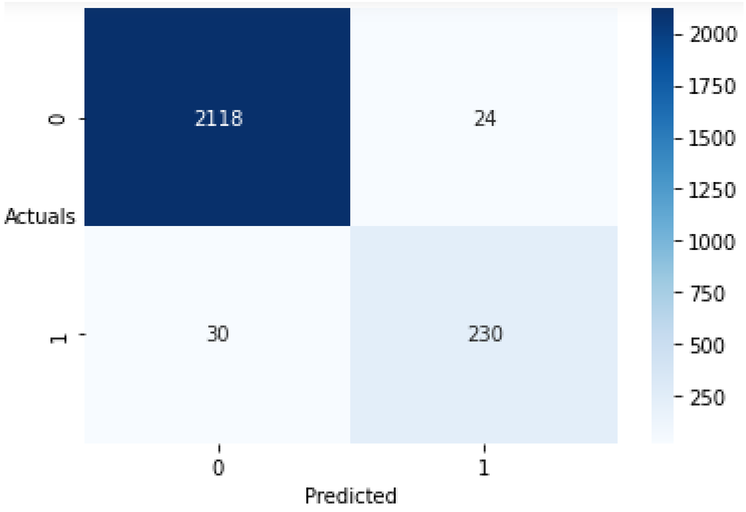
```
GridSearchCV(cv=None, error_score=nan,
             estimator=RandomForestClassifier(bootstrap=True, ccp_alpha=0.0,
                                              class_weight=None,
                                              criterion='gini', max_depth=None,
                                              max_features='auto',
                                              max_leaf_nodes=None,
                                              max_samples=None,
                                              min_impurity_decrease=0.0,
                                              min_impurity_split=None,
                                              min_samples_leaf=1,
                                              min_samples_split=2,
                                              min_weight_fraction_leaf=0.0,
                                              n_estimators=100, n_jobs=None,
                                              oob_score=False,
                                              random_state=None, verbose=0,
                                              warm_start=False),
             iid='deprecated', n_jobs=None,
             param_grid={'max_depth': [8, 9, 10], 'max_features': [10, 13, 16],
                         'min_samples_leaf': [18, 20],
                         'min_samples_split': [45, 50], 'n_estimators': [101]},
             pre_dispatch='2*n_jobs', refit=True, return_train_score=False,
             scoring=None, verbose=0)
```

- To know the best selected parameters by the model, using best_params_ attribute.

```
{'max_depth': 10,
 'max_features': 10,
 'min_samples_leaf': 20,
 'min_samples_split': 45,
 'n_estimators': 101}
```

- Assigning the best estimator attribute to the model and using them for predicting the train and test data.

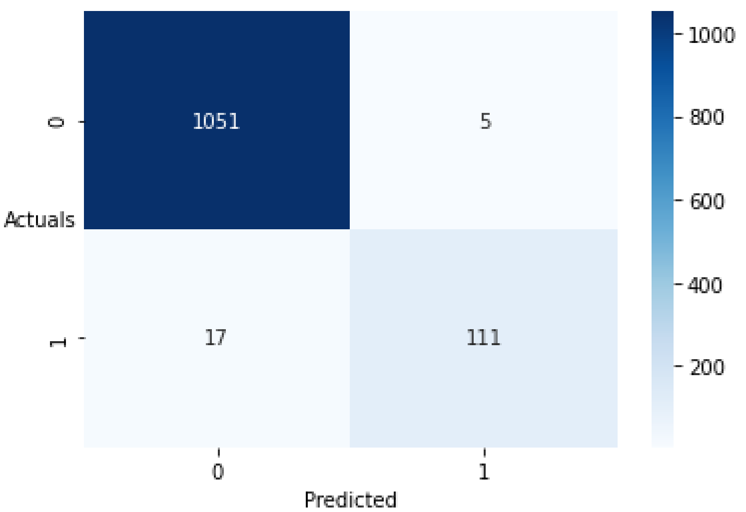
CONFUSION MATRIX OF RF ON TRAIN DATA:



CLASSIFICATION REPORT OF RF ON TRAIN DATA:

	precision	recall	f1-score	support
0	0.986	0.989	0.987	2142
1	0.906	0.885	0.895	260
accuracy			0.978	2402
macro avg	0.946	0.937	0.941	2402
weighted avg	0.977	0.978	0.977	2402

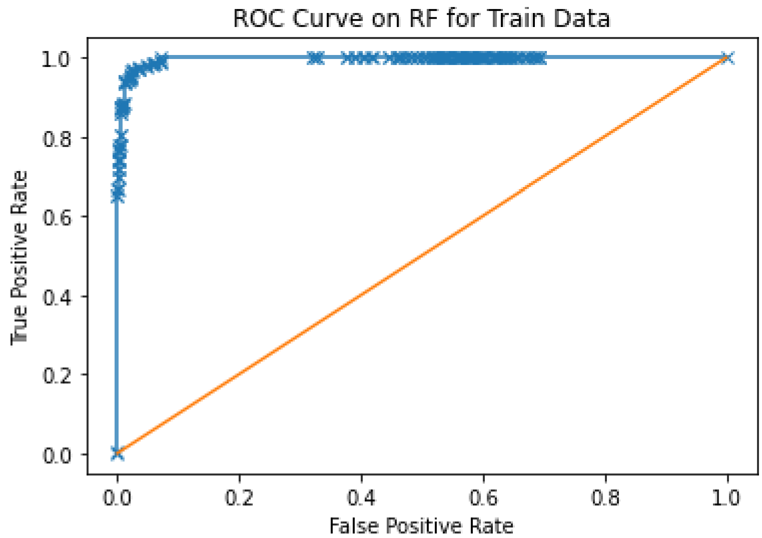
CONFUSION MATRIX OF RF ON TEST DATA:



CLASSIFICATION REPORT OF RF ON TEST DATA:

