FINAL PROJECT

Pairs Trading using Machine Learning Model

- Sathya Balasubramani

The Jupyter notebook has been uploaded along with this document for your review at your convenience.

Table of Contents:

1. Pairs Trading:

Pairs trading is an advanced investment strategy that hinges on exploiting the relationship between two correlated assets. The core idea behind pairs trading is to identify pairs of securities that tend to move together in price over time. Traders then take advantage of temporary divergences from this historical relationship by simultaneously taking long and short positions in the assets. For example, if one asset in the pair experiences a price increase while the other lags, traders may buy the underperforming asset and sell short the outperforming one, anticipating that the prices will eventually converge back to their historical relationship.

Successful pairs trading relies on thorough statistical analysis and robust quantitative modeling to identify suitable pairs and determine optimal entry and exit points. This often involves calculating correlation coefficients, establishing mean-reverting relationships, and implementing risk management strategies to mitigate potential losses. While pairs trading offers the potential for consistent profits and can serve as a hedge against overall market movements, it requires diligent monitoring and adaptation to changing market conditions. Traders must be prepared to adjust their positions as deviations from historical correlations occur and employ effective risk management techniques to safeguard their capital. Overall, pairs trading represents a sophisticated approach favored by quantitative

traders and investors seeking to capitalize on short-term mispricings in correlated assets while managing risk effectively.

2. Selection of Pairs:

As discussed for the project, I intend to focus on pairs trading within the Indian banking sector, particularly targeting the top four components of the NIFTY Bank index: State Bank of India (SBI), HDFC Bank, ICICI Bank, and Axis Bank. This strategy involves identifying pairs of these banking stocks and capitalizing on their price movements. By leveraging the correlations between these major players in the Indian banking industry, we aim to exploit temporary divergences from their historical relationships, ultimately seeking to generate profits from their convergence.

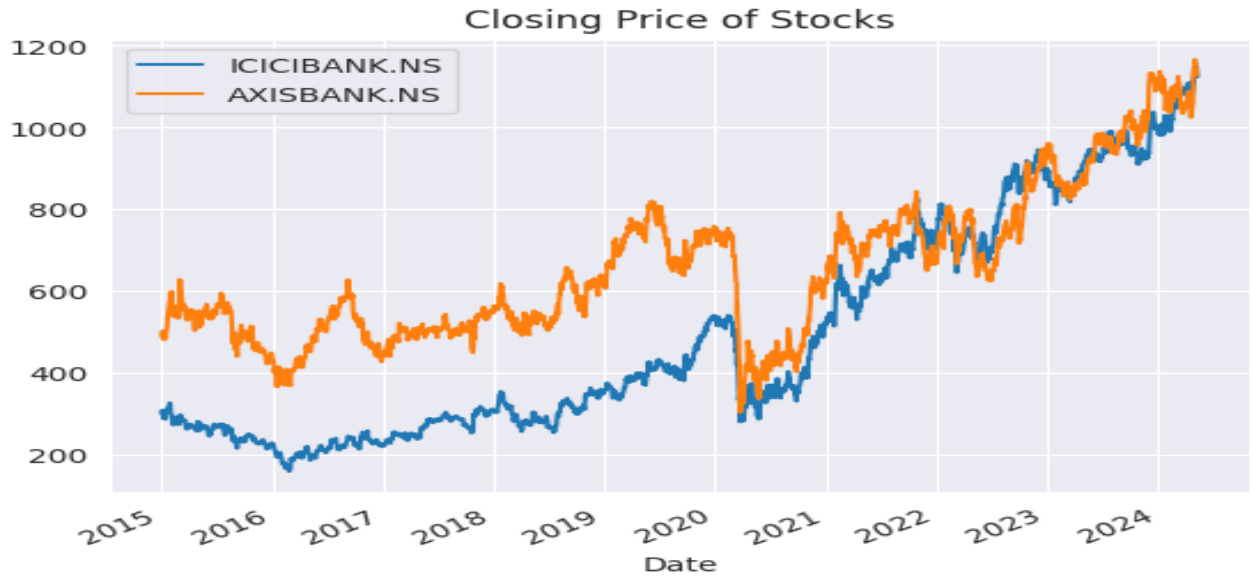
Cointegration and the Dickey-Fuller test play crucial roles in pairs trading strategies for banking stocks like SBI, HDFC, ICICI, and Axis Bank.

