CREDIT RISK

Problem Statement

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

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

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

Build a Random Forest Model on Train Dataset

Initially we read the dataset, do the EDA, impute the missing values and NaNs that were outliers and then split the train and test sets and then build a model based on Random Forest.
We import the required packages and libraries i.e RandomForestClassifier from the sklearn.ensemble.

```
RandomForestClassifier(max_leaf_nodes=7)
```

We select the number of trees as 100 and give 7 as max leaf nodes and fit the model. Random Forest model is immune to outliers and works. Now we predict on train and test sets and then check for model validation based on the performance metrics.

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

```
Train Set Accuracy:- 0.985428809325562
Test Set Accuracy:- 0.9831081081081081
```

```
Confusion Matrix for Train Set:-
 [[2154    3]
 [  32  213]]
```

```
Classification Report for Train Set:-
               precision    recall  f1-score   support

         0.0       0.99      1.00      0.99      2157
         1.0       0.99      0.87      0.92       245

    accuracy                           0.99      2402
   macro avg       0.99      0.93      0.96      2402
weighted avg       0.99      0.99      0.99      2402
```

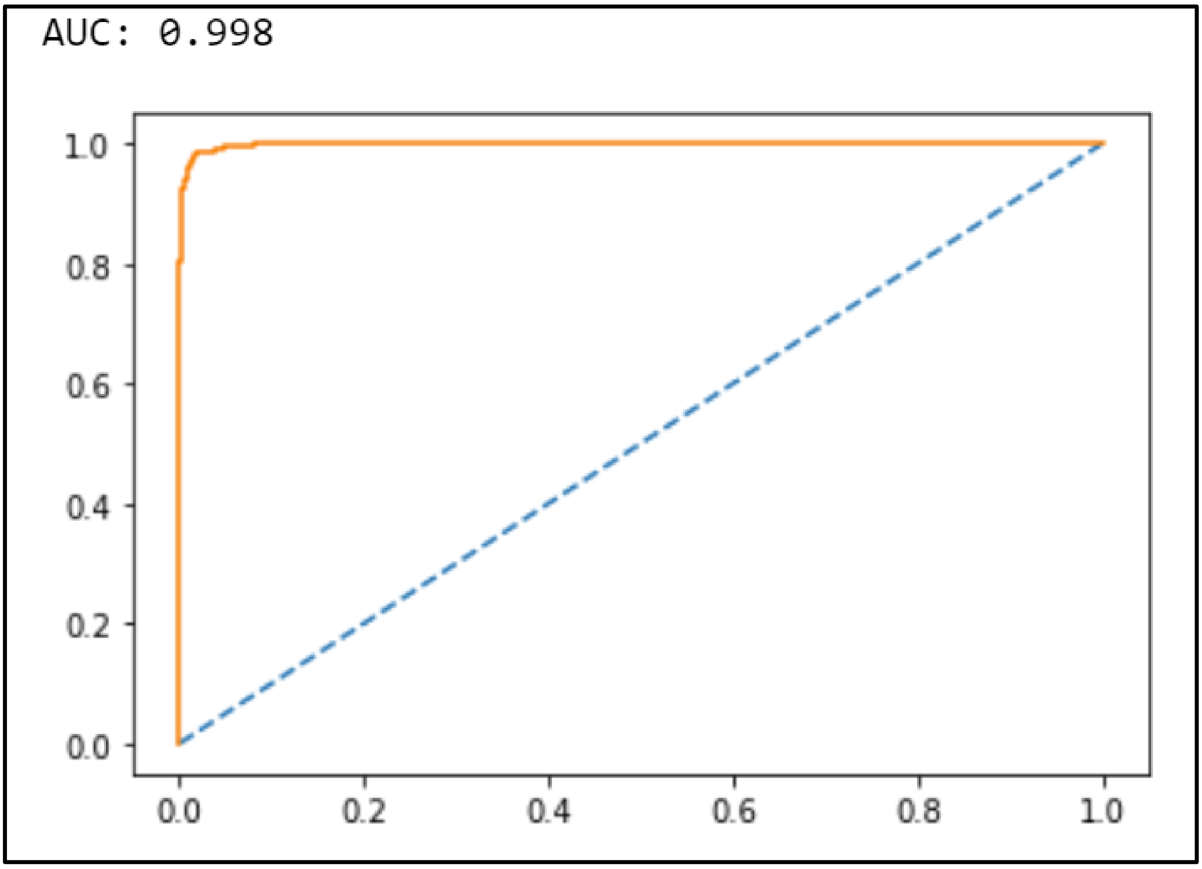
```
Confusion Matrix for Test Set:-
 [[1038    3]
 [  17  126]]
```

```
Classification Report for Test Set:-
               precision    recall  f1-score   support

         0.0       0.98      1.00      0.99      1041
         1.0       0.98      0.88      0.93       143

    accuracy                           0.98      1184
   macro avg       0.98      0.94      0.96      1184
weighted avg       0.98      0.98      0.98      1184
```

