FINANCIAL RISK ANALYSIS

Project FRA

Abstract

This assignment delves into the realm of financial risk management, focusing on the critical task of identifying and predicting defaulters to mitigate credit risk for a company. It involves the exploration of diverse machine learning models, including Logistic Regression, Random Forest, and Linear Discriminant Analysis (LDA), along with techniques like SMOTE for handling imbalanced data. The analysis extends to model evaluation, hyperparameter tuning, and comparison, providing insights into precision, recall, and F1-score metrics. Tailored recommendations are offered to align the modeling approach with the overarching business goal of reducing credit risk effectively

Sudheendra.K
Sudhi0404@gmail.com

Contents

PART A

Problem Statement:

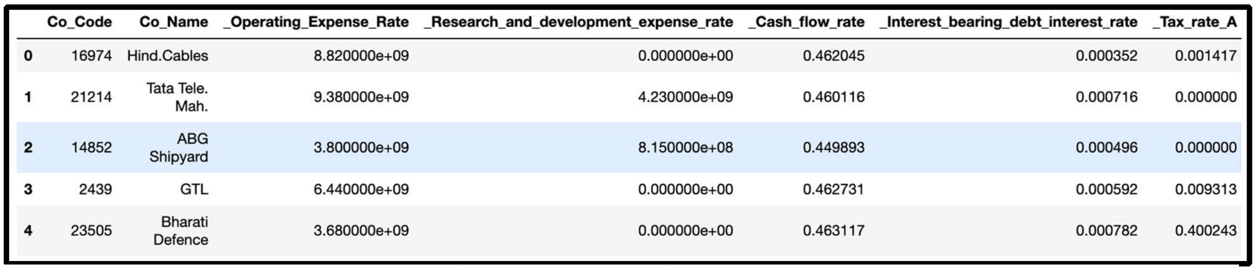
Businesses or companies can fall prey to default if they are not able to keep up their debt obligations. Defaults will lead to a lower credit rating for the company which in turn reduces its chances of getting credit in the future and may have to pay higher interest on existing debts as well as any new 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.

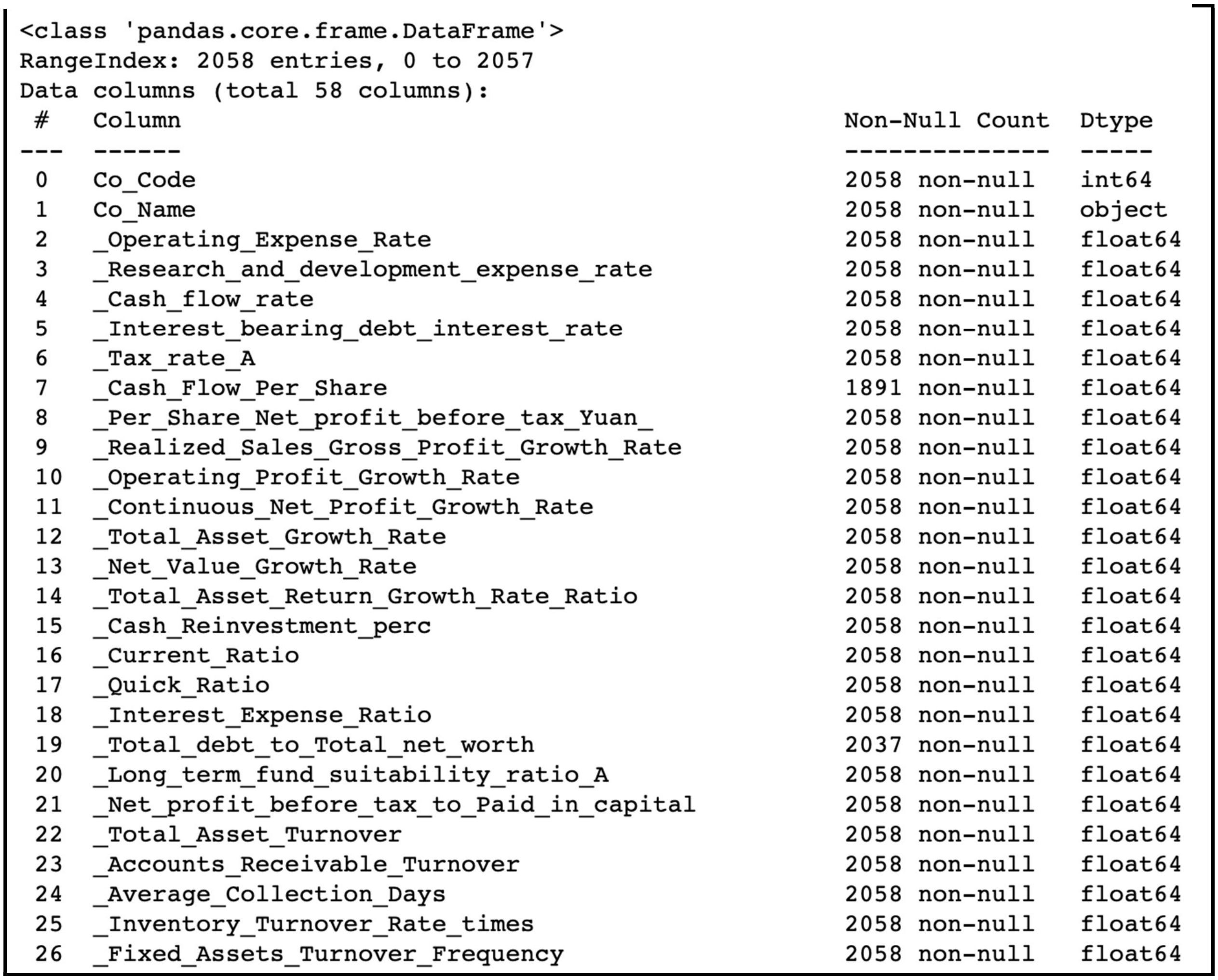
EDA

The preview of the dataset



	Co_Code	Co_Name	_Operating_Expense_Rate	_Research_and_development_expense_rate	_Cash_flow_rate	_Interest_bearing_debt_interest_rate	_Tax_rate_A
0	16974	Hind.Cables	8.820000e+09	0.000000e+00	0.462045	0.000352	0.001417
1	21214	Tata Tele. Mah.	9.380000e+09	4.230000e+09	0.460116	0.000716	0.000000
2	14852	ABG Shipyard	3.800000e+09	8.150000e+08	0.449893	0.000496	0.000000
3	2439	GTL	6.440000e+09	0.000000e+00	0.462731	0.000592	0.009313
4	23505	Bharati Defence	3.680000e+09	0.000000e+00	0.463117	0.000782	0.400243

Snippet 1. 1 Head of the dataset



```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2058 entries, 0 to 2057
Data columns (total 58 columns):
 #   Column                                    Non-Null Count  Dtype  
---  ------                                    --------------  -----  
 0   Co_Code                                   2058 non-null   int64  
 1   Co_Name                                   2058 non-null   object 
 2   _Operating_Expense_Rate                   2058 non-null   float64
 3   _Research_and_development_expense_rate    2058 non-null   float64
 4   _Cash_flow_rate                           2058 non-null   float64
 5   _Interest_bearing_debt_interest_rate      2058 non-null   float64
 6   _Tax_rate_A                               2058 non-null   float64
 7   _Cash_Flow_Per_Share                      1891 non-null   float64
 8   _Per_Share_Net_profit_before_tax_Yuan_    2058 non-null   float64
 9   _Realized_Sales_Gross_Profit_Growth_Rate  2058 non-null   float64
 10  _Operating_Profit_Growth_Rate             2058 non-null   float64
 11  _Continuous_Net_Profit_Growth_Rate        2058 non-null   float64
 12  _Total_Asset_Growth_Rate                  2058 non-null   float64
 13  _Net_Value_Growth_Rate                    2058 non-null   float64
 14  _Total_Asset_Return_Growth_Rate_Ratio     2058 non-null   float64
 15  _Cash_Reinvestment_perc                   2058 non-null   float64
 16  _Current_Ratio                            2058 non-null   float64
 17  _Quick_Ratio                              2058 non-null   float64
 18  _Interest_Expense_Ratio                   2058 non-null   float64
 19  _Total_debt_to_Total_net_worth            2037 non-null   float64
 20  _Long_term_fund_suitability_ratio_A       2058 non-null   float64
 21  _Net_profit_before_tax_to_Paid_in_capital 2058 non-null   float64
 22  _Total_Asset_Turnover                     2058 non-null   float64
 23  _Accounts_Receivable_Turnover             2058 non-null   float64
 24  _Average_Collection_Days                  2058 non-null   float64
 25  _Inventory_Turnover_Rate_times            2058 non-null   float64
 26  _Fixed_Assets_Turnover_Frequency          2058 non-null   float64
```

Snippet 1. 2 Data attributes

- Number of Rows and Columns: The dataset contains 2058 rows and 58 columns, indicating that there are 2058 observations and 58 variables.

- Data Types: The dataset primarily consists of three data types:

- Integer: There are four integer columns, including "Co_Code," "_Liability_Assets_Flag," "_Net_Income_Flag," and "Default."

- Float: Most of the columns are of float data type, containing decimal values. These columns represent various financial ratios and metrics.

- Object: There is one object column named "Co_Name," which likely contains the names of the companies.

- Missing Values: Some columns have missing values, which are indicated by the "Non-Null Count" in the summary. For example, the "_Cash_Flow_Per_Share" column has missing values, as well as a few other columns like "_Total_debt_to_Total_net_worth," "_Cash_to_Total_Assets," and "_Current_Liability_to_Current_Assets."

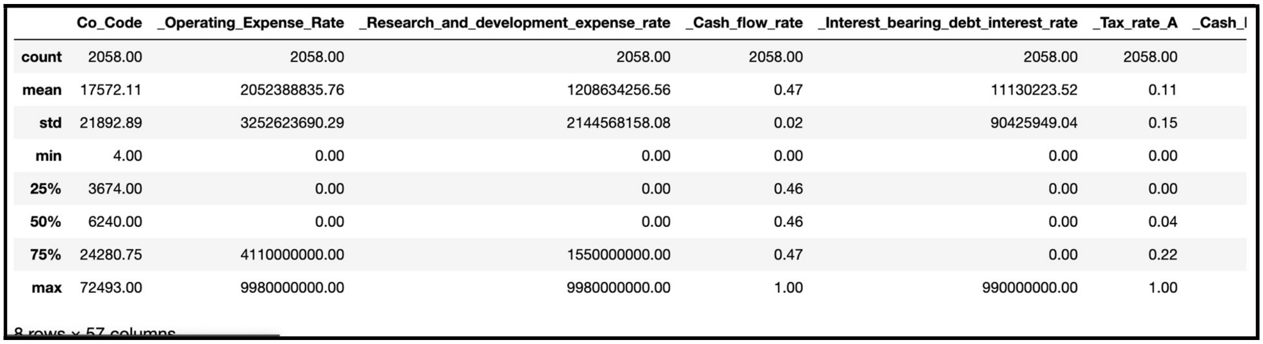
- Numeric Data: The majority of the dataset consists of numeric data, with 53 columns containing float values. These columns likely represent various financial and operational metrics of the companies.

- Categorical Data: There are a few integer columns, such as "_Liability_Assets_Flag" and "_Net_Income_Flag," which might be binary indicators or flags.

- Target Variable: The "Default" column appears to be the target variable, as mentioned in problem statement. It is of integer type, which suggests it might be a binary variable indicating whether a company defaulted (1) or not (0).

- Company Names: The "Co_Name" column likely contains the names of the companies in the dataset, which could be useful for identification but might not be directly used in predictive modeling.