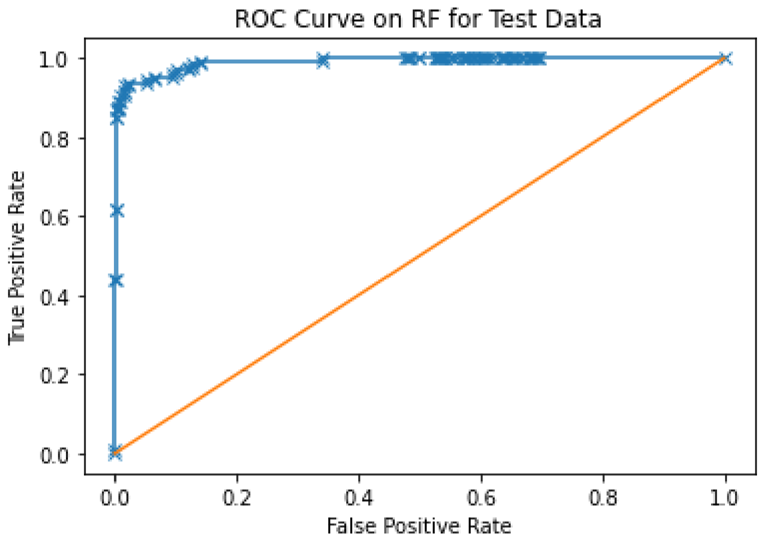
	precision	recall	f1-score	support
0	0.984	0.995	0.990	1056
1	0.957	0.867	0.910	128
accuracy			0.981	1184
macro avg	0.970	0.931	0.950	1184
weighted avg	0.981	0.981	0.981	1184

ROC-CURVE AND AUC-SCORE OF RF ON TRAIN DATA:



AUC score on RF for Train Data: 0.9962166918049271

ROC-CURVE AND AUC-SCORE OF RF ON TEST DATA:



AUC score on RF for Test Data: 0.9903823390151516

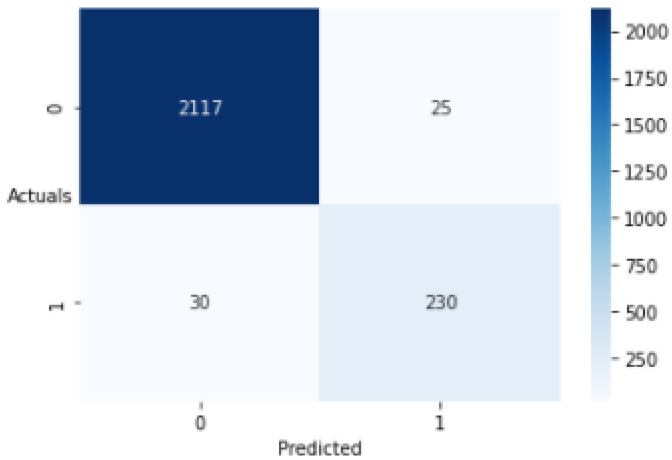
INTERPRETATIONS FROM THE RANDOM FOREST MODEL:

- The GridSearchCV on the Random Forest model works well on the test data than on train data.
- The Recall, Accuracy, Precision, F1 Score seems high in test data than in train data.
- We fetch the important features (values above 0.000) from this and build Random Forest model (using GridSearchCV) using those features.

	Imp
Book_Value_Adj_Unit_Curr	0.29
Book_Value_Unit_Curr	0.25
Networth	0.21
Curr_Ratio_Latest	0.05
Capital_Employed	0.04
PBDT	0.02
PBIT	0.02
CP	0.02
PBIDT	0.02
CEPS_annualised_Unit_Curr	0.01
ROG_Net_Worth_perc	0.01
Net_Working_Capital	0.01
PAT	0.01
PBIDTM_perc_Latest	0.01
PBT	0.01
Total_Debt	0.01

- Checking on the Confusion matrix and Classification report from the optimized Random Forest model using GridSearchCV.

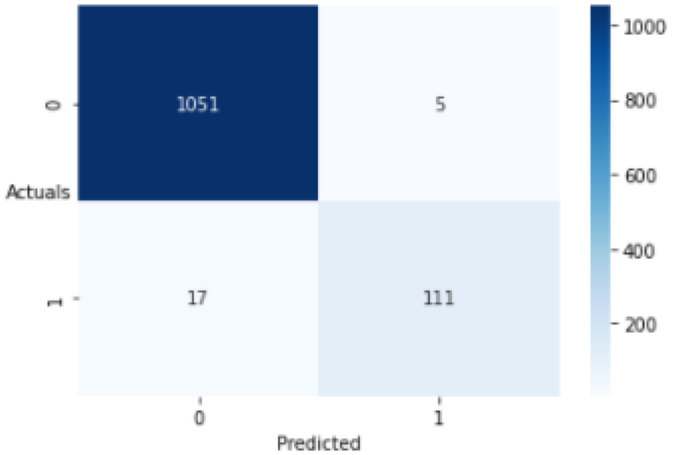
CONFUSION MATRIX OF RF ON TRAIN DATA:



CLASSIFICATION REPORT OF RF ON TRAIN DATA:

	precision	recall	f1-score	support
0	0.986	0.988	0.987	2142
1	0.902	0.885	0.893	260
accuracy			0.977	2402
macro avg	0.944	0.936	0.940	2402
weighted avg	0.977	0.977	0.977	2402

CONFUSION MATRIX OF RF ON TEST DATA:



CLASSIFICATION REPORT OF RF ON TEST DATA:

	precision	recall	f1-score	support
0	0.984	0.995	0.990	1056
1	0.957	0.867	0.910	128
accuracy			0.981	1184
macro avg	0.970	0.931	0.950	1184
weighted avg	0.981	0.981	0.981	1184

- We can see that the metric values remain nearly the same in both models (model built using all features and important features) across the train and test data.
- The important features from the second model can be seen as below.