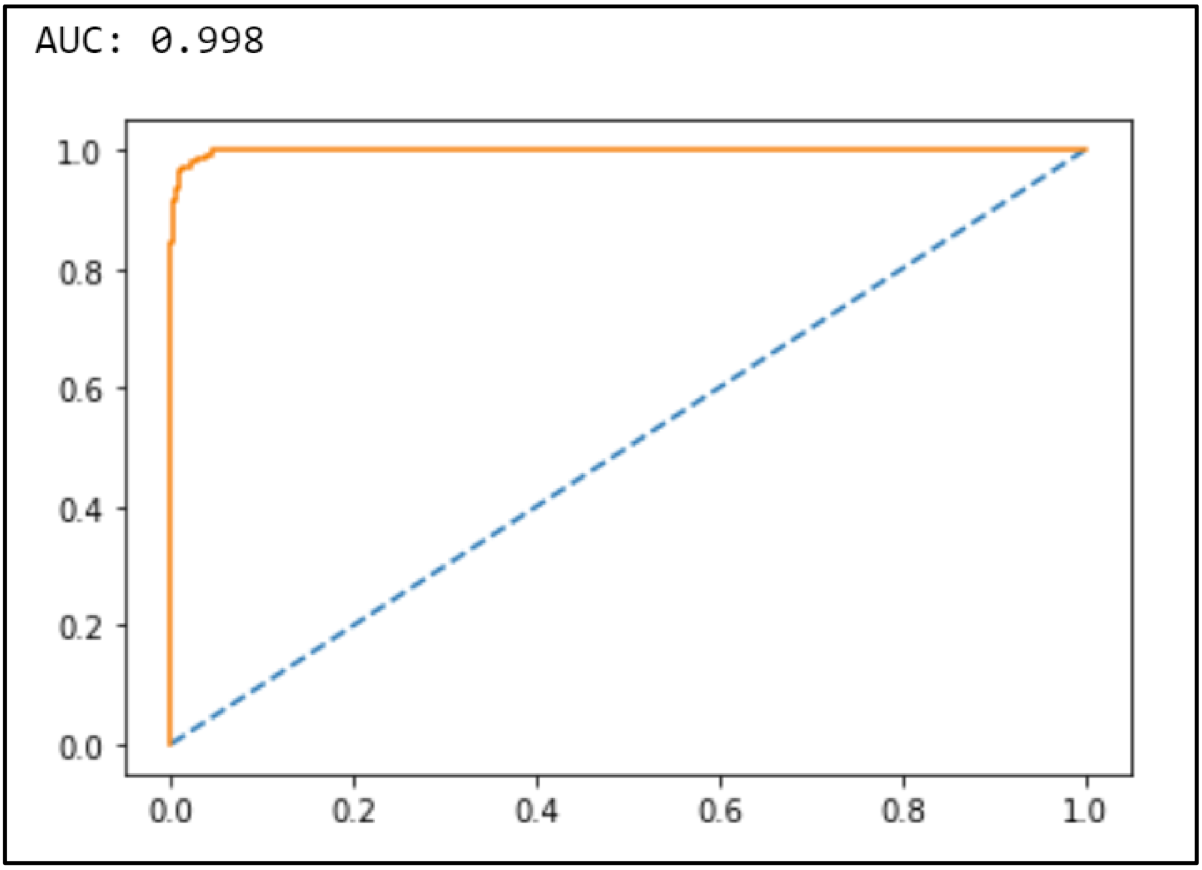
From the above performance measures, we can see that the accuracy is really high i.e the model predicts the data for both train and test sets very well. The confusion matrix also shows that the number of false positives and true negatives are really less . The classification report shows the recall is really good for predicting both – defaulters and non-defaulters and the precision cannot be any better. Hence this is one of the best models.