Before proceeding with any analysis or modeling, it's essential to handle missing values, encode categorical variables if necessary, and explore the relationships between the independent variables and the target variable. Additionally, understanding the context of the financial ratios and metrics in the dataset is crucial for meaningful analysis.

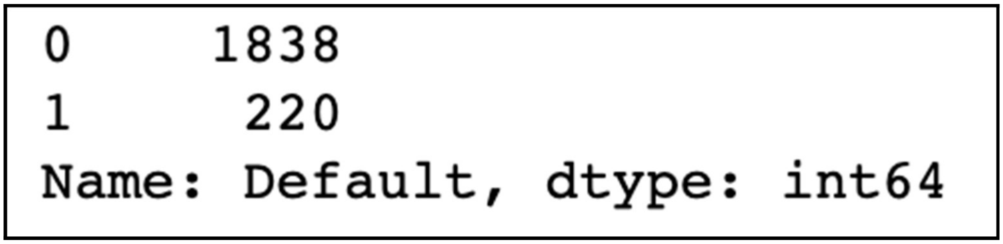


	Co_Code	_Operating_Expense_Rate	_Research_and_development_expense_rate	_Cash_flow_rate	_Interest_bearing_debt_interest_rate	_Tax_rate_A	_Cash_
count	2058.00	2058.00	2058.00	2058.00	2058.00	2058.00	
mean	17572.11	2052388835.76	1208634256.56	0.47	11130223.52	0.11	
std	21892.89	3252623690.29	2144568158.08	0.02	90425949.04	0.15	
min	4.00	0.00	0.00	0.00	0.00	0.00	
25%	3674.00	0.00	0.00	0.46	0.00	0.00	
50%	6240.00	0.00	0.00	0.46	0.00	0.04	
75%	24280.75	4110000000.00	1550000000.00	0.47	0.00	0.22	
max	72493.00	9980000000.00	9980000000.00	1.00	990000000.00	1.00	

8 rows × 57 columns

Snippet 1. 3 Data Description

- The above is the snippet of the statistical summary of the data set



```
0     1838
1      220
Name: Default, dtype: int64
```

Snippet 1. 4 Default Variable Counts

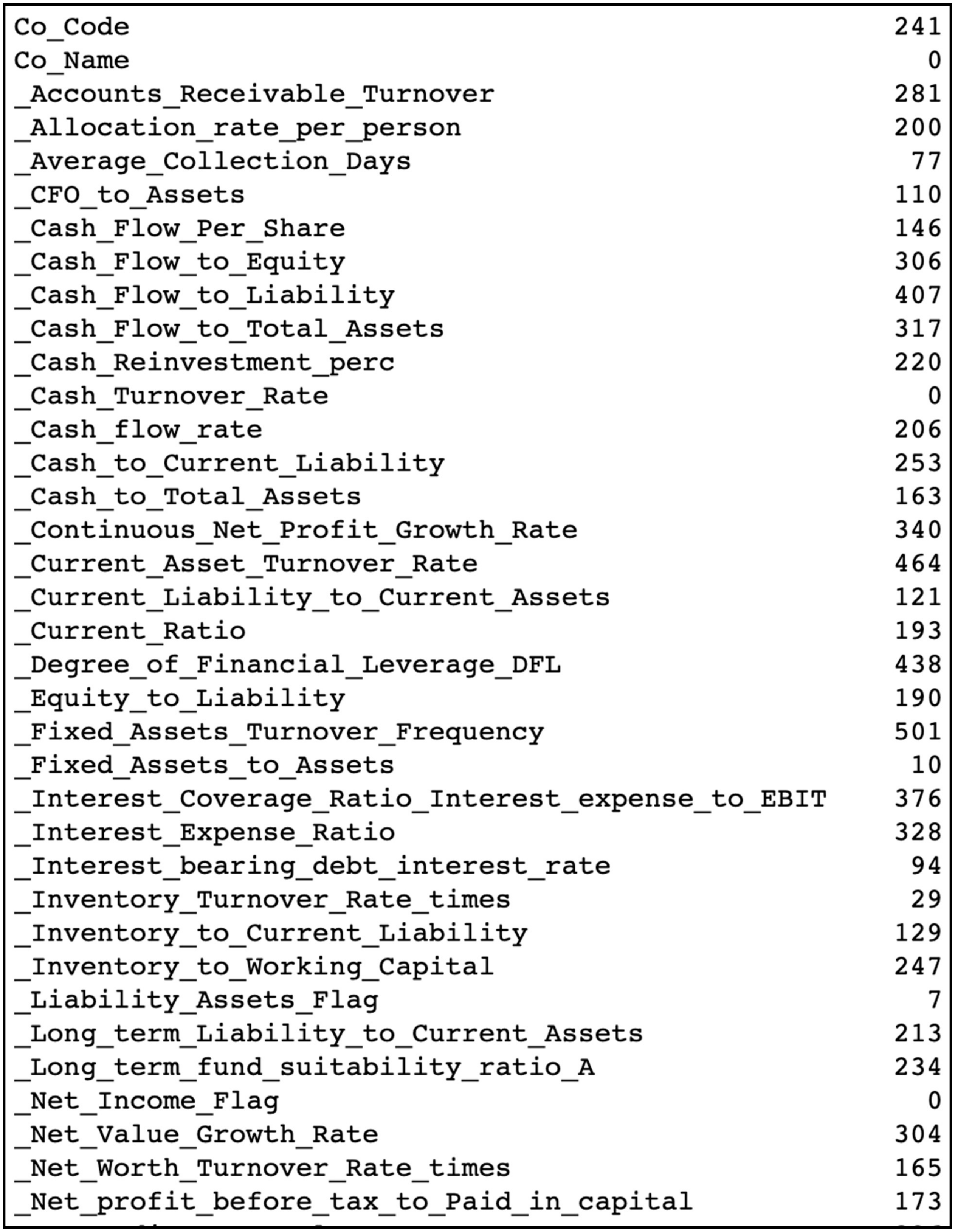
- The "Default" variable in the dataset has two distinct values, and here are their counts:

- Default value 0: There are 1,838 instances where the company did not default.

- Default value 1: There are 220 instances where the company did default.

This information provides an overview of the distribution of default and non-default cases in the dataset. It's important to note that imbalanced datasets like this one (where one class significantly outnumbers the other) can sometimes pose challenges in machine learning model training and evaluation. Depending on analysis goals, we may need to consider techniques like resampling, using different evaluation metrics, or employing advanced modeling approaches to handle the class imbalance appropriately.

1. Outlier Treatment

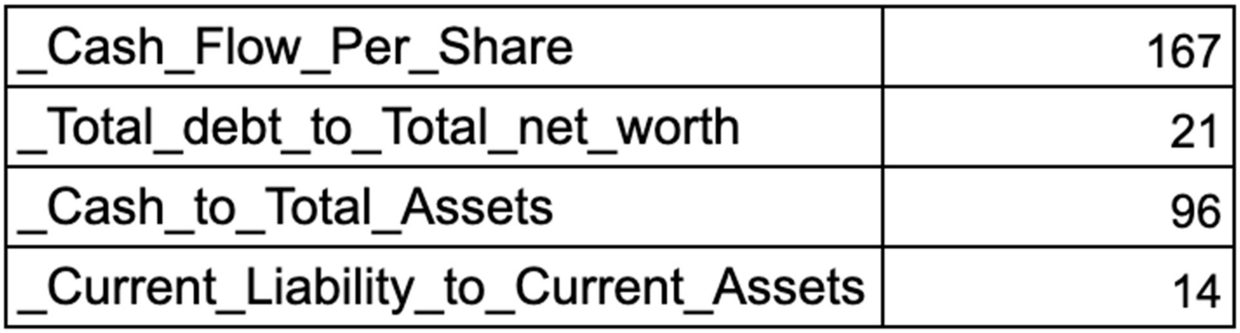


```
Co_Code                                             241
Co_Name                                               0
_Accounts_Receivable_Turnover                       281
_Allocation_rate_per_person                         200
_Average_Collection_Days                             77
_CFO_to_Assets                                      110
_Cash_Flow_Per_Share                                146
_Cash_Flow_to_Equity                                306
_Cash_Flow_to_Liability                             407
_Cash_Flow_to_Total_Assets                          317
_Cash_Reinvestment_perc                             220
_Cash_Turnover_Rate                                   0
_Cash_flow_rate                                     206
_Cash_to_Current_Liability                          253
_Cash_to_Total_Assets                               163
_Continuous_Net_Profit_Growth_Rate                  340
_Current_Asset_Turnover_Rate                        464
_Current_Liability_to_Current_Assets                121
_Current_Ratio                                      193
_Degree_of_Financial_Leverage_DFL                   438
_Equity_to_Liability                                190
_Fixed_Assets_Turnover_Frequency                    501
_Fixed_Assets_to_Assets                              10
_Interest_Coverage_Ratio_Interest_expense_to_EBIT   376
_Interest_Expense_Ratio                             328
_Interest_bearing_debt_interest_rate                 94
_Inventory_Turnover_Rate_times                       29
_Inventory_to_Current_Liability                     129
_Inventory_to_Working_Capital                       247
_Liability_Assets_Flag                                7
_Long_term_Liability_to_Current_Assets              213
_Long_term_fund_suitability_ratio_A                 234
_Net_Income_Flag                                      0
_Net_Value_Growth_Rate                              304
_Net_Worth_Turnover_Rate_times                      165
_Net_profit_before_tax_to_Paid_in_capital           173
```

Snippet 1. 5 Outliers

The summary, that is this snippet calculates the upper and lower outlier thresholds for each column based on the IQR and then replaces any data points outside these thresholds with NaN values. This is a common method for handling outliers, and it helps prevent extreme values from unduly affecting the analysis or modeling process. Keep in mind that this treatment strategy does not remove the outliers entirely but rather masks them by setting their values to NaN.

2. Missing Value Treatment



_Cash_Flow_Per_Share	167
_Total_debt_to_Total_net_worth	21
_Cash_to_Total_Assets	96
_Current_Liability_to_Current_Assets	14

Snippet 1. 6 Missing Values

Here's what the result indicates:

- _Cash_Flow_Per_Share: This column has the highest proportion of missing values, accounting for approximately 167 counts of missing values in the dataset.
- _Total_debt_to_Total_net_worth: This column has the third-highest proportion of missing values, accounting for approximately 21 counts of missing values.
- _Current_Liability_to_Current_Assets: This column has the second-highest proportion of missing values, accounting for approximately 96 counts of missing values.

After identifying these columns with a high proportion of missing values(about 20% we shall drop off the top three columns This is a common data preprocessing step to address missing data and reduce the dimensionality of the dataset when columns have a substantial number of missing values.