	Imp
Book_Value_Adj_Unit_Curr	0.38
Networth	0.36
Book_Value_Unit_Curr	0.22
Curr_Ratio_Latest	0.01
CEPS_annualised_Unit_Curr	0.01
ROG_Net_Worth_perc	0.01
PBDT	0.01
PBIT	0.00
CPM_perc_Latest	0.00
PBIDT	0.00
Capital_Employed	0.00
CP	0.00
PBT	0.00
Total_Asset_Turnover_Ratio_Latest	0.00
Adjusted_PAT	0.00
PAT	0.00
Net_Working_Capital	0.00

MODEL BUILDING - LINEAR DISCRIMINANT ANALYSIS:

- The Linear Discriminant Analysis model is a classifier with a linear decision boundary, generated by fitting class conditional densities to the data and using Bayes' rule.

```
LinearDiscriminantAnalysis(
    *,
    solver='svd',
    shrinkage=None,
    priors=None,
    n_components=None,
    store_covariance=False,
    tol=0.0001,
)
```

- The parameters with their default values are:
  - Solver: svd – Singular Value Decomposition, lsqr – Least Square solution, eigen –Eigen Value decomposition.
  - Shrinkage: none – no shrinkage, auto – automatic shrinkage, give float between 0 and 1.
  - Priors: class prior probabilities as an array.
  - N_components: number of components for dimensionality reduction.
  - Store_covariance:
  - Tol: absolute threshold.

- We apply the Linear Discriminant Analysis model on the data with default value of parameters.

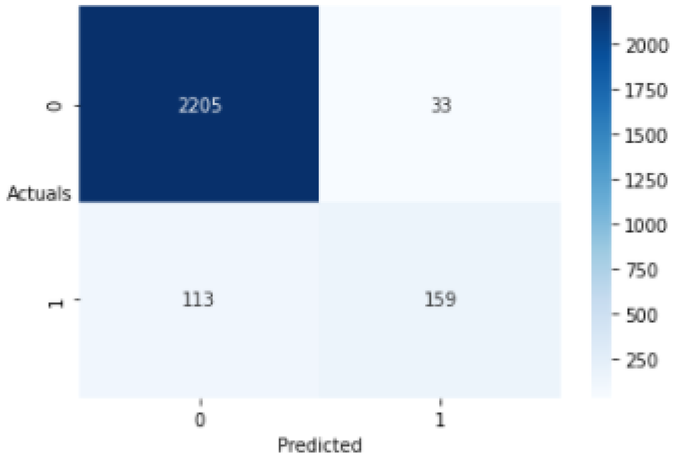
```
LinearDiscriminantAnalysis(n_components=None, priors=None, shrinkage=None,
                           solver='svd', store_covariance=False, tol=0.0001)
```

- We can predict the model by using predict function on train or test data.

Validation on the Train data:

- By default, we classify the predicted target values using 0.5 threshold.
- We get the confusion matrix and classification report as below.

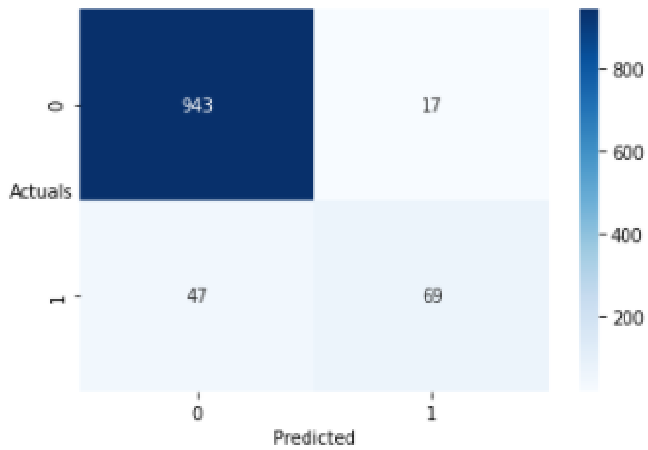
CONFUSION MATRIX OF LDA ON TRAIN DATA:



CLASSIFICATION REPORT OF LDA ON TRAIN DATA:

	precision	recall	f1-score	support
0	0.951	0.985	0.968	2238
1	0.828	0.585	0.685	272
accuracy			0.942	2510
macro avg	0.890	0.785	0.827	2510
weighted avg	0.938	0.942	0.937	2510

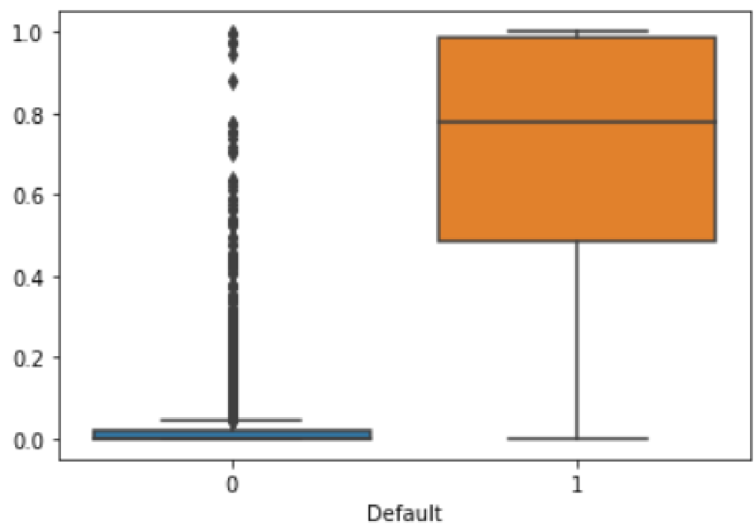
CONFUSION MATRIX OF LDA ON TEST DATA:



CLASSIFICATION REPORT OF LDA ON TEST DATA:

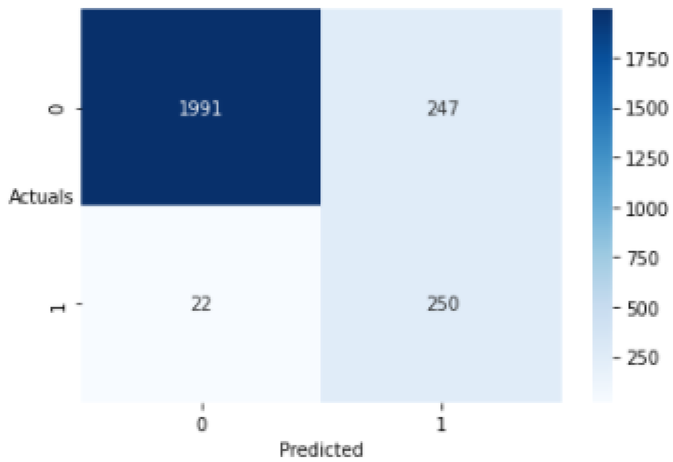
	precision	recall	f1-score	support
0	0.953	0.982	0.967	960
1	0.802	0.595	0.683	116
accuracy			0.941	1076
macro avg	0.877	0.789	0.825	1076
weighted avg	0.936	0.941	0.937	1076

- The Recall value seems low in both train and test data. We change the threshold and check the metrics again.
- The target column is unbalanced, so we use the optimum cut-off to classify the target variables as binary.