AUC and ROC curves

Train Set



Test Set



Build a LDA Model on Train Dataset

Initially we read the dataset, do the EDA, impute the missing values and NaNs that were outliers and then split the train and test sets and then build a model based on LDA.
We import the required packages and libraries i.e LinearDiscriminantAnalysis from the sklearn.discriminant_analysis.

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

```
Train Set Accuracy:- 0.9350541215653622
Test Set Accuracy:- 0.9197635135135135

Confusion Matrix for Train Set:-
 [[2127   30]
 [ 126  119]]

Classification Report for Train Set:-
              precision    recall  f1-score   support

         0.0       0.94      0.99      0.96      2157
         1.0       0.80      0.49      0.60       245

    accuracy                           0.94      2402
   macro avg       0.87      0.74      0.78      2402
weighted avg       0.93      0.94      0.93      2402


Confusion Matrix for Test Set:-
 [[1029   12]
 [  83   60]]

Classification Report for Test Set:-
              precision    recall  f1-score   support

         0.0       0.93      0.99      0.96      1041
         1.0       0.83      0.42      0.56       143

    accuracy                           0.92      1184
   macro avg       0.88      0.70      0.76      1184
weighted avg       0.91      0.92      0.91      1184
```

From the above metrics and performance measures, we can see that Train and Test accuracy is good but less than RandomForest. This is because Random Forest handles outliers well. The recall and precision for non-defaulters is good but recall or prediction for defaulters is not as good. We have also checked how we can improve this model.