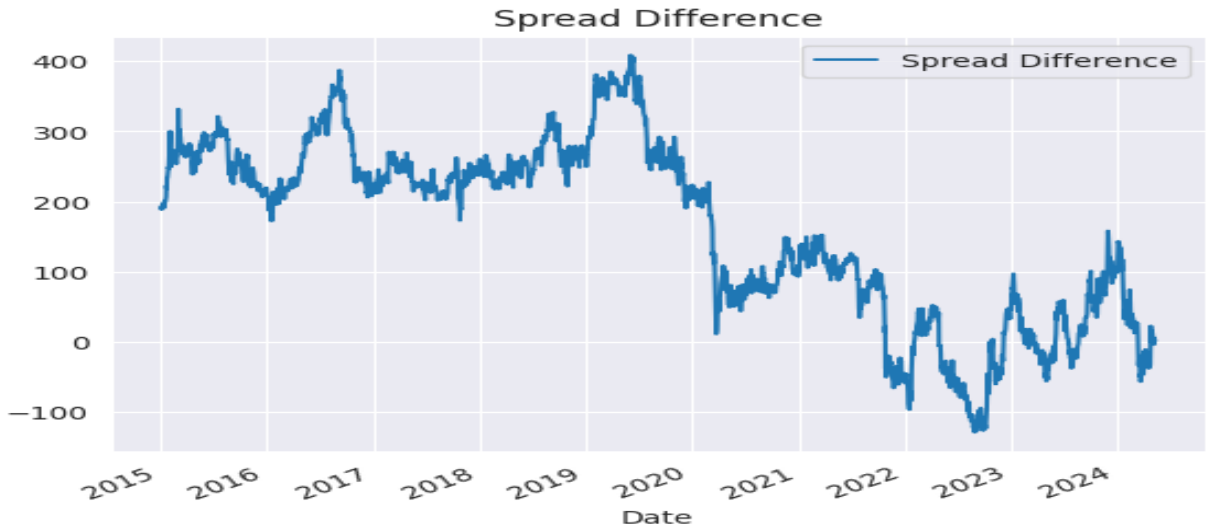
With a p-value of 0.03 from the Augmented Dickey-Fuller (ADF) test and an ADF statistic of -2.96, there is compelling statistical evidence against the null hypothesis of non-stationarity in the time series data. These findings suggest that the data may exhibit some degree of stationarity, a favorable characteristic for various statistical analyses. However, it's essential to consider the specific context and potential limitations of the data before drawing definitive conclusions.

Test	Values
ADF Statistic	**-2.96**
p-value	**0.03**

3. Data Analysis

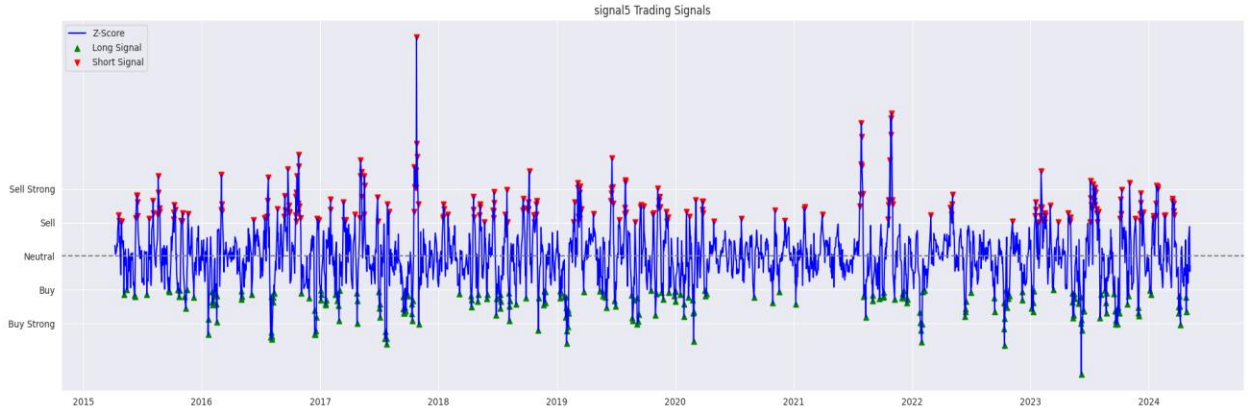
The plot displays the closing prices of stocks for ICICI Bank (ICICIBANK.NS) and Axis Bank (AXISBANK.NS) from 2015 to 2024. ICICI Bank's stock, represented by the blue line, shows a consistent upward trend throughout the period, starting from below 400 in 2015 and climbing to just under 1200 by 2024. Axis Bank's stock, depicted in orange, experiences more volatility with significant ups and downs.