3. Univariate (4 marks) & Bivariate (6 marks) analysis with proper interpretation. (You may choose to include only those variables which were significant in the model building)

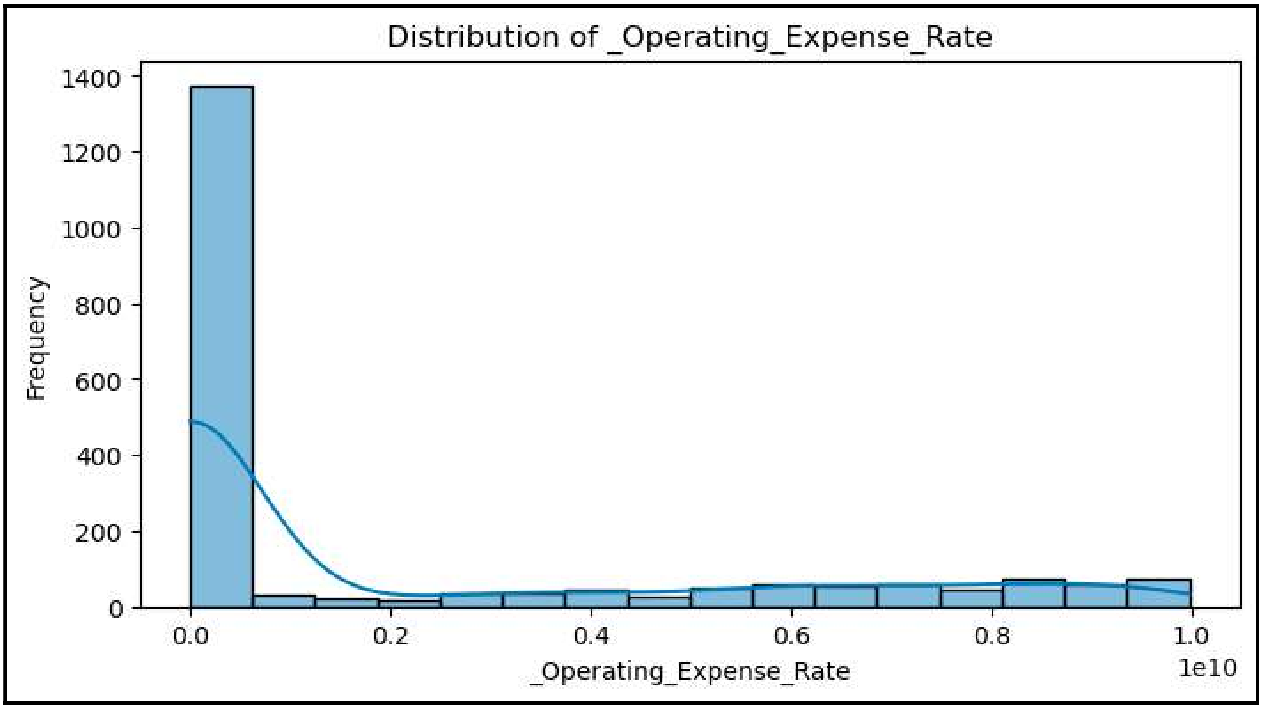


Figure 1. 1 Histogram for Operating Expenses

- We could observe that distribution of the attribute is Right skewed

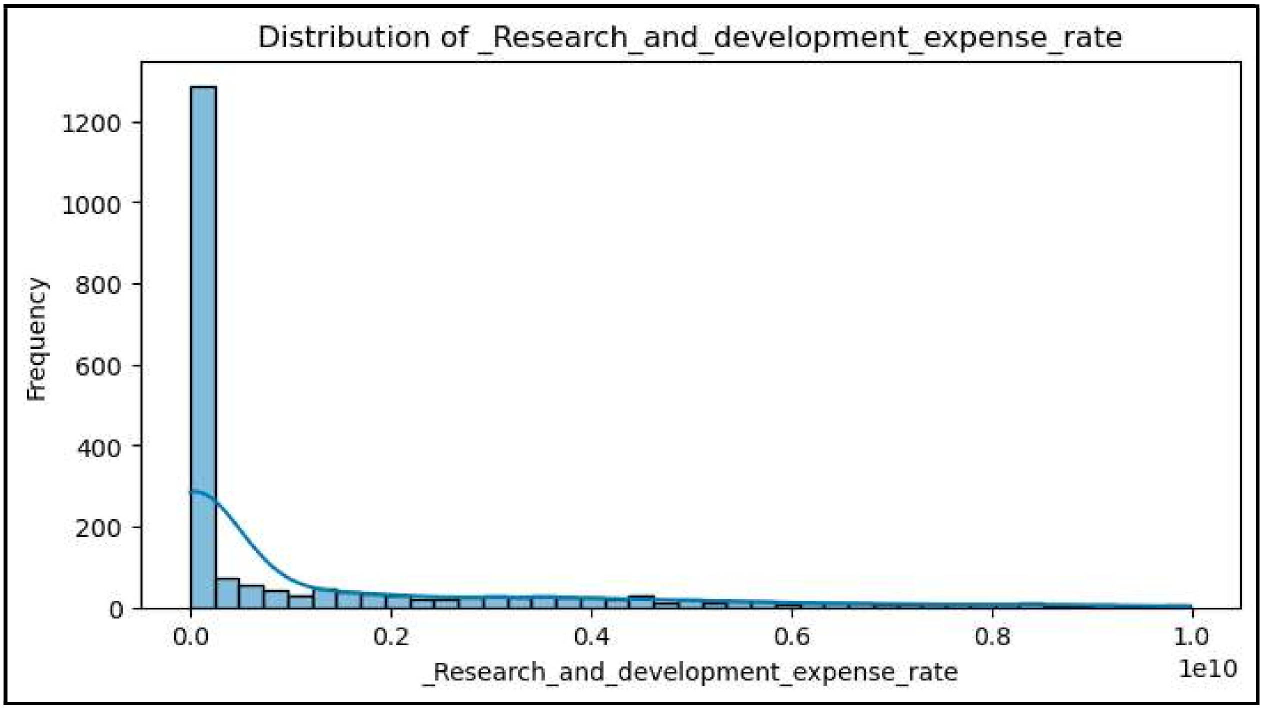


Figure 1. 2 Histogram for R&D expense rate

- We could observe that distribution of the attribute is Right skewed

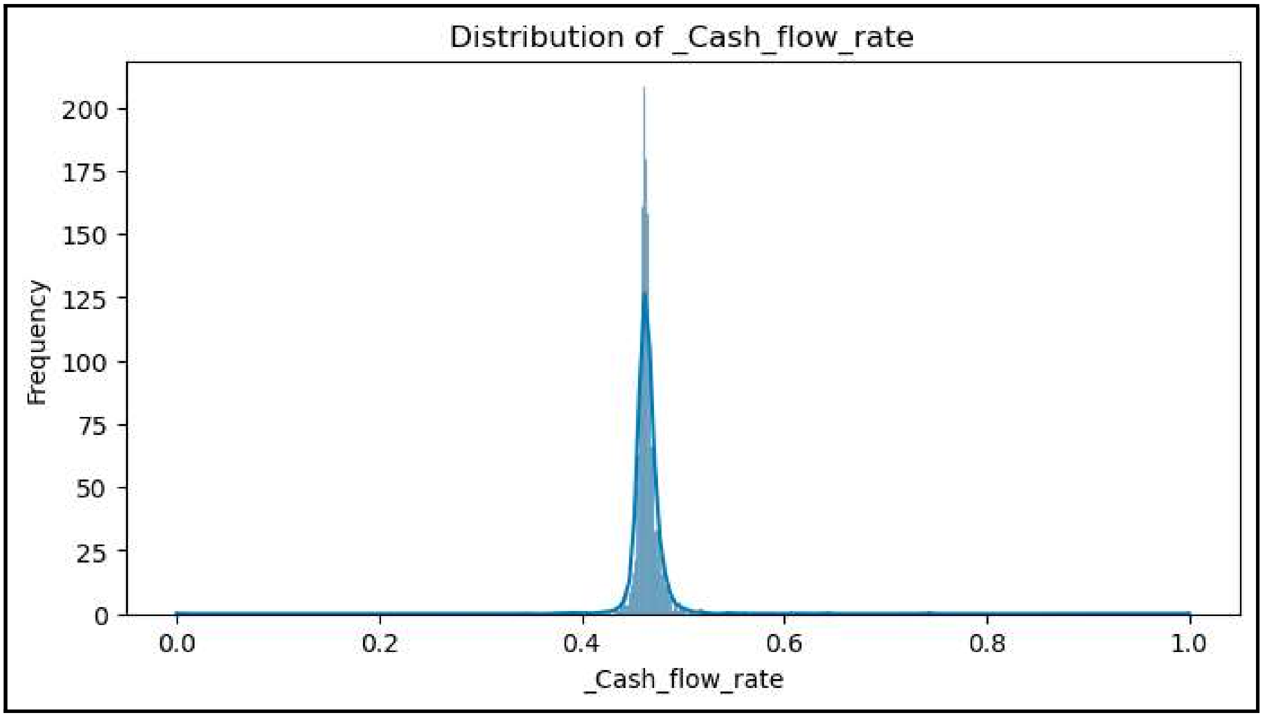


Figure 1. 3 Histogram for cash flow rate

- The distribution of Cash flow follows normal distribution

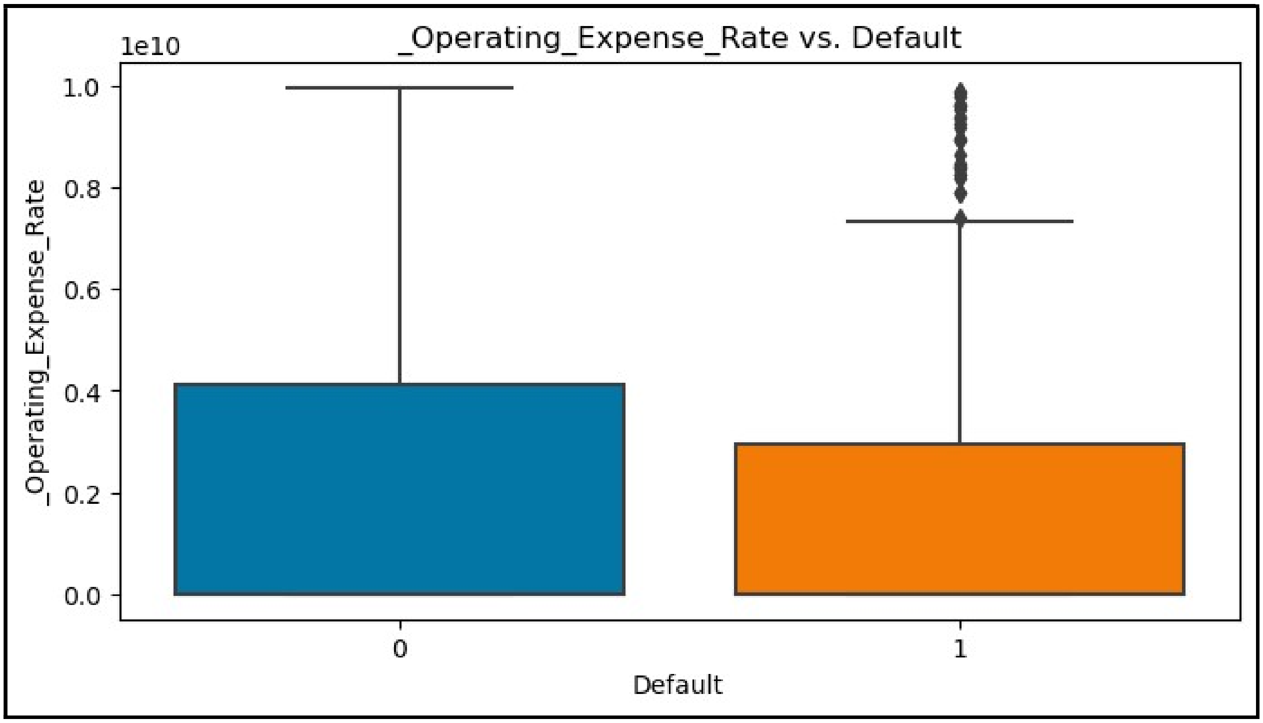


Figure 1. 4 Box plot for Default Vs Operating expenses

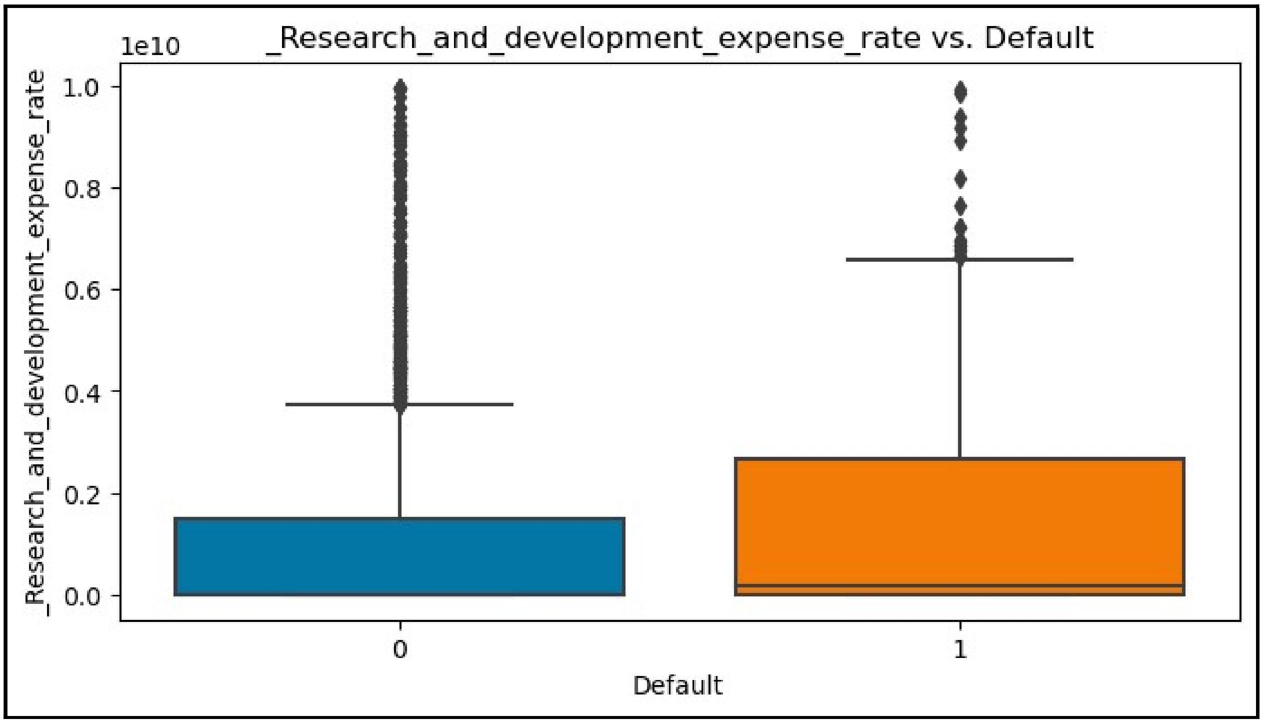


Figure 1. 5 Box plot for Default Vs R&D expense rate

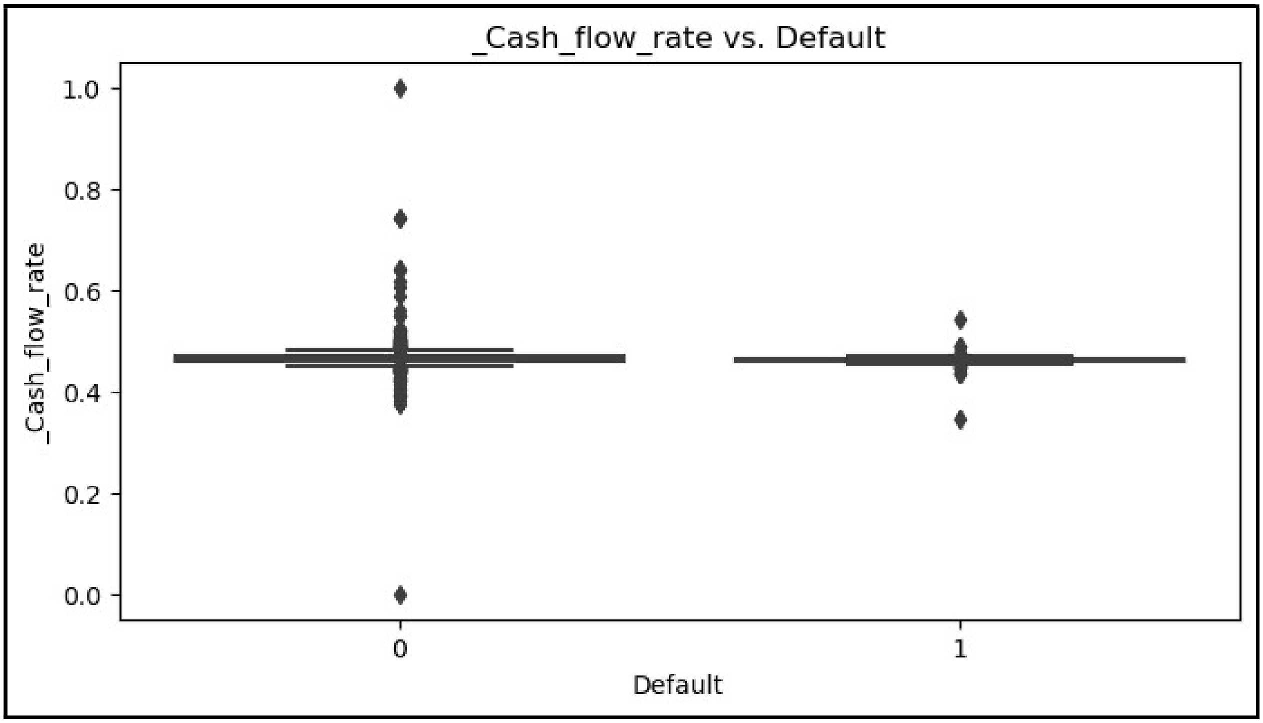


Figure 1. 6 Box plot for Default Vs Cashflow rate

- We can witness the outliers in most of the attributes

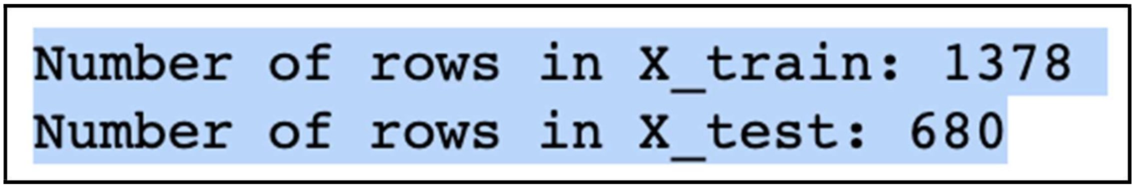
- This excerpt outlines a procedure to determine the upper and lower boundaries for outliers in each column using the Interquartile Range (IQR).

- Any data points that fall outside of these boundaries are subsequently replaced with NaN values.

- This technique serves as a common approach to address outliers, safeguarding against the undue influence of extreme values during analysis or modeling.

- It's important to note that this method doesn't eliminate outliers entirely; instead, it conceals them by assigning NaN values in their place.

4. Split the data into train and test datasets in the ratio of 67:33 and use a random state of 42 (random_state=42). Model building is to be done on the train dataset and model validation is to be done on the test dataset.



```
Number of rows in X_train: 1378
Number of rows in X_test: 680
```

Snippet 1. 7 Train Test split

- The data set has been divided into train and test on 67:33 ratio on a random state of 42 as prescribed.