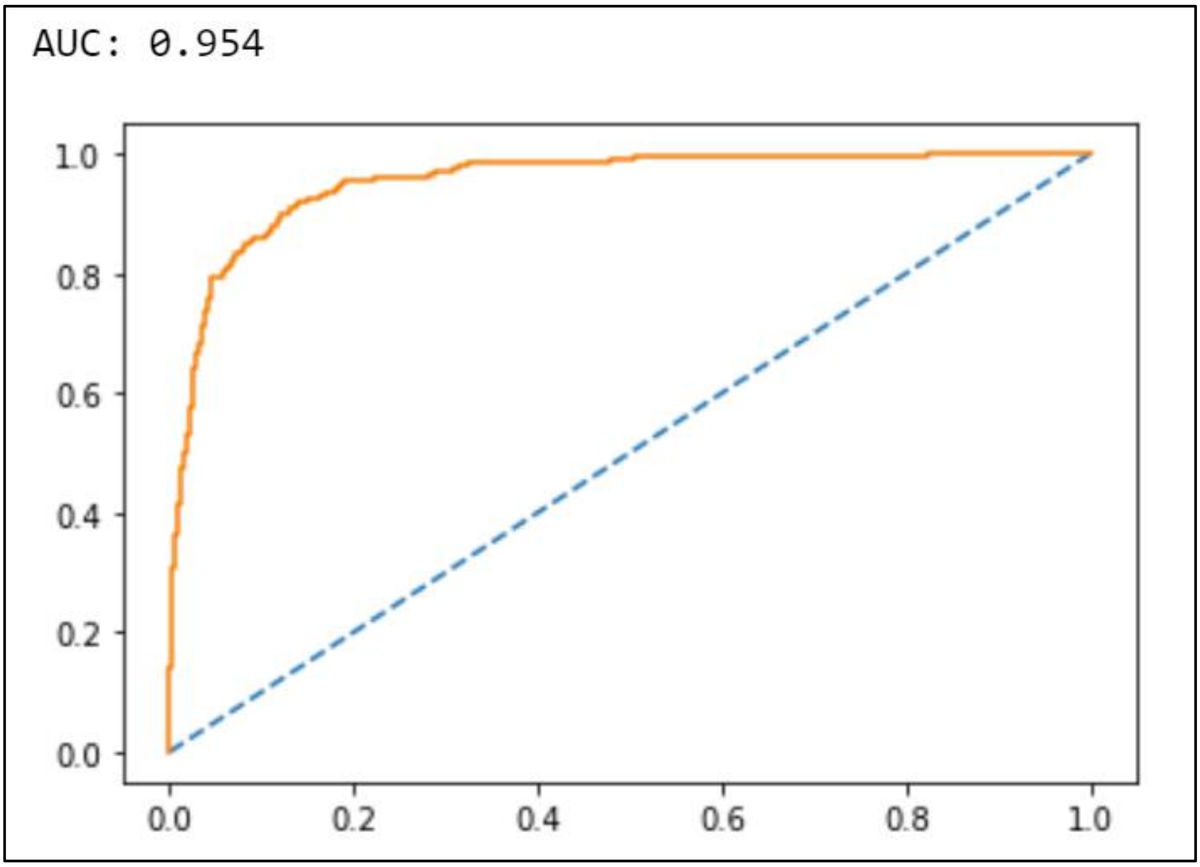
So, we change the cut-off values or the threshold values and find that 0.2 is the optimal value.

```
0.1

Accuracy Score 0.912
F1 Score 0.6624

0.15

Accuracy Score 0.929
F1 Score 0.6952

0.2

Accuracy Score 0.938
F1 Score 0.7098

0.25

Accuracy Score 0.938
F1 Score 0.6888

0.3

Accuracy Score 0.938
F1 Score 0.6769

0.35

Accuracy Score 0.935
F1 Score 0.6453

0.4

Accuracy Score 0.934
F1 Score 0.6238

0.45

Accuracy Score 0.933
F1 Score 0.6025
```