- The optimum cut-off is selected using the parameters fpr,tpr,threshold returned from the ROC-Curve.
- Using the obtained optimum threshold, we classify the predicted target column as 0 and 1.
- The rows which are predicted with target value below the threshold (< 0.18) are classified as 0 and above the threshold (> 0.18) are classified as 1.
- We can check using the optimum threshold (0.06) if the Recall value increases.

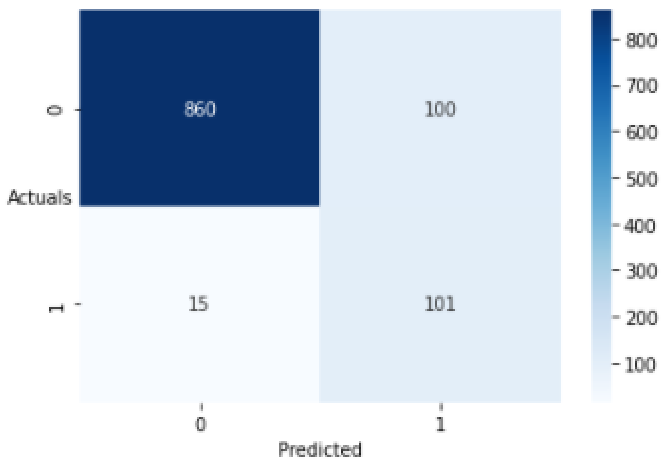
CONFUSION MATRIX OF LDA ON TRAIN DATA:



CLASSIFICATION REPORT OF LDA ON TRAIN DATA:

	precision	recall	f1-score	support
0	0.989	0.890	0.937	2238
1	0.503	0.919	0.650	272
accuracy			0.893	2510
macro avg	0.746	0.904	0.793	2510
weighted avg	0.936	0.893	0.906	2510

CONFUSION MATRIX OF LDA ON TEST DATA:

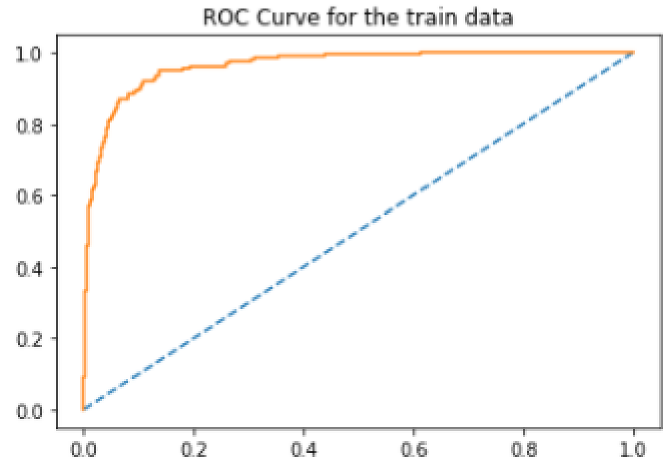


CLASSIFICATION REPORT OF LDA ON TEST DATA:

	precision	recall	f1-score	support
0	0.983	0.896	0.937	960
1	0.502	0.871	0.637	116
accuracy			0.893	1076
macro avg	0.743	0.883	0.787	1076
weighted avg	0.931	0.893	0.905	1076

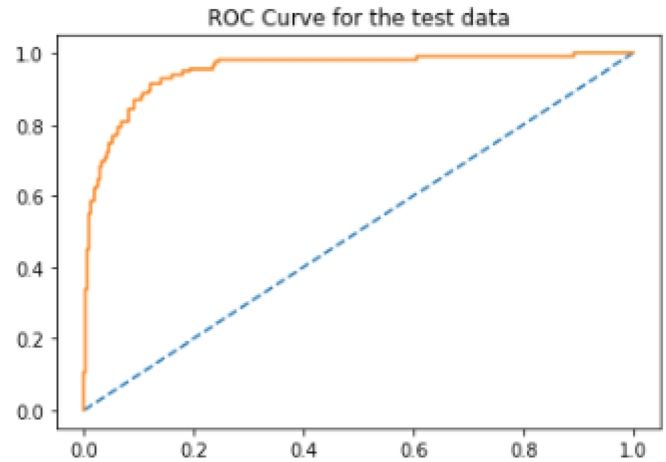
ROC-CURVE AND AUC-SCORE OF LDA ON TRAIN DATA:

AUC of the Linear Discriminant Analysis for Train data: 0.963



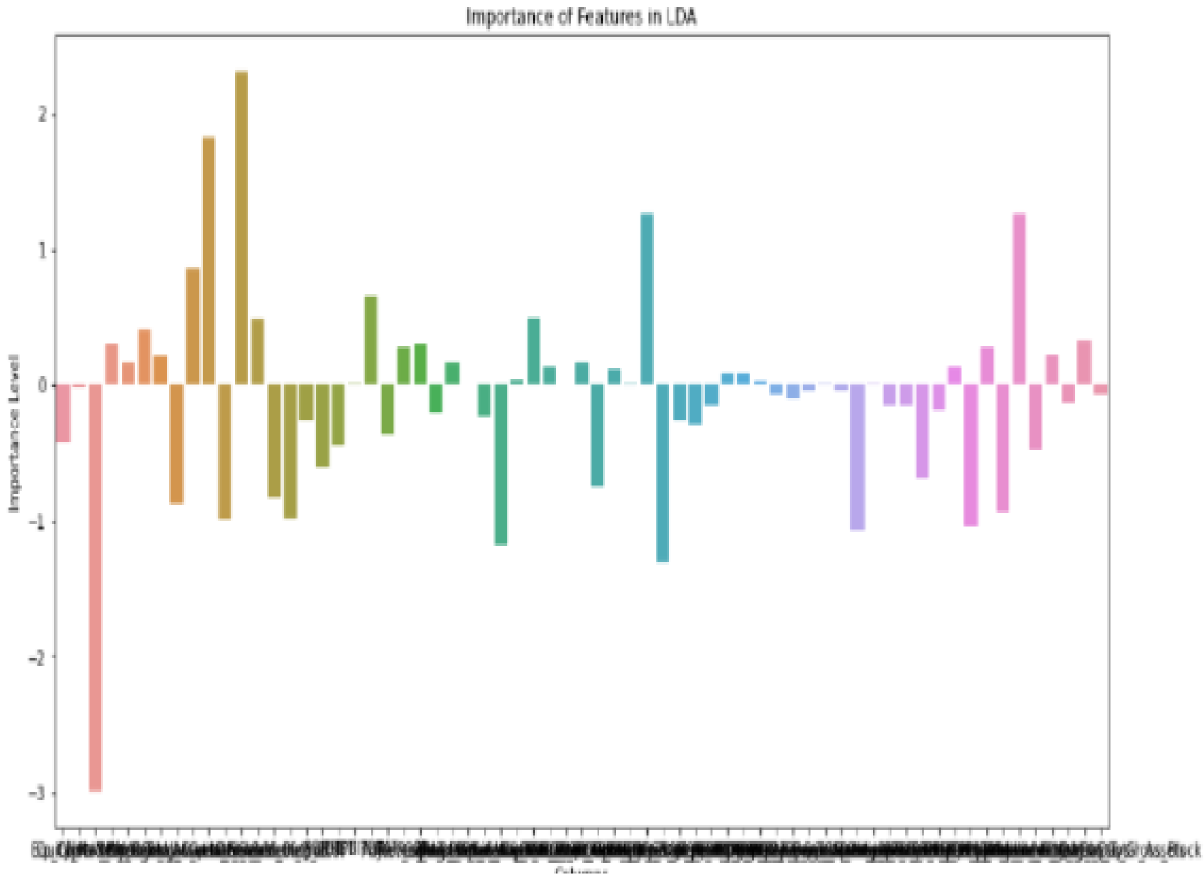
ROC-CURVE AND AUC-SCORE OF LDA ON TEST DATA:

AUC of the Linear Discriminant Analysis for Test data: 0.953



INTERPRETATION FROM LDA MODEL:

- The model works better with the Optimum threshold than the default threshold.
- The Recall is high in both the train and test data, while classifying the predicted target using optimum threshold.
- Checking on the important features from the LDA model and we build the LDA model again only using the important features.

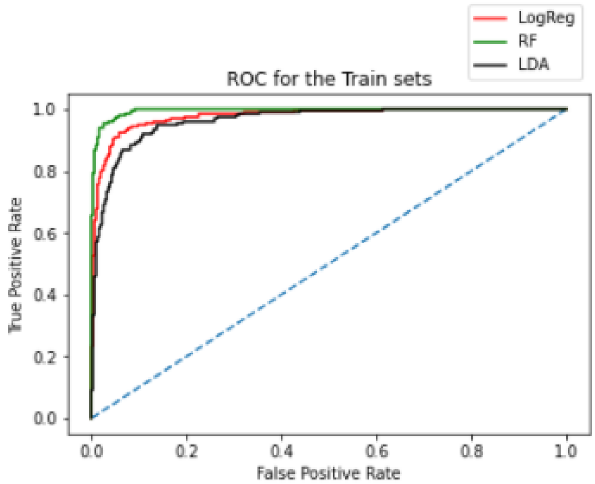


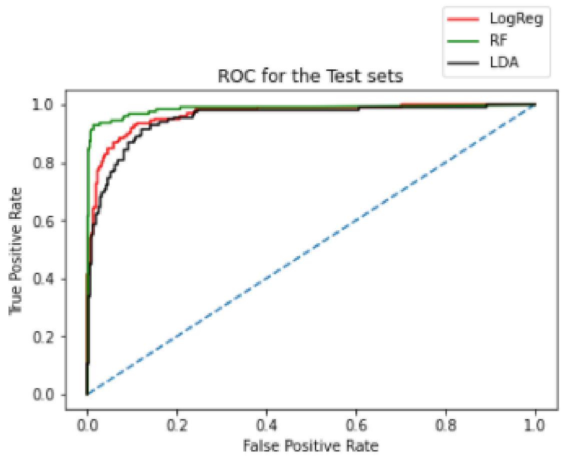
Compare the performances of Logistics, Radom Forest and LDA models

- By combining all the performance metrics from Logistic Regression, RF and LDA models into a dataframe, we get the below table.

	Logit Train	Logit Test	RF Train	RF Test	LDA Train	LDA Test
Accuracy	0.94	0.93	0.98	0.91	0.89	0.89
AUC Score	0.98	0.97	1.00	0.99	0.96	0.95
Recall	0.92	0.89	0.88	0.87	0.92	0.87
Precision	0.66	0.61	0.90	0.96	0.50	0.50
F1 Score	0.76	0.72	0.89	0.91	0.65	0.64

- The Random Forest model has high accuracy than Logistic Regression and LDA models, but RF seems to be slightly overfitting.
- Based on the accuracy, the Logit model performs well for both the train and test data.
- The AUC score is high in Random Forest model.
- The Logit model has high Recall, but Precision wise it does not perform much.
- The Random Forest model has high Precision than other models.
- The F1 Score is high in Random Forest model.
- The ROC curve is better for the Random Forest in both train and test data.





BUSINESS INSIGHTS

- Our primary concern is the Recall in this problem.
- Hence, the Logistic Regression model is selected as the best model having **91% recall on train and 89% recall on test.**
- This is a very good model and further enhancement is possible provided on the tuning and having hyper parameters to improve the precision of the approach.
- On the outset, it is good we shall be able to cover more organisations which are brink of a defaulting as the precision is low and pinpointed.
- This shall help in identifying at very early stages of attention to some false positive companies as they could enhance their accelerated growth and profitability once under the magnification glass.

RECOMMENDATIONS

- With the recall being high it also gives credentials to the model as the number of true negatives are being so less.
- Once identifying the defaults, we can work on them to increase the Network of Next year.
- Find the factors that lead to the negative Network of the company.
- Check the possible ways to balance the factors and make the Network high for next year.