5. Build Logistic Regression Model (using statsmodels library) on most important variables on train dataset and choose the optimum cut-off. Also showcase your model building approach

- Initialization: The model building process starts with the initialization of a logistic regression classifier (LogR).

- Feature Selection: Recursive Feature Elimination (RFE) is employed as a feature selection technique to choose the most relevant features for predicting the target variable.

- Feature Ranking: RFE ranks the features based on their importance in predicting the target. Features are iteratively eliminated, and their rankings are determined.

- Top Feature Selection: The top 15 ranked features are selected for model building. These features are considered the most informative for the classification task.

- Model Training (Training Set): The logistic regression model is trained using the selected features on the training dataset (X_train and y_train).

- Model Prediction (Training Set): The trained model is used to make predictions on the same training dataset to evaluate its performance on data it has seen during training.

- Model Prediction (Test Set): The trained model is then tested on a separate test dataset (X_test) to assess its generalization performance on unseen data.

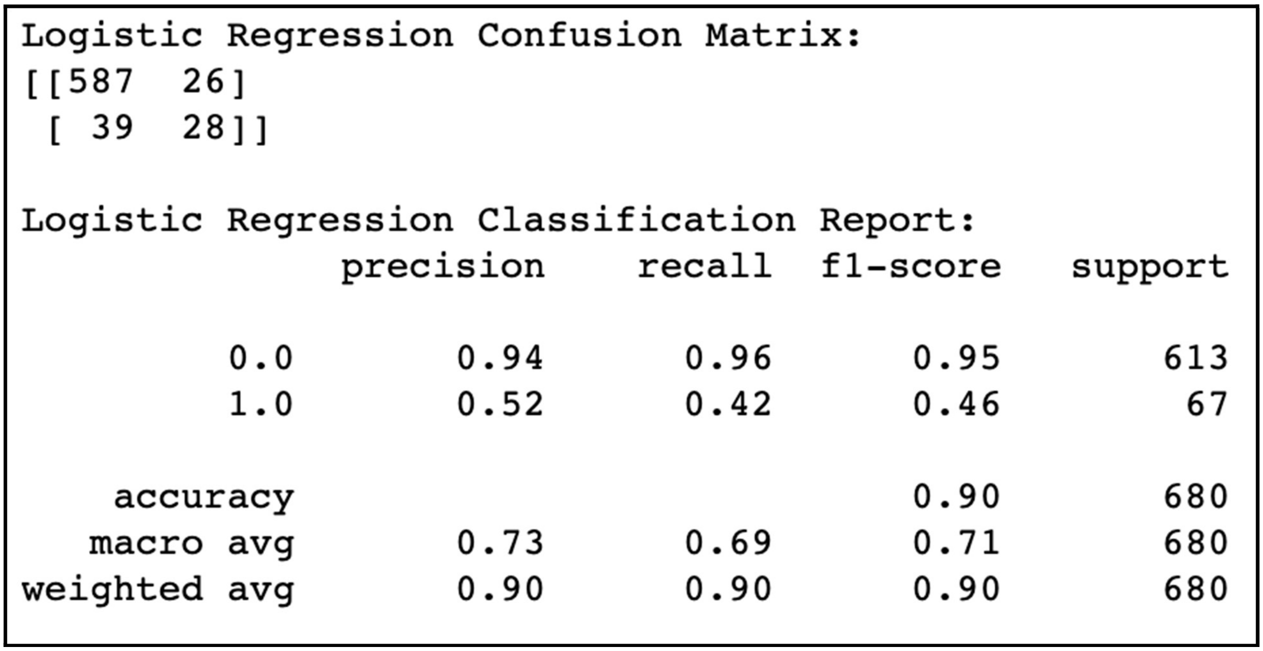
- Evaluation Metrics: The model's performance is evaluated using various classification metrics, such as precision, recall, and F1-score, on both the training and test datasets.

- Interpretability: By focusing on the top-ranked features, the model aims to provide insights into which features are the most influential in determining the target variable.

- Iterative Process: The model building process is often iterative, allowing for refinement and improvement of feature selection and model performance.

In summary, the approach combines feature selection using RFE with logistic regression modeling to build a predictive model. The emphasis is on selecting a subset of features that are most relevant for the classification task while ensuring model interpretability and generalization to unseen data.