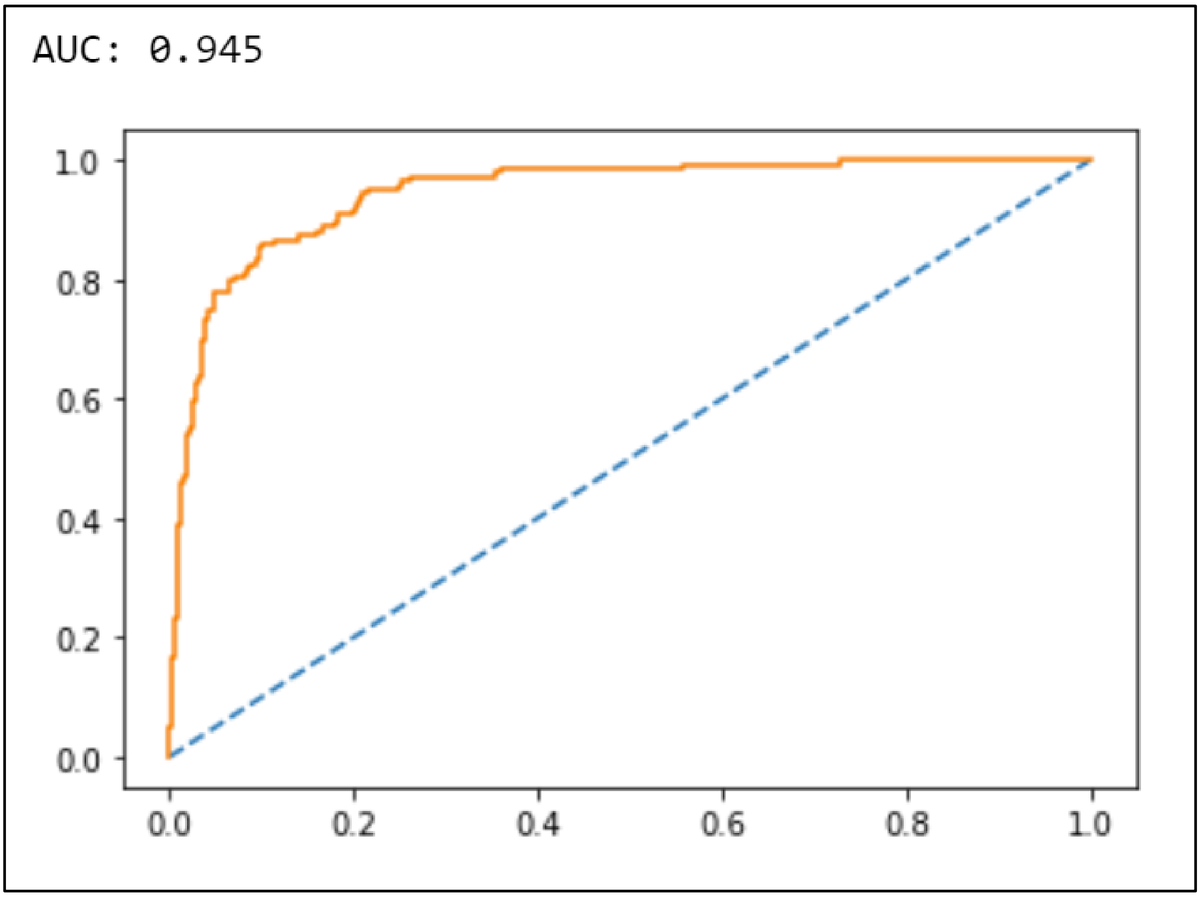
```
0.5

Accuracy Score 0.935
F1 Score 0.6041

0.55

Accuracy Score 0.933
F1 Score 0.5737

0.6

Accuracy Score 0.93
F1 Score 0.5499

0.65

Accuracy Score 0.93
F1 Score 0.5359

0.7

Accuracy Score 0.928
F1 Score 0.4971

0.75

Accuracy Score 0.927
F1 Score 0.4854

0.8

Accuracy Score 0.925
F1 Score 0.4578

0.85

Accuracy Score 0.924
F1 Score 0.4277

0.9

Accuracy Score 0.921
F1 Score 0.3883

0.95

Accuracy Score 0.918
F1 Score 0.3356
```

AUC and ROC curves

Train Set

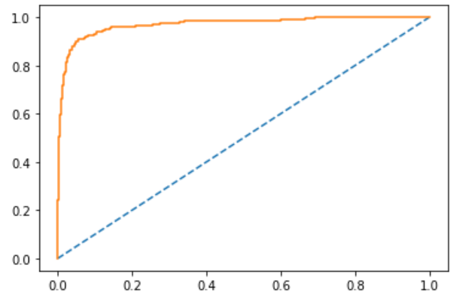


Test Set

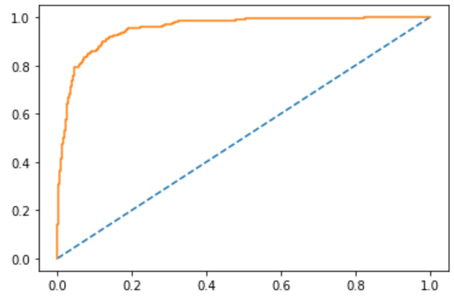


Compare the performances Logistics, Radom Forest and LDA models (include ROC Curve)