2. STATEMENT

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.

EXPLORATORY DATA ANALYSIS

- Reading the data from the csv file and checking on the head of the data.

	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

- Renaming the features, as to work efficiently with the data.

	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

- Checking on the shape of the data.

```
The number of rows (observations) is 314
The number of columns (variables) is 11
```

- Checking the information on the data.

```
<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
```

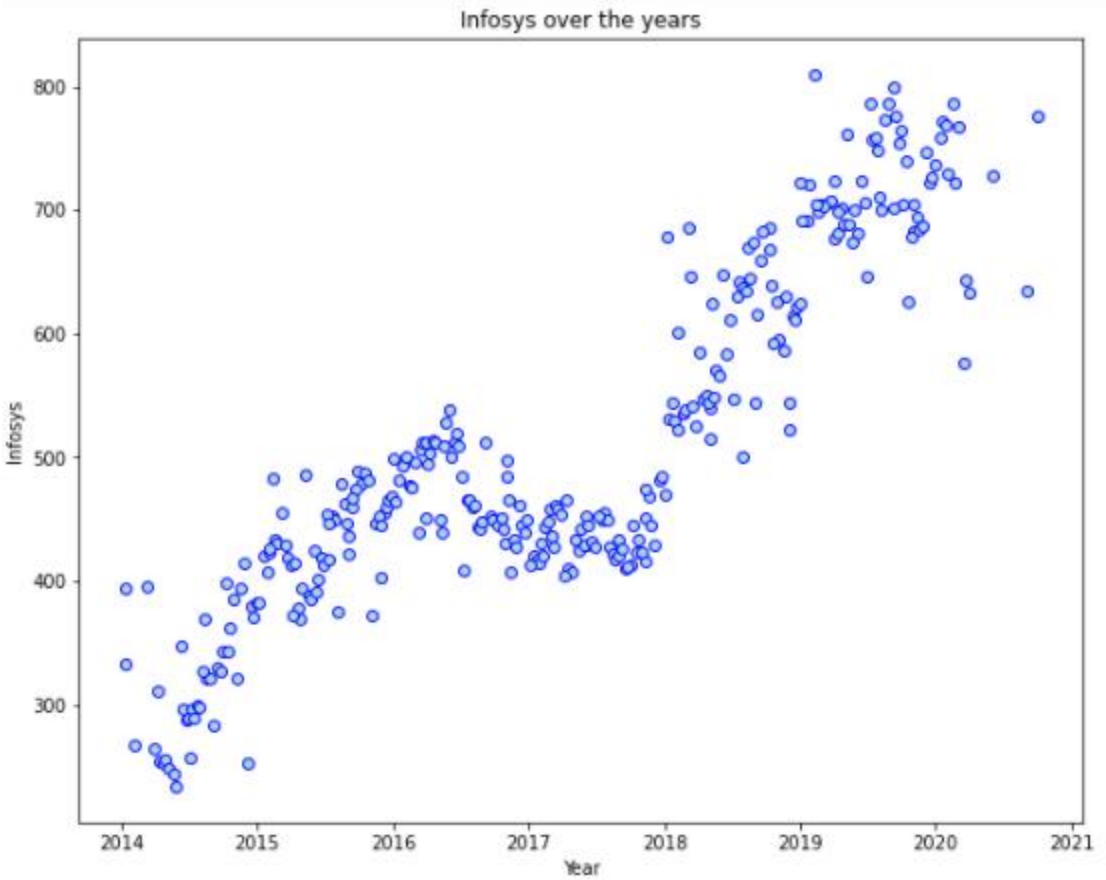
- The Descriptive Statistics is shown using the 5 point summary.

	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

- There are no null values present in the data.
- There are no duplicate values are preset in the data.
- The data is taken from the year 2014 – 2020.

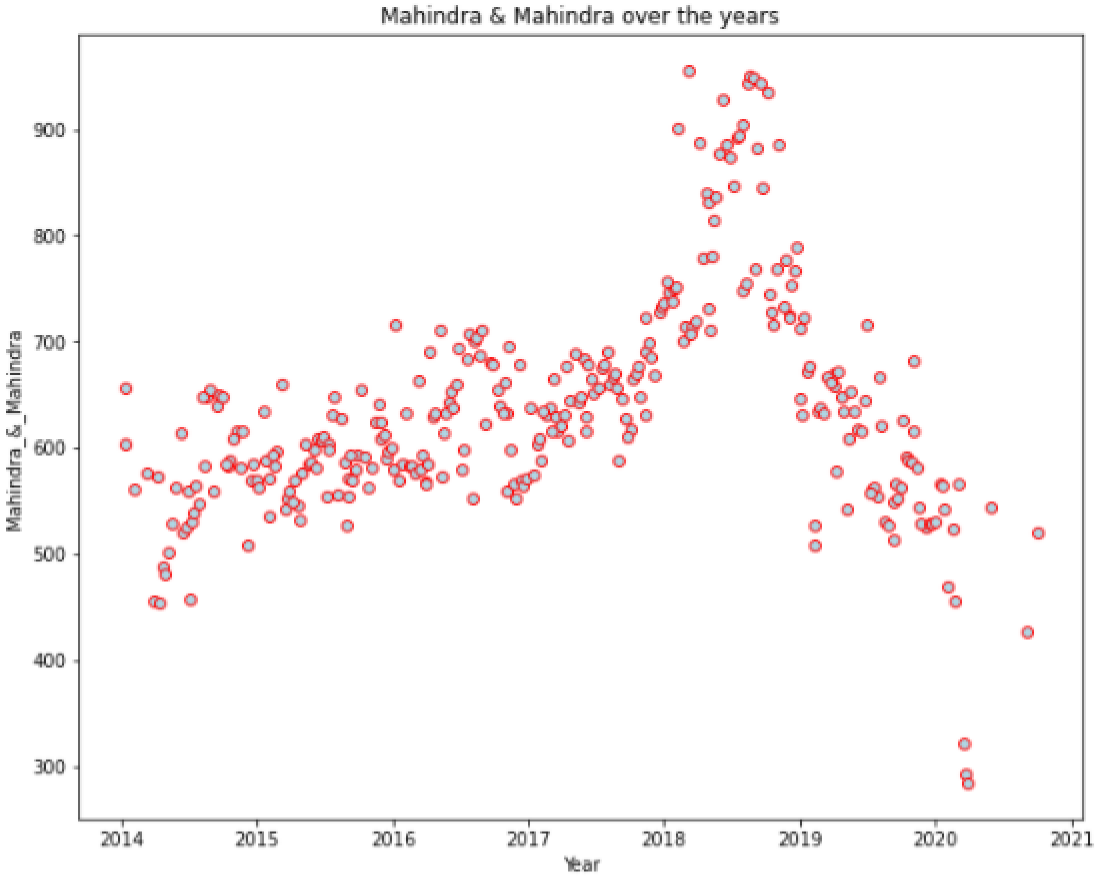
Stock Price Graph (Stock Price vs Time):