6. Validate the Model on Test Dataset and state the performance metrics. Also state interpretation from the model



```
Logistic Regression Confusion Matrix:
[[587  26]
 [ 39  28]]

Logistic Regression Classification Report:
              precision    recall  f1-score   support

         0.0       0.94      0.96      0.95       613
         1.0       0.52      0.42      0.46        67

    accuracy                           0.90       680
   macro avg       0.73      0.69      0.71       680
weighted avg       0.90      0.90      0.90       680
```

Snippet 1. 8 Logistic Regression Performance on Test data

- This confusion matrix provides insights into the model's predictions. It shows that:

    1. True Positives (TP): 28 instances were correctly classified as defaulters.
    2. True Negatives (TN): 587 instances were correctly classified as non-defaulters.
    3. False Positives (FP): 26 instances were wrongly classified as defaulters.
    4. False Negatives (FN): 39 instances were wrongly classified as non-defaulters.

- Precision: The precision for non-defaulters (class 0) is high (0.94), indicating that when the model predicts a customer as a non-defaulter, it is correct 94% of the time. However, the precision for defaulters (class 1) is relatively low (0.52), suggesting that there are false positives.

- Recall: The recall for non-defaulters is high (0.96), indicating that the model correctly identifies 96% of non-defaulters. For defaulters, the recall is lower (0.42), meaning that the model misses some defaulters.

- F1-score: The F1-score is a balance between precision and recall. It is higher for non-defaulters (0.95) than for defaulters (0.46).

- Accuracy: The overall accuracy of the model is 90%, which means it correctly predicts the target variable in 90% of cases.

Interpretation:

- The model performs well in identifying non-defaulters, as indicated by high precision and recall.
- However, it struggles to identify defaulters, as indicated by lower precision and recall for class 1.

There is room for improvement in correctly identifying defaulters, possibly through feature engineering, model tuning, or addressing class imbalance.
Overall, the model is decent at making predictions but may require further refinement to be more effective in identifying potential defaulters.

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

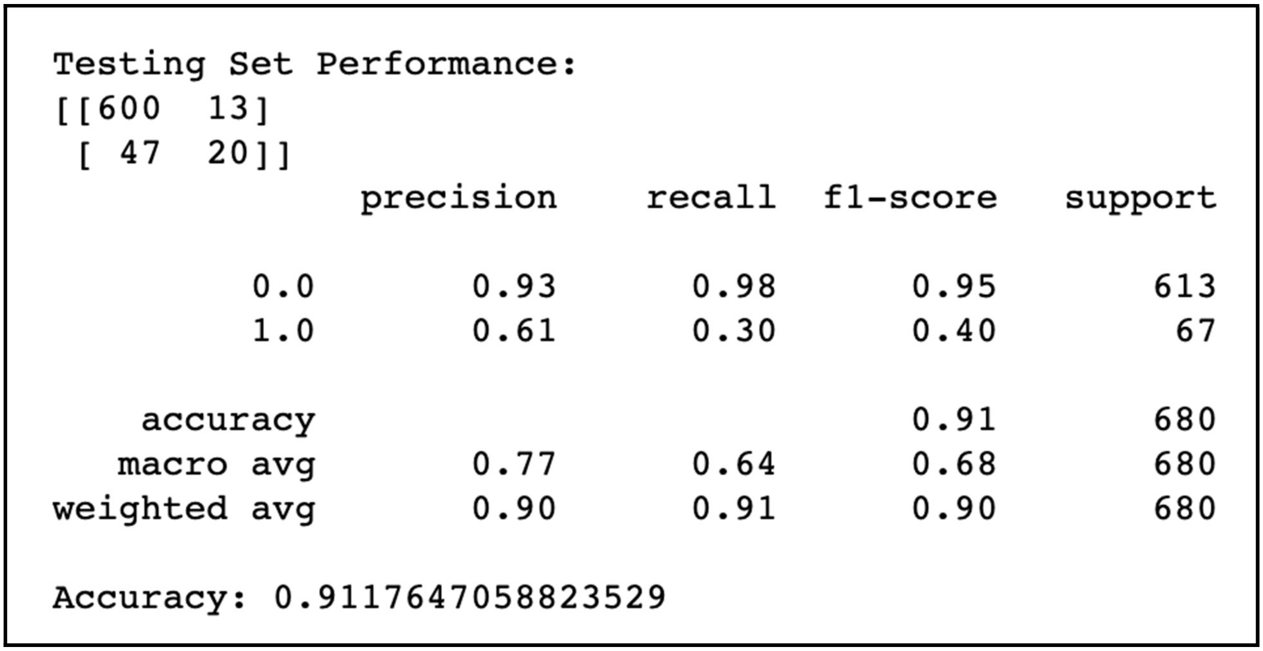
Random Forest Model Building:

- Import the Random Forest classifier from scikit-learn.
- Initialize the Random Forest classifier with desired hyperparameters (we can adjust these hyperparameters based on our needs).
- Fit the Random Forest model to the training data.
- Model Evaluation:
- After fitting the model, we can evaluate its performance on both the training and testing datasets.
- we can use various metrics like confusion matrix, classification report, accuracy, precision, recall, and F1-score to assess the model's performance.

Tuning Hyperparameters (Optional):

- We could further improve the Random Forest model by tuning hyperparameters using techniques like grid search or random search. This can help optimize the model's performance.

8. Validate the Random Forest Model on the test Dataset and state the performance metrics. Also state interpretation from the model



```
Testing Set Performance:
[[600  13]
 [ 47  20]]
              precision    recall  f1-score   support

         0.0       0.93      0.98      0.95       613
         1.0       0.61      0.30      0.40        67

    accuracy                           0.91       680
   macro avg       0.77      0.64      0.68       680
weighted avg       0.90      0.91      0.90       680

Accuracy: 0.9117647058823529
```

Snippet 1. 9 Random Forest Model performance on test data

Interpretation:

- Accuracy: The Random Forest model achieved an accuracy of approximately 0.91 on the test dataset. This means that it correctly classified about 91% of the instances. While the accuracy is reasonably high, it's important to consider other metrics, especially due to the class imbalance.

Precision and Recall:

- For class 0 (Non-Default): The model achieved high precision (0.93) and recall (0.98), indicating that it correctly identified a high proportion of non-default instances while maintaining a low false positive rate.

- For class 1 (Default): The model achieved lower precision (0.61) and recall (0.30) for default instances. This suggests that the model has difficulty correctly classifying instances of default. It has a higher false negative rate for class 1, indicating that it misses some default cases.

- F1-Score: The F1-score is a balanced measure that considers both precision and recall. For class 0, the F1-score is high (0.95), but for class 1, it is lower (0.40), reflecting the imbalance in class distribution.

- Macro and Weighted Averages: The macro-average and weighted-average values provide an overall summary. The macro-average is lower due to the lower performance on class 1, while the weighted-average considers the class distribution and is higher due to the dominance of class 0.

- Class Imbalance Impact: The lower performance on class 1 (Default) can be attributed to the class imbalance, where there are significantly more instances of class 0 (Non-Default) than class 1. The model is biased toward the majority class and may require techniques like oversampling or changing the decision threshold to improve performance on the minority class.

In summary, while the Random Forest model performs well in terms of accuracy and precision for the majority class, it struggles with recall and F1-score for the minority class (Default).

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

Linear Discriminant Analysis (LDA) is a dimensionality reduction and classification technique that aims to find a linear combination of features that maximizes the separation between different classes. Here's a summary of the approach taken to build the LDA model:

- Model Initialization: The LDA model is initialized using LinearDiscriminantAnalysis() from scikit-learn.

- Model Fitting: The LDA model is trained on the training data using the .fit() method.

- Model Prediction: The trained model is used to make predictions on the test dataset.

- Performance Evaluation: The model's performance is assessed using classification metrics like precision, recall, and F1-score, which provide insights into how well the model can classify instances into the target classes.

- Interpretation: The classification report summarizes the model's performance, including metrics for both classes (default and non-default). It helps assess the model's accuracy and its ability to correctly classify instances of each class.

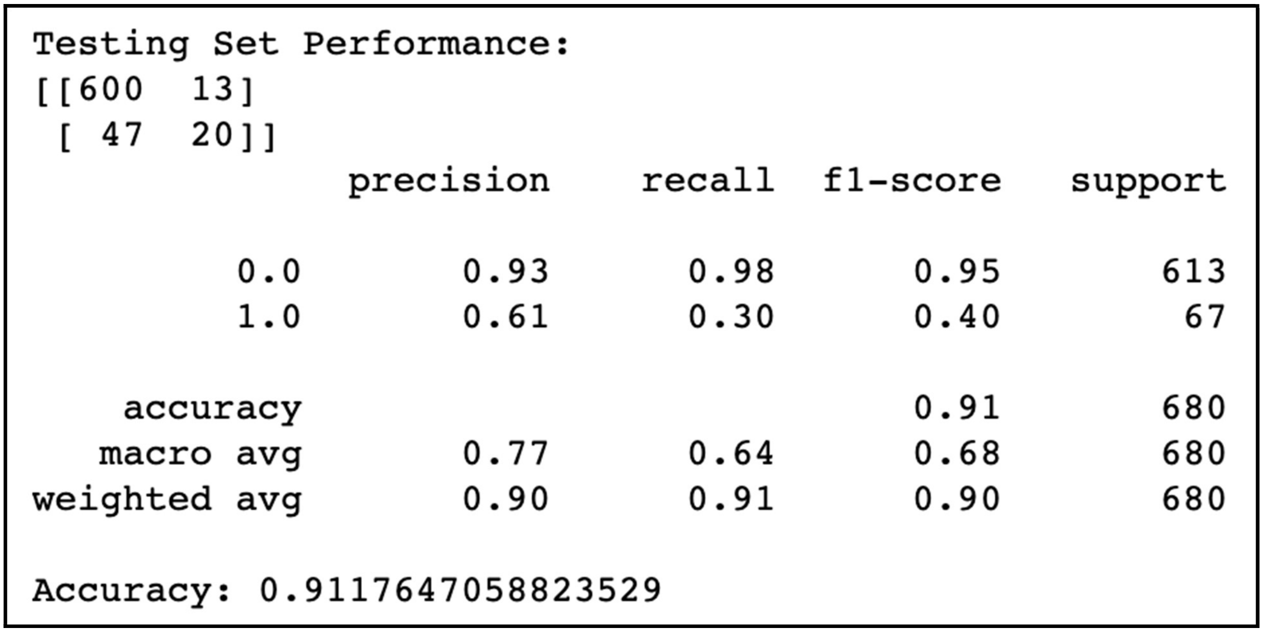
- Iterative Process: Model building and evaluation can be iterative. We may fine-tune hyperparameters or preprocess features based on the initial results to improve model performance.

- Class Imbalance Handling: If there is class imbalance in the dataset (e.g., more non-default cases than default cases), consider techniques like oversampling, under sampling, or setting class weights to address this issue.

- Model Deployment: Once satisfied with the model's performance, it can be deployed for making predictions on new, unseen data.

This approach ensures that the LDA model is built, evaluated, and interpreted effectively, providing valuable insights into its classification capabilities.

10. Validate the LDA Model on test Dataset and state the performance metrics. Also state interpretation from the model



```
Testing Set Performance:
[[600  13]
 [ 47  20]]
              precision    recall  f1-score   support

         0.0       0.93      0.98      0.95       613
         1.0       0.61      0.30      0.40        67

    accuracy                           0.91       680
   macro avg       0.77      0.64      0.68       680
weighted avg       0.90      0.91      0.90       680

Accuracy: 0.9117647058823529
```

Snippet 1. 10 LDA performance on test data

Interpretation:

- Accuracy: The Random Forest model achieved an accuracy of approximately 0.91 on the test dataset.
- This means that it correctly classified about 91% of the instances. While the accuracy is reasonably high, it's important to consider other metrics, especially due to the class imbalance.