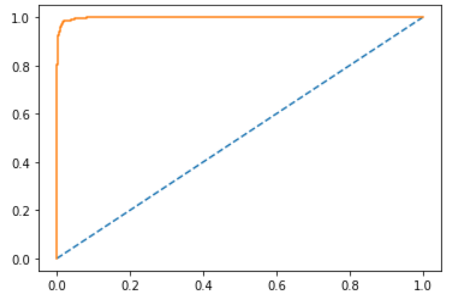
Parameters	Default	Logistic Regression	Linear Discriminant Analysis	Random Forest
Train_Accruacy		95.58%	93.50%	98.54%
Test_Accruacy		95.35%	91.97%	98.31%
Train_Precision	0	96.00%	94.00%	99.00%
	1	88.00%	80.00%	99.00%
Test_Precision	0	96.00%	93.00%	98.00%
	1	88.00%	83.00%	98.00%
Train_Recall	0	99.00%	99.00%	100.00%
	1	66.00%	49.00%	87.00%
Test_Recall	0	99.00%	99.00%	100.00%
	1	71.00%	42.00%	88.00%
Train_AUC		97.10%	95.40%	99.80%
Test_AUC		97.60%	94.50%	99.80%



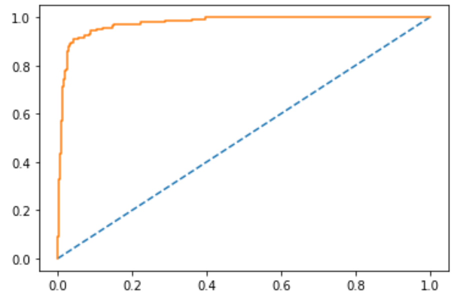
Train Set ROC - LR



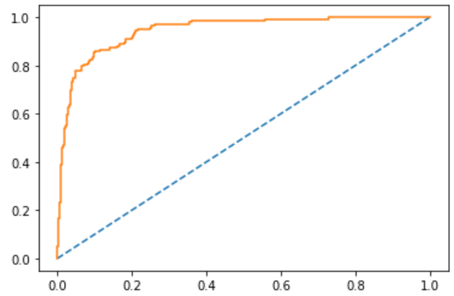
Train Set ROC - LDA



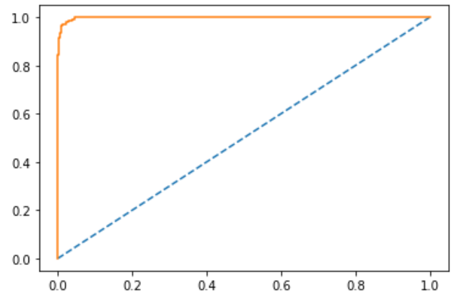
Train Set ROC - RF



Test Set ROC - LR



Test Set ROC - LDA



Test Set ROC - RF

State Recommendations from the above models

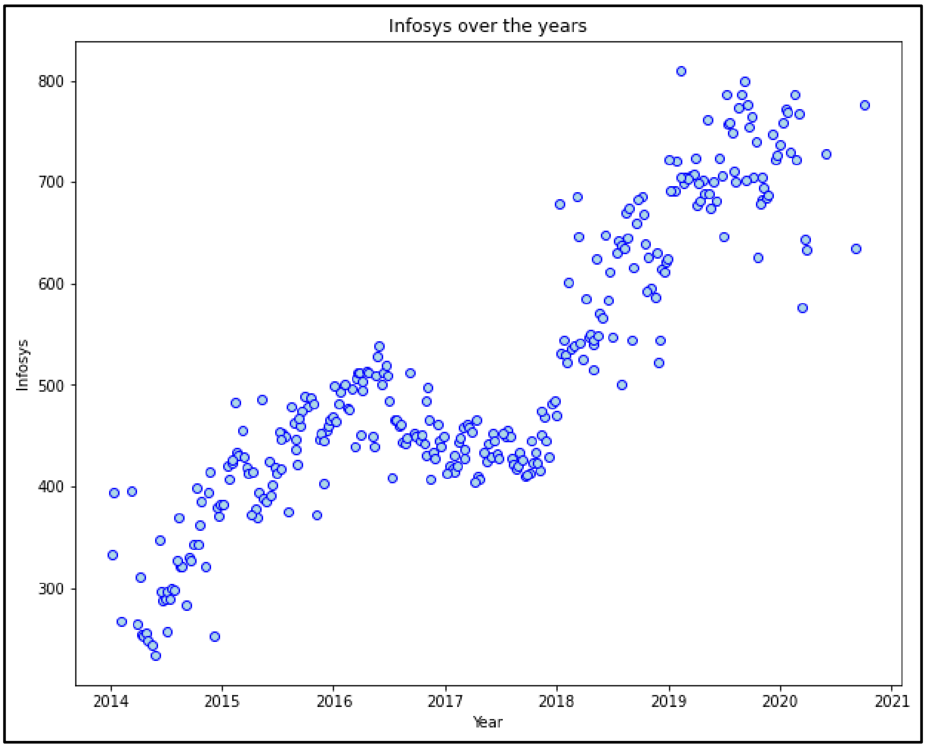
Based on the models that we have plotted, we can see that Random Forest does the best, this is because we have imputed the values for the outliers and this model, works better than the other two when we still have outliers in our model. We can change the threshold rate to 0.2 if we wish to use the LDA model. Or another approach can be removing the outliers or capping it to mean/ whisker values and then using that data with Logistic Regression or LDA model. From the data, we can see that True Negatives are more in number than False Positives and we can recommend that we can change the thresholds to lessen that data. We can thereby predict in more efficient way.