- **INFOSYS**

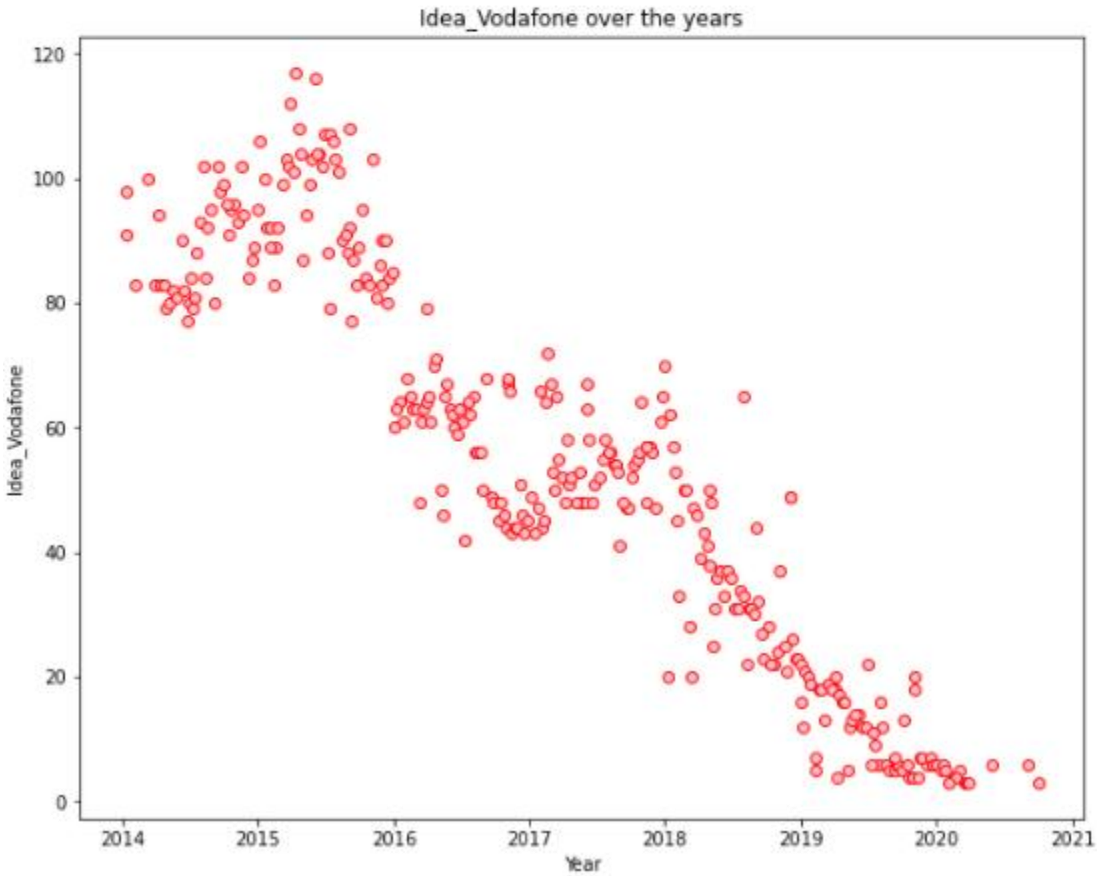


  - The overall trend for the Infosys Stock price is seen upwards.
  - There is flat curve during the period mid 2016 to 2018.
  - The stock price of the company again rose higher after 2018.

- **Mahindra & Mahindra**
  - The Stock prices of the Mahinda & Mahindra company seem to have downward trend.
  - At the initial period, there is a flat curve seen between 2014 and 2018.
  - Over the years, the stock price has raised during 2018-2019.
  - After 2018, the stock price started to drop than the initial period.