Precision and Recall:

For class 0 (Non-Default): The model achieved high precision (0.93) and recall (0.98), indicating that it correctly identified a high proportion of non-default instances while maintaining a low false positive rate.

- For class 1 (Default): The model achieved lower precision (0.61) and recall (0.30) for default instances. This suggests that the model has difficulty correctly classifying instances of default. It has a higher false negative rate for class 1, indicating that it misses some default cases.

- F1-Score: The F1-score is a balanced measure that considers both precision and recall. For class 0, the F1-score is high (0.95), but for class 1, it is lower (0.40), reflecting the imbalance in class distribution.

- Macro and Weighted Averages: The macro-average and weighted-average values provide an overall summary. The macro-average is lower due to the lower performance on class 1, while the weighted-average considers the class distribution and is higher due to the dominance of class 0.

- Class Imbalance Impact: The lower performance on class 1 (Default) can be attributed to the class imbalance, where there are significantly more instances of class 0 (Non-Default) than class 1. The model is biased toward the majority class and may require techniques like oversampling or changing the decision threshold to improve performance on the minority class.

In summary, while the Random Forest model performs well in terms of accuracy and precision for the majority class, it struggles with recall and F1-score for the minority class (Default). Further efforts are needed to address class imbalance and improve the model's ability to correctly identify instances of default. Additionally, I’ve performed hyperparameter tuning to enhance predictive performance further.

11. Compare the performances of Logistic Regression, Random Forest, and LDA models (include ROC curve)

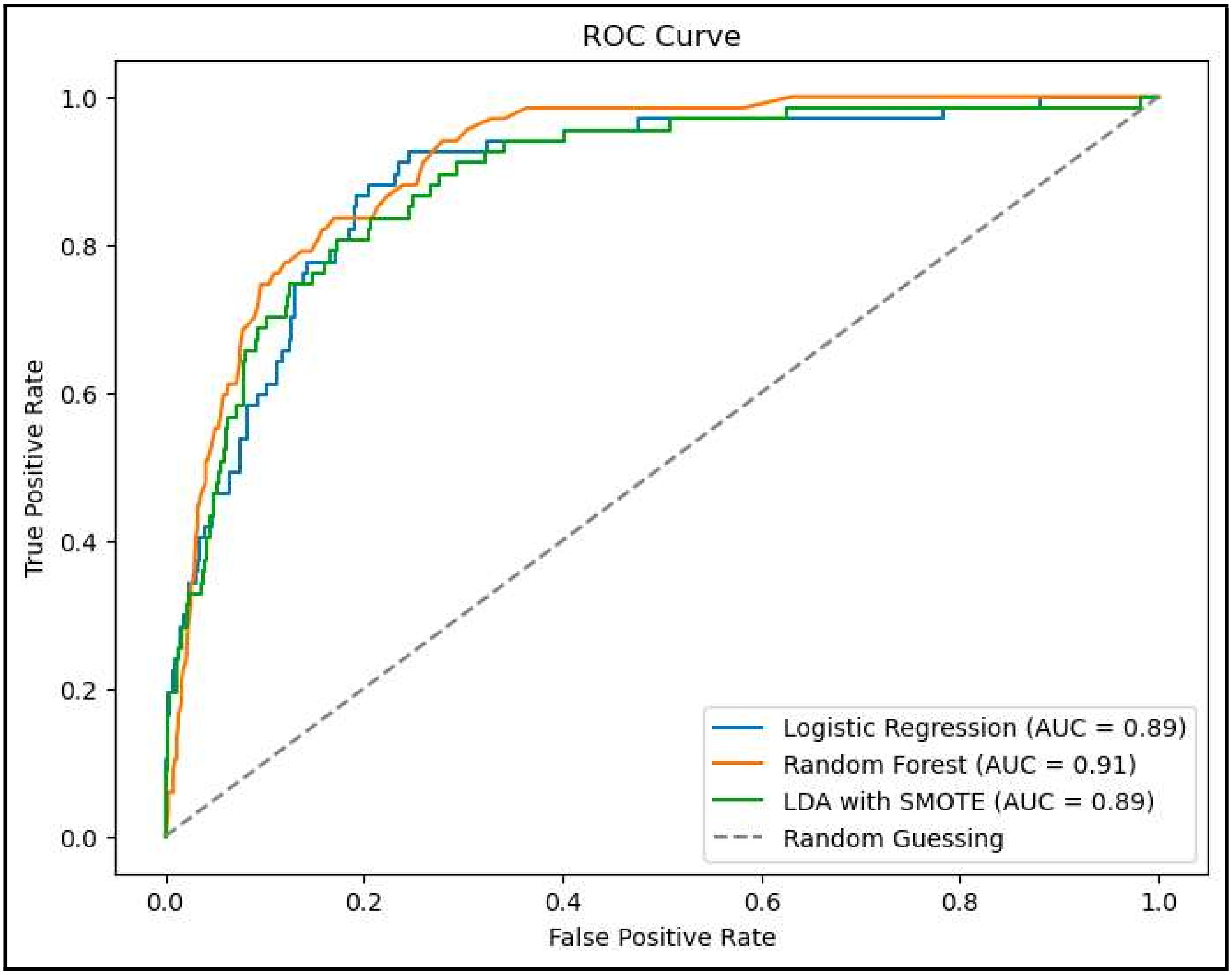


Figure 1. 7 ROC Curve

Performance Comparison:

Logistic Regression vs. Logistic Regression-SMOTE:

- The Logistic Regression model with SMOTE oversampling shows a substantial improvement in recall (0.76) for the minority class (Default = 1) compared to the non-SMOTE Logistic Regression model (0.42).

However, the trade-off is a decrease in precision (0.35) for the minority class in the SMOTE model, resulting in a lower F1-score (0.48). This suggests that the SMOTE model identifies more true positives but also increases false positives.

Random Forest vs. Random Forest - Grid Search:

- The Random Forest model and the one with hyperparameter tuning (Grid Search) have similar performance in terms of precision, recall, and F1-score. Grid Search does not provide a significant improvement in this case.

LDA vs. LDA-SMOTE:

- Similar to Logistic Regression, LDA with SMOTE significantly improves recall for the minority class (0.79) compared to the non-SMOTE LDA model (0.51).
- LDA-SMOTE also exhibits a higher F1-score (0.52) due to the better balance between precision and recall.

12. Conclusions and Recommendations

Conclusions:

Class Imbalance: Class imbalance in the dataset significantly affects model performance. SMOTE, as applied to Logistic Regression and LDA, helps address this issue by generating synthetic samples of the minority class. This boosts recall but might result in slightly lower precision.

Model Choice: The choice of the best model depends on the specific goals of the analysis. Logistic Regression and LDA are linear models, while Random Forest is a non-linear ensemble model. Consider the linearity assumption and complexity of the data when selecting a model.

Hyperparameter Tuning: In this case, hyperparameter tuning (Grid Search) did not yield a substantial improvement for the Random Forest model. It's essential to experiment with different hyperparameter values to identify the best settings for the specific dataset.

Recommendations:

Model Ensemble: We might consider ensembling the models to benefit from their individual strengths. For example, we can combine the Logistic Regression-SMOTE and LDA-SMOTE models to improve overall classification performance.

Threshold Adjustment: Depending on the business context, we can adjust the classification threshold to optimize precision or recall based on the specific requirements for identifying default cases accurately or minimizing false positives.

Feature Engineering: Further explore feature engineering techniques to extract more relevant information from the data that can potentially enhance model performance.

Data Collection: Collecting more data, especially for the minority class, can help improve model generalization and performance.

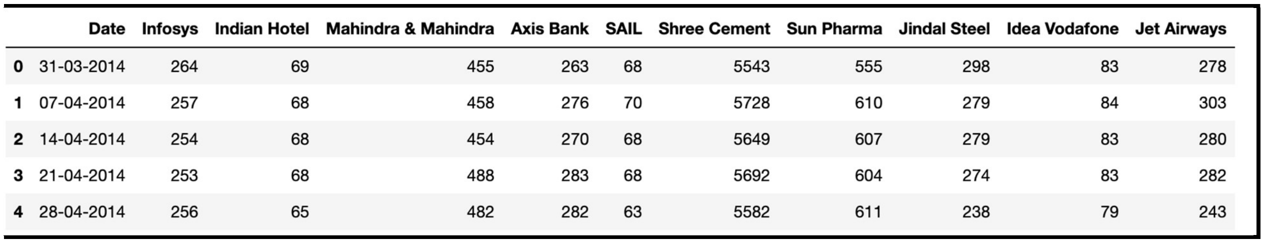
Continuous Monitoring: Keep in mind that the data distribution might change over time. Continuously monitor model performance and retrain it as needed to adapt to new patterns.

In summary, addressing class imbalance and carefully selecting models and hyperparameters are crucial steps in improving the predictive performance of models for default prediction. The choice of the best approach should consider the specific objectives and constraints of business problem.

PART B

Problem Statement:

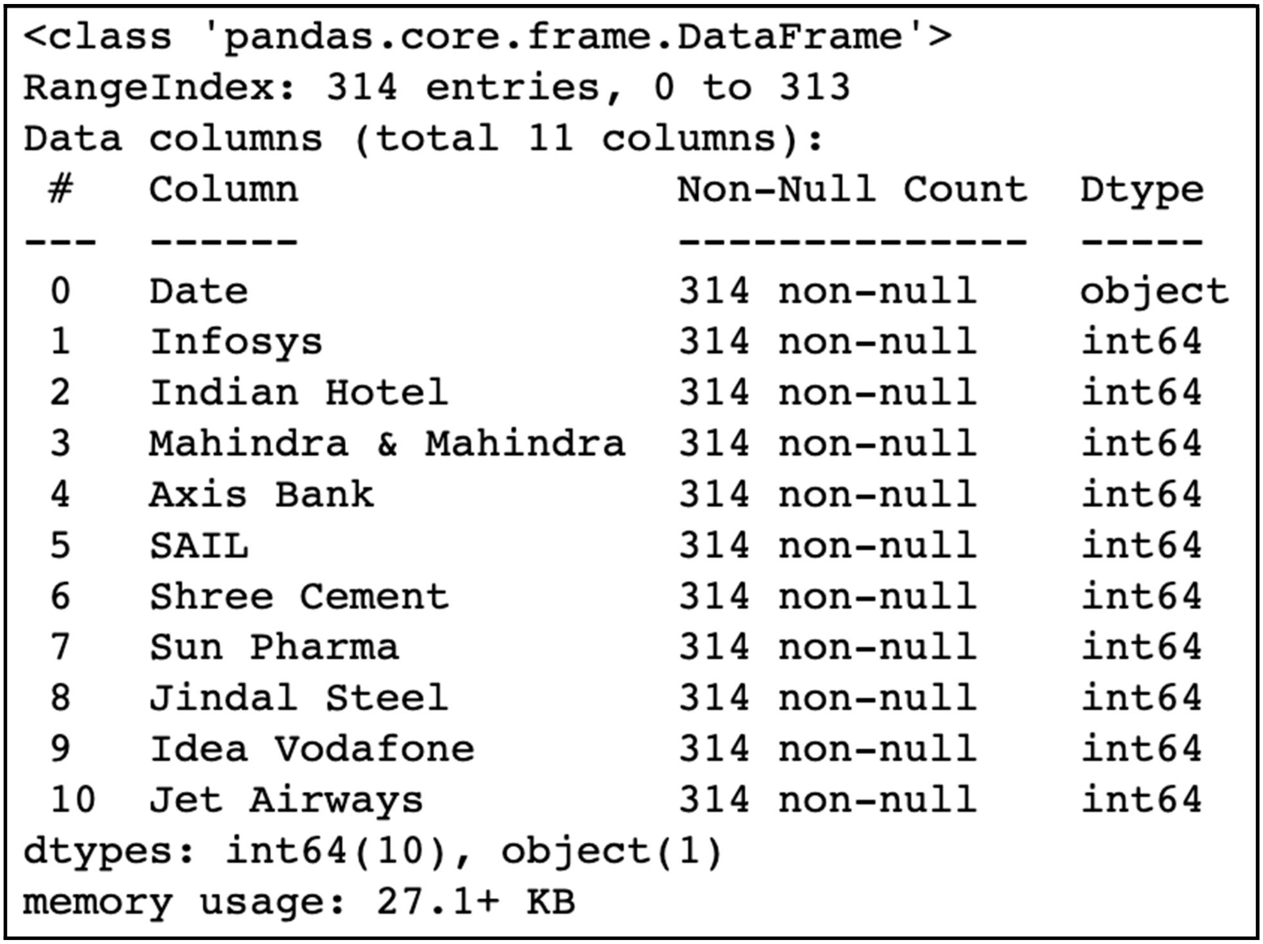
The dataset contains 6 years of information(weekly stock information) on the stock prices of 10 different Indian Stocks. Calculate the mean and standard deviation on the stock returns and share insights. You are expected to do the Market Risk Analysis using Python.