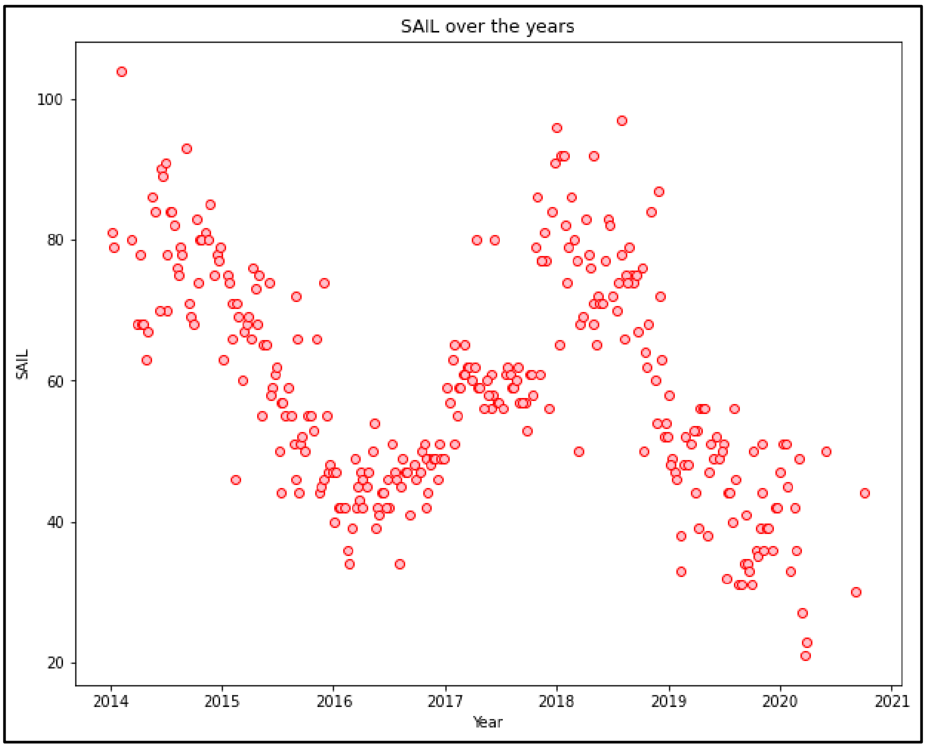
MARKET-RISK and RETURNS

The dataset has data of 10 stock listed companies over a period from 2014 to 2021. So we have to calculate the stock means, volatility and trends over time.

Draw Stock Price Graph(Stock Price vs Time) for any 2 given stocks

We use the to_datetime() to split the date to year,month and even granular components. And then we plot scatterplot for stocks with stock price on Y- axis and date(Year) on X- axis.