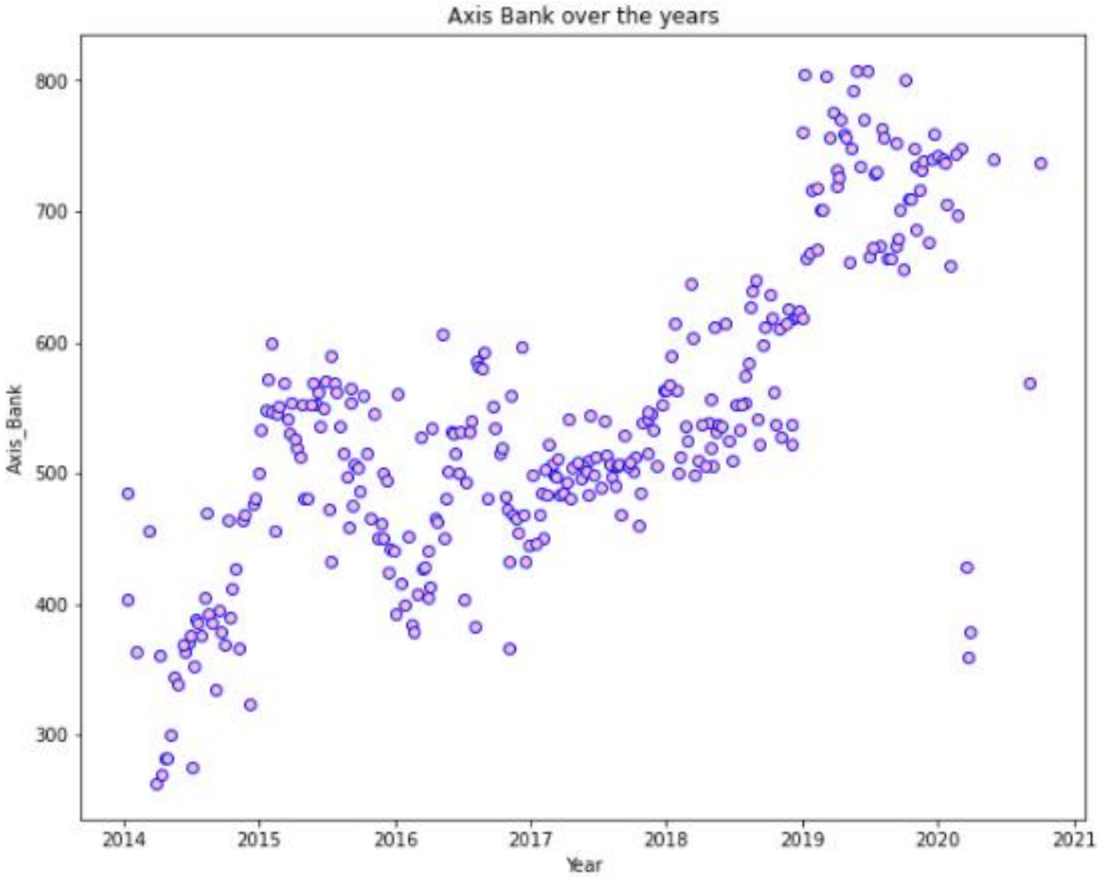


➢ **Idea Vodafone**



- The Idea Vodafone was separate telecommunication companies and was merged on March 2017.
- Before their merger, Vodafone was the second-largest company and Idea being the third-largest company.
- At the initial phase in 2014, both the companies had high stock prices.
- There seems the overall downward trend, even after their merger.
- The stock prices seem to reach the 0 during mid of 2019.

➢ Axis Bank



- The Stock prices of the Axis bank has upward trend.
- At 2015, it reached the high and there were slight fluctuations till 2017.
- There is gradual increase in the stock price during the period 2016 to 2019.
- Overall, the stock price seems to rise till the year 2020.

ANALYZING RETURNS

- Steps for calculating returns from prices:
  - Take logarithms
  - Take differences
- Use the function log() from the numpy package.
- After taking log, values are differenced once for all the company stocks (row-wise).
- Here the date column is dropped from the dataframe at initial.
- Checking the head of the data as 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
5	-0.031749	-0.015504	0.040656	0.061875	0.061558	0.011400	-0.008217	0.024898	0.012579	-0.016598
6	0.019961	0.060625	0.011881	0.076961	0.112795	0.067622	-0.016639	0.097543	0.048790	0.020705
7	-0.036221	0.199333	0.038615	0.059898	0.136859	0.056790	-0.049881	0.105732	-0.024098	0.169258
8	-0.041847	-0.012121	0.064183	-0.014642	-0.023530	0.048090	0.044835	-0.010084	-0.012270	-0.181630
9	0.135666	0.081917	-0.003559	0.071154	0.213574	0.105167	-0.018724	0.132686	0.024391	0.072031

INFERENCE

- All the companies have mixed high and low returns across the data.
- The first row is null as there is no records prior to that available in the dataset.

Looking at Means & Standard Deviations of the returns:

- **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.

- We can check the Stock mean as below.

```
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
```

- We can check the Stock Standard deviation as below.

```
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
```

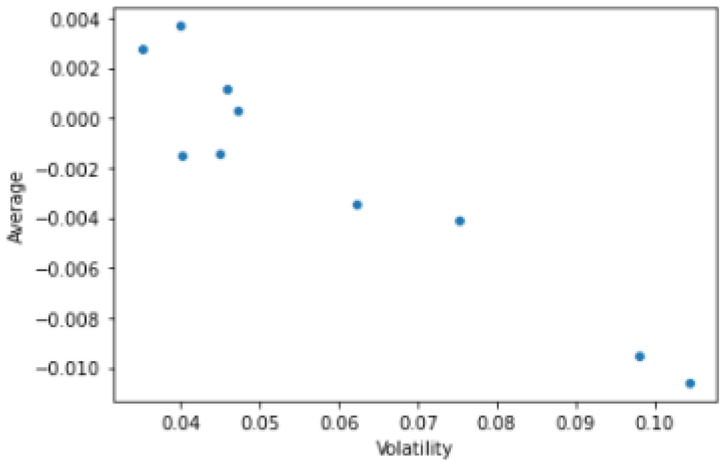
- We name the Stock Mean as Average and Stock Standard deviation as Volatility.
- Creating a dataframe and checking the combined values across the companies as below.

	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

INFERENCE

- There are stocks which has low volatility with high average returns such as Shree Cement and Vodafone.
- There are stocks which has negative average returns with high volatility such as Idea Vodafone and Jet Airways.
- Some stocks have moderate volatility and better average returns such as Axis bank and Indian Hotel.

Plot of Stock Means vs Standard Deviation:



INFERENCE

- ❖ Shree_Cement has high average returns of 0.003681 with low volatility as 0.039917(highest in stock).
- ❖ Infosys has low volatility as 0.035070 with high average returns as 0.002794(second highest in stock).
- ❖ Jet_Airways has very low average return as -0.009548 with very high volatility of 0.097972 (second lowest in stock).

- ❖ Idea_Vodafone has very low average return as -0.010608 with very high volatility of 0.104315(lowest in stock).

CONCLUSIONS:

- ❖ Stock with a lower mean & higher standard deviation does 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 - Ones with higher return for a comparative or lower risk are considered better.

RECOMMENDATIONS:

- ❖ The preference for the stocks will be higher for the ones with low volatility and high/moderate average returns.
- ❖ The stock prices with moderate fluctuating are generally opted.
- ❖ To keep the stock prices as high or moderate depends on the overall performance of the company.
- ❖ The factors that help in keeping the stock price floating must be monitored in regular basis.
- ❖ This way it helps the company to raise their stock price year on year and earn more revenues for the company.