	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

Snippet 2. 1 Head of the dataset

The above is the preview of the dataset



```
<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
```

Snippet 2. 2 data info

We can observe there are no null values from the above dataset and about 10 attributes are present in the dataset

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

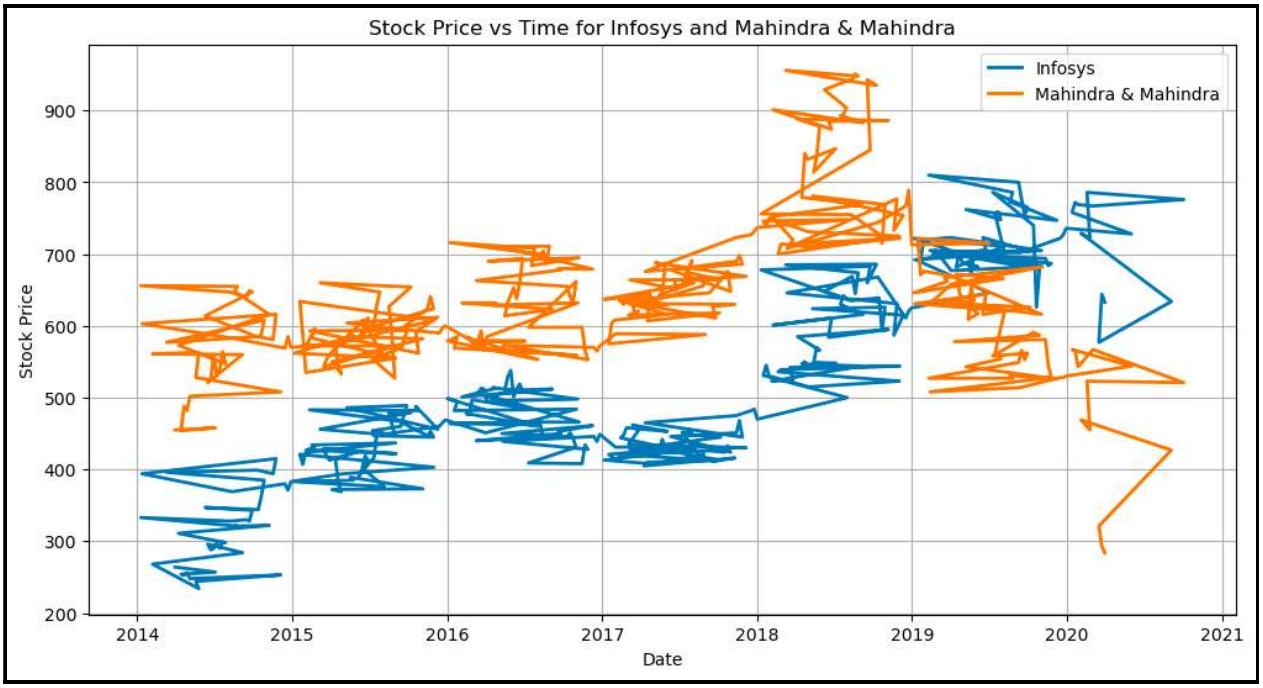


Figure 2. 1 Graphical representation of the stock price over years

Calculate Returns for all stocks with inference

Now that we have calculated the returns for each stock, we can perform various analyses and gain insights from this data:

- Direction of Returns: Look at the signs (positive or negative) of the returns to understand the direction of price changes for each stock. Positive returns indicate an increase in price, while negative returns indicate a decrease.

- Magnitude of Returns: Analyze the magnitude of the returns to identify stocks with larger price fluctuations (higher volatility) and those with more stable prices (lower volatility).

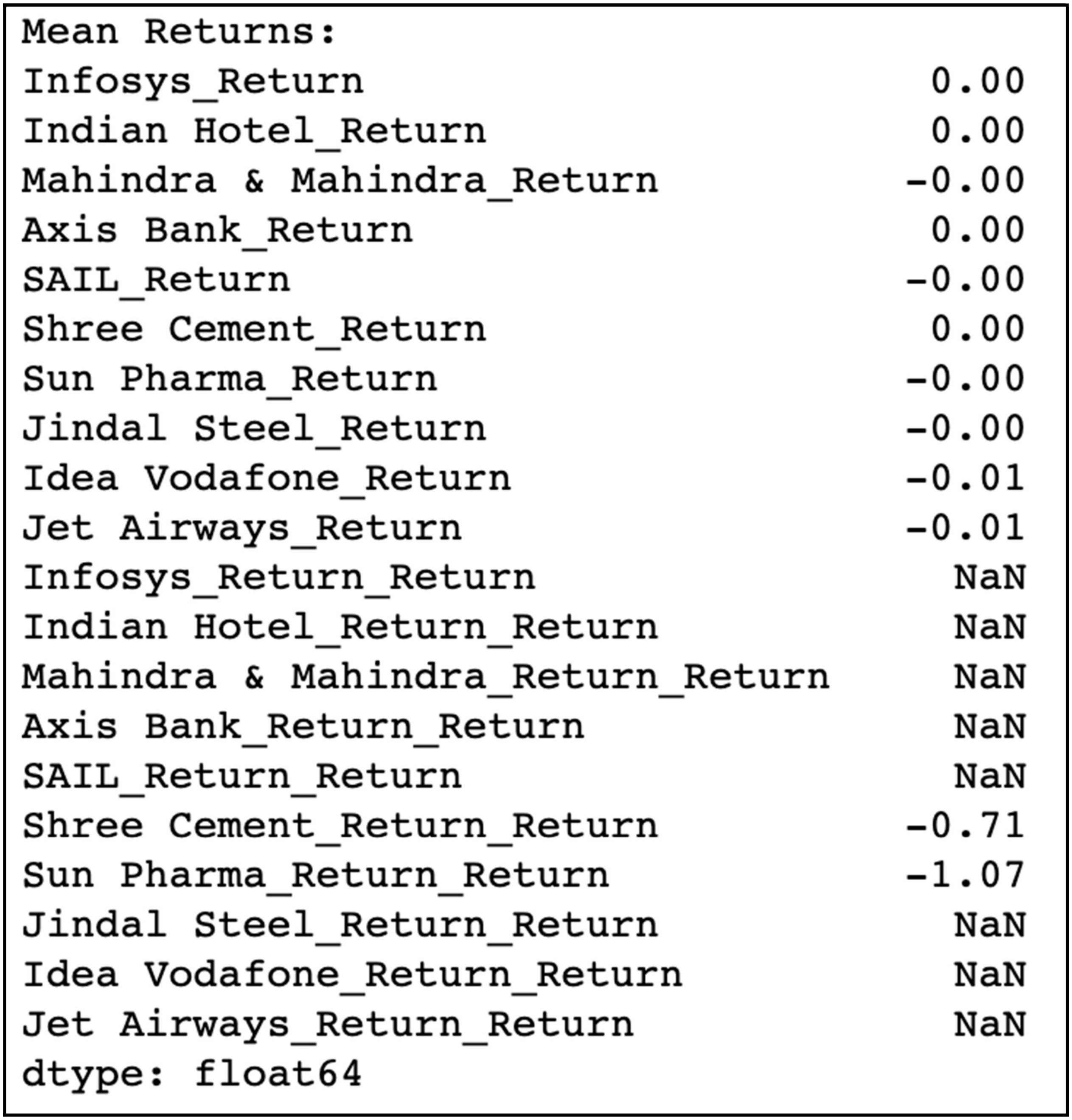
- Risk Assessment: Calculate the standard deviation of returns for each stock. Stocks with higher standard deviations are generally riskier as they exhibit more price volatility.

- Performance Comparison: Calculate the mean returns for each stock to compare their historical performance. Stocks with higher mean returns have, on average, provided better returns to investors.

- Correlation Analysis: Examine the correlations between the returns of different stocks. Positive correlations suggest that two stocks tend to move in the same direction, while negative correlations suggest they move in opposite directions. This can help with portfolio diversification.

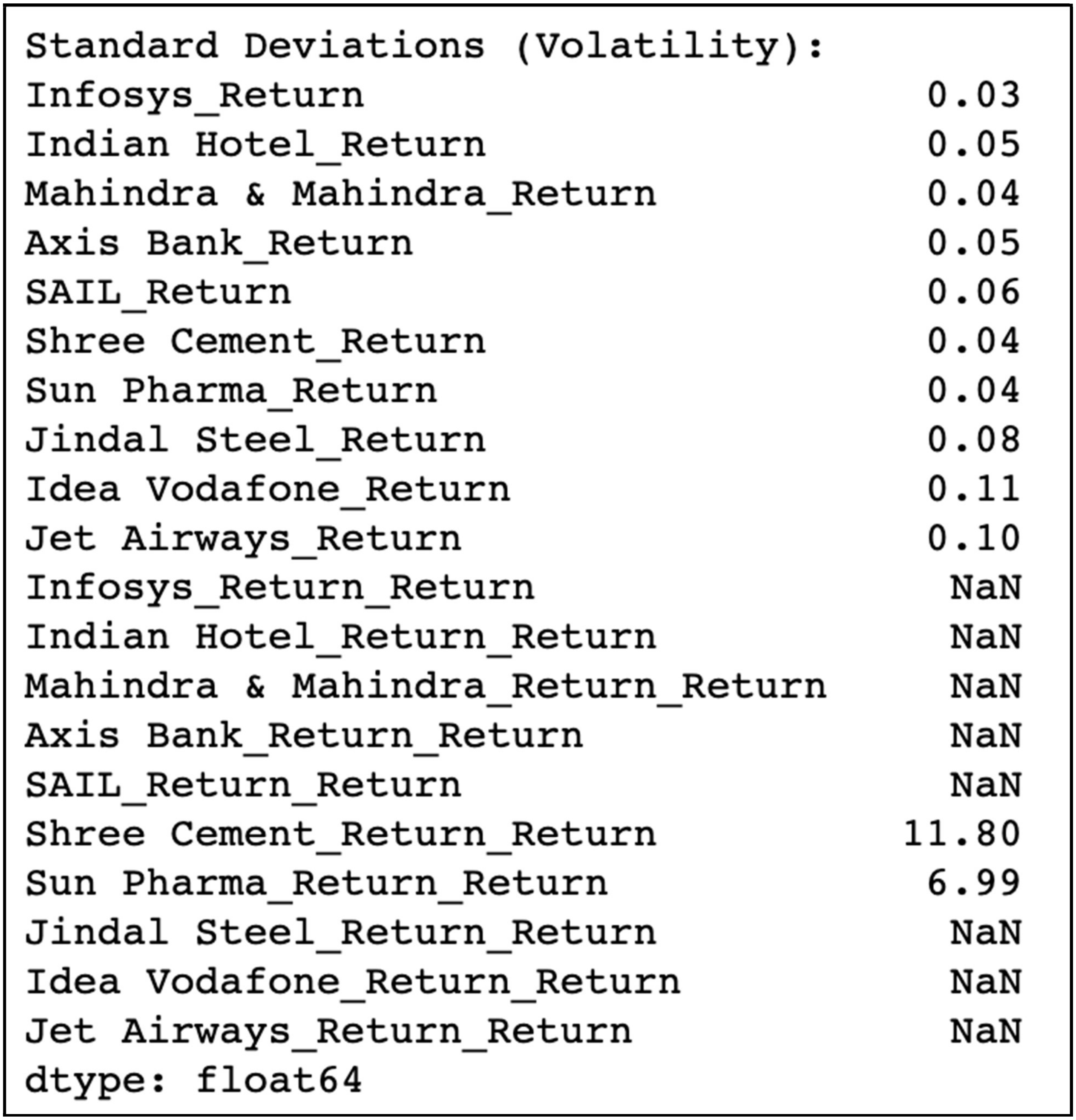
- Portfolio Allocation: Based on the risk tolerance and investment goals, we can use the return and risk metrics to allocate the portfolio among these stocks.