Calculate Returns for all stocks

Returns for the stocks can be calculated as –

Take the logarithm of stock prices

- Compute their differences
- Calculate Stock Means and Standard Deviation for all stocks.

We use log() and diff() for the same.

```
stock_returns.head()
```

Shape of data: (314, 10)

	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

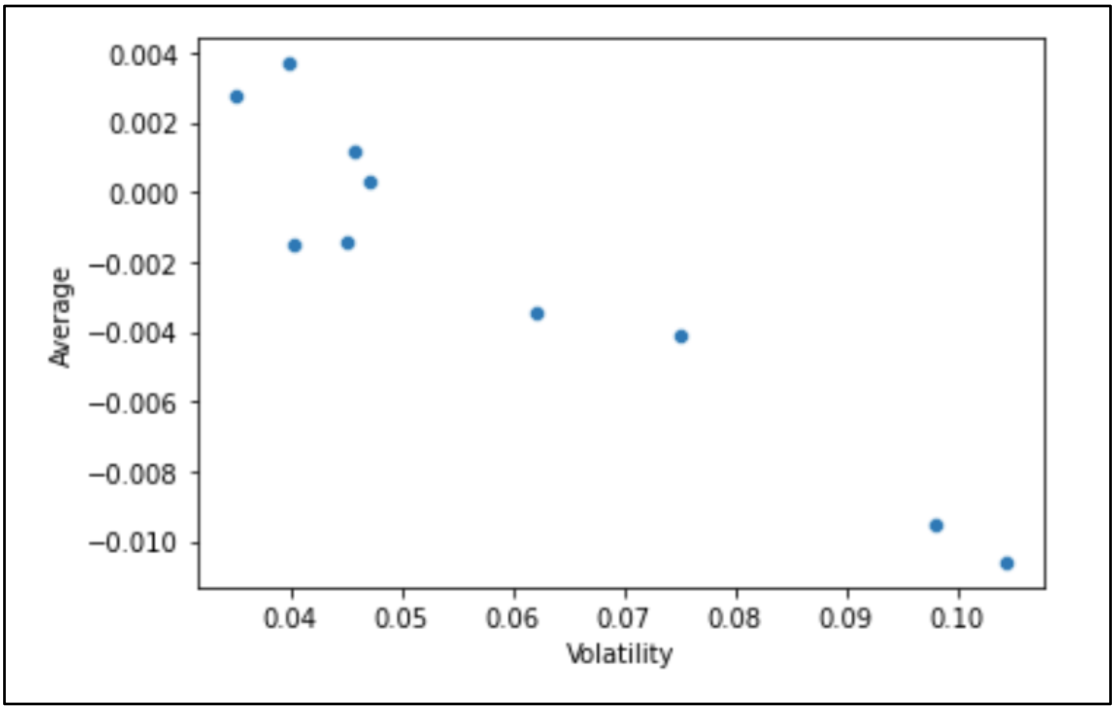
Draw a plot of Stock Means vs Standard Deviation

Means shows the average value of the stock returns and standard deviation shows the volatility of the stock.

```
stock_means

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

```
stock_sd

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
```

	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



The above plotted graph is a scatterplot of stock volatility (Standard Deviation) vs stock average (Mean)

Conclusion and Recommendations

We can see that the average returns are highest on Infosys and Shree Cements, so we can infer that the IT and Construction/Cement companies have done better compared to the other sectors in the timespan of last 7 years. The most volatile stocks were Idea_Vodafone and Jet Airways followed by SAIL and Jindal_Steel. So we can see that steel and telecom industry were effected a lot by sentiments of the market and Jet may be volatile because of tourism being a seasonal industry. We can recommend trading the less volatile stocks often and trading the more volatile stocks as per the sentiments of the markets and based on their seasonalities. Shree Cements, Infosys and Indian_Hotels have been more profitable (based on returns) and have been more stable in volatility compared to the other stocks. So, in terms of long term investments we can add more Shree Cements and Infosys to the portfolio.