2. Calculate Stock Means and Standard Deviation for all stocks with inference



```
Mean Returns:
Infosys_Return                             0.00
Indian Hotel_Return                        0.00
Mahindra & Mahindra_Return                -0.00
Axis Bank_Return                           0.00
SAIL_Return                               -0.00
Shree Cement_Return                        0.00
Sun Pharma_Return                         -0.00
Jindal Steel_Return                       -0.00
Idea Vodafone_Return                      -0.01
Jet Airways_Return                        -0.01
Infosys_Return_Return                       NaN
Indian Hotel_Return_Return                  NaN
Mahindra & Mahindra_Return_Return           NaN
Axis Bank_Return_Return                     NaN
SAIL_Return_Return                          NaN
Shree Cement_Return_Return                -0.71
Sun Pharma_Return_Return                  -1.07
Jindal Steel_Return_Return                  NaN
Idea Vodafone_Return_Return                 NaN
Jet Airways_Return_Return                   NaN
dtype: float64
```

Snippet 2. 3 Means Returns



```
Standard Deviations (Volatility):
Infosys_Return                       0.03
Indian Hotel_Return                  0.05
Mahindra & Mahindra_Return           0.04
Axis Bank_Return                     0.05
SAIL_Return                          0.06
Shree Cement_Return                  0.04
Sun Pharma_Return                    0.04
Jindal Steel_Return                  0.08
Idea Vodafone_Return                 0.11
Jet Airways_Return                   0.10
Infosys_Return_Return                 NaN
Indian Hotel_Return_Return            NaN
Mahindra & Mahindra_Return_Return     NaN
Axis Bank_Return_Return               NaN
SAIL_Return_Return                    NaN
Shree Cement_Return_Return          11.80
Sun Pharma_Return_Return             6.99
Jindal Steel_Return_Return            NaN
Idea Vodafone_Return_Return           NaN
Jet Airways_Return_Return             NaN
dtype: float64
```

Snippet 2. 4 Standard Deviation Returns

Mean Returns:

- Close to Zero Mean Returns: Most of the stocks have mean returns close to zero (around 0.00). This suggests that, on average, these stocks did not exhibit a strong upward or downward trend over the analyzed period.

- Negative Mean Returns: Both Idea Vodafone and Jet Airways have negative mean returns (-0.01). This indicates that, on average, these stocks experienced a slight decrease in value over the period.

- Infosys_Return_Return and Other NaN Values: It seems there are some columns with NaN values, such as "Infosys_Return_Return." These NaN values might occur due to the way returns are calculated. We should investigate why these columns have missing values.

Standard Deviations (Volatility):

- Low Volatility: Most of the stocks have relatively low standard deviations, indicating lower volatility in their returns. Stocks like Infosys, Mahindra & Mahindra, and Shree Cement have low volatility, which suggests that their returns have been relatively stable over time.

- Higher Volatility: Idea Vodafone and Jet Airways have higher standard deviations (0.11 and 0.10, respectively), indicating higher volatility. These stocks have experienced more significant price fluctuations, which can be riskier for investors.

- Extreme Value: The "Shree Cement_Return_Return" standard deviation is very high (11.80). This might be an outlier or a data issue. It's important to verify the data and consider whether this value is accurate.

- NaN Values in Standard Deviations: Similar to mean returns, there are NaN values in the standard deviations. Investigate why these values are missing.

In summary, based on the mean returns and standard deviations, we can make the following observations:

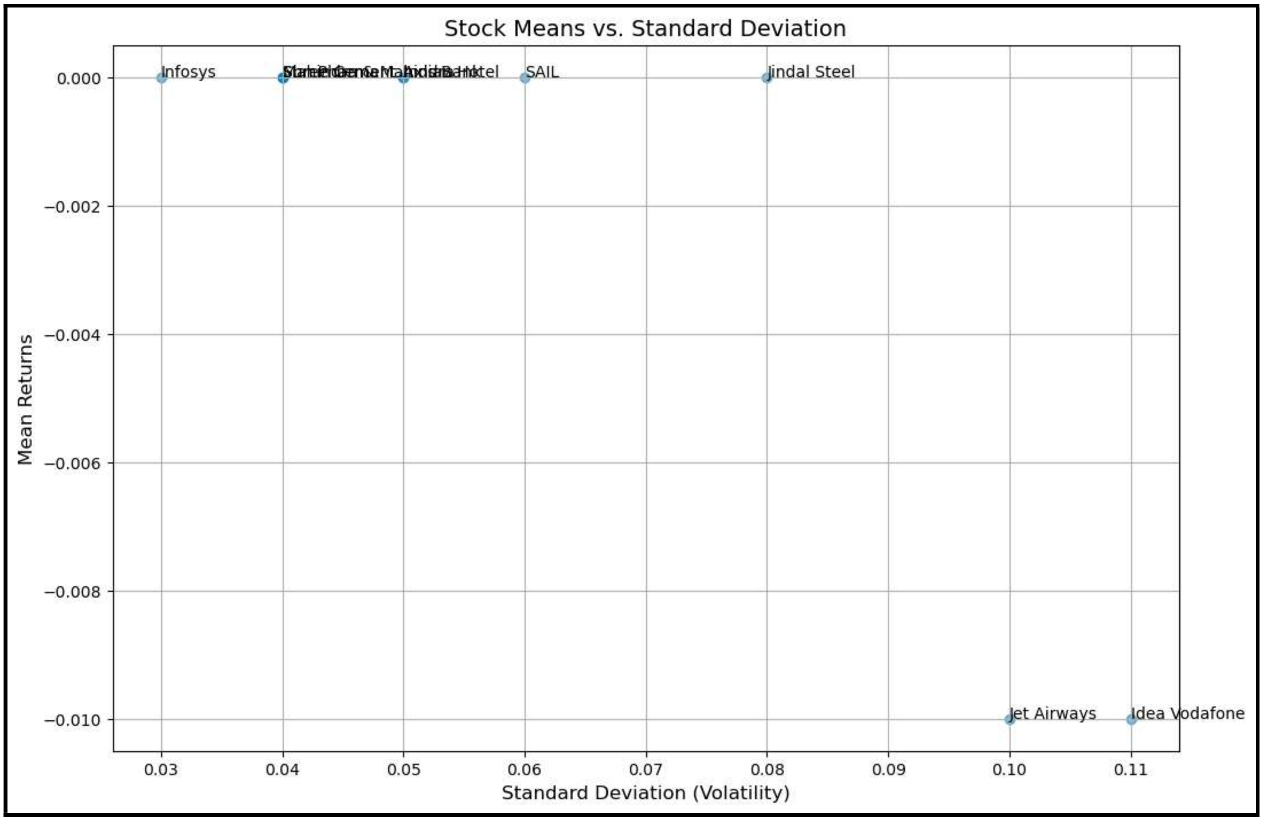
- Most stocks have exhibited relatively stable returns with mean returns close to zero.

- Idea Vodafone and Jet Airways have shown slightly negative mean returns and higher volatility, indicating some degree of risk associated with these stocks.

- There may be data issues or outliers in the "Shree Cement_Return_Return" column that should be examined.

- It's crucial to conduct further analysis, including correlation analysis and considering other factors like market conditions and news, to make informed investment decisions.

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



Snippet 2. 5 Mean Vs SD

Inference based on the result

- The plot illustrates the risk-return trade-off. Stocks with higher standard deviations (volatility) generally have the potential for higher mean returns, while stocks with lower volatility tend to have lower mean returns.

- "Idea Vodafone" and "Jet Airways" appear to be outliers, with higher volatility (standard deviation) and negative mean returns. These stocks carry higher risk and have underperformed during the analyzed period.

- "Shree Cement" and "Sun Pharma" stand out with very low volatility (standard deviation close to 0.04) and near-zero mean returns. These stocks are relatively stable but have limited potential for significant returns.

- "Idea Vodafone" has the highest volatility (0.11) among the stocks, indicating substantial price fluctuations, but it has negative mean returns, suggesting poor performance during this period.

- "Jet Airways" also exhibits high volatility (0.10) and negative mean returns, making it a high-risk, low-return stock.

- This analysis provides a snapshot of risk and return characteristics for the specified stocks during the analyzed period. Investors should consider their risk tolerance and investment objectives when selecting stocks for their portfolios. Diversification and risk management strategies are essential for managing a portfolio with a mix of risk profiles.

4. Conclusions and Recommendations

Conclusions:

- Risk-Return Trade-off: The analysis confirms the well-known risk-return trade-off in finance. Stocks with higher standard deviations (volatility) tend to have the potential for higher mean returns, while stocks with lower volatility typically have lower mean returns.

- Outliers: "Idea Vodafone" and "Jet Airways" are identified as outliers. They exhibit high volatility and negative mean returns, indicating that these stocks are riskier and have underperformed during the analyzed period. Investors should exercise caution when considering these stocks for investment.

- Stability vs. Growth: Stocks like "Shree Cement" and "Sun Pharma" have low volatility and near-zero mean returns, suggesting stability but limited potential for significant returns. These stocks may be suitable for conservative investors looking for stability in their portfolios.

Recommendations:

Diversification: Diversifying a portfolio across different stocks with varying risk-return profiles can help manage overall risk. Balancing high-volatility stocks with low-volatility ones can reduce the overall risk exposure of a portfolio.

Risk Management: Given the high volatility and negative mean returns of "Idea Vodafone" and "Jet Airways," consider these stocks carefully and assess risk tolerance before investing in them. It may be advisable to limit exposure to such high-risk stocks.

Long-term Perspective: While the analysis provides insights into historical performance, remember that past performance does not guarantee future results. Investors should take a long-term perspective and consider factors like company fundamentals and industry trends when making investment decisions.

Professional Advice: If it is unsure about the investment choices or risk tolerance, it's advisable to consult with a financial advisor or investment professional who can provide personalized guidance based on financial goals and risk tolerance.

Continuous Monitoring: The stock market is dynamic, and stock performance can change over time. Regularly monitoring investments and be prepared to adjust the portfolio as needed to align with the financial objectives.

Diversified Portfolio: Consider building a diversified portfolio that includes a mix of asset classes, such as stocks, bonds, and other investments, to spread risk and potentially enhance returns.