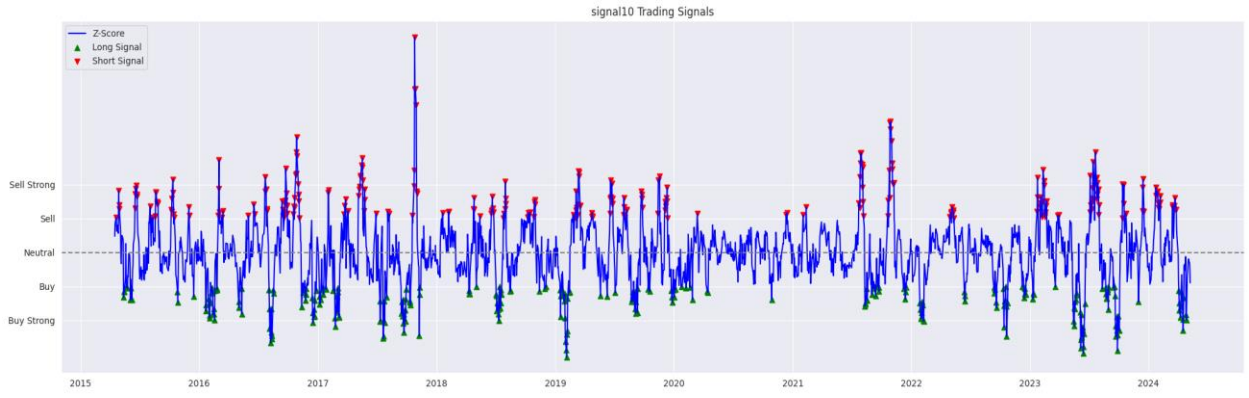


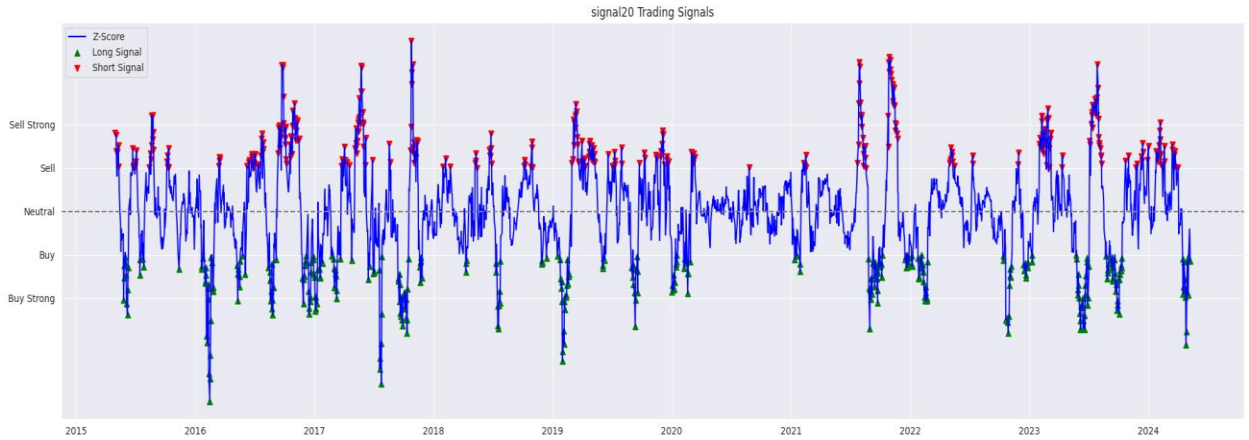


Analysis was conducted on the adjusted closing prices of ICICI and Axis Bank over the past nine years were plotted for reference. The analysis utilized pairs assignment methodology, involving tasks such as spread identification, spread normalization, inverse volatility determination, and signal generation matrix creation. Threshold parameters including longEntrythreshold, shortEntrythreshold, longcap, shortcap, and holding period were applied to generate long and short signals. Upon determining the positions, returns were calculated using the z-score of various days, and corresponding Sharpe ratios and equity graphs were computed and visualized.

The z-score differences for 5, 10, and 20 days of two stocks were normalized and plotted over the entire time series. Additionally, thresholds for longcap and shortcap were highlighted on these time series plots. You'll notice that as the period increases, the noise or volatility decreases. Various holding periods were experimented with, keeping all other variables constant, and found the optimal values of [-1,1,1,-1,16] (longEntryThreshold, longcap, shortEntryThreshold, shortcap and holding period]. I've followed the Pairs trading assignment closely for the data analysis and signal generation. A generalized python method was written to compute the signal generation array by inputting the data of z-score spreads of any days, caps, thresholds and holding periods.







3.1 Signal and Caps Counts:

Z-scores	Long Signals	Short Signals	LongCap	Short Cap
z-score-5	253	250	43	37
z-score-10	276	269	19	23
z-score-20	368	355	14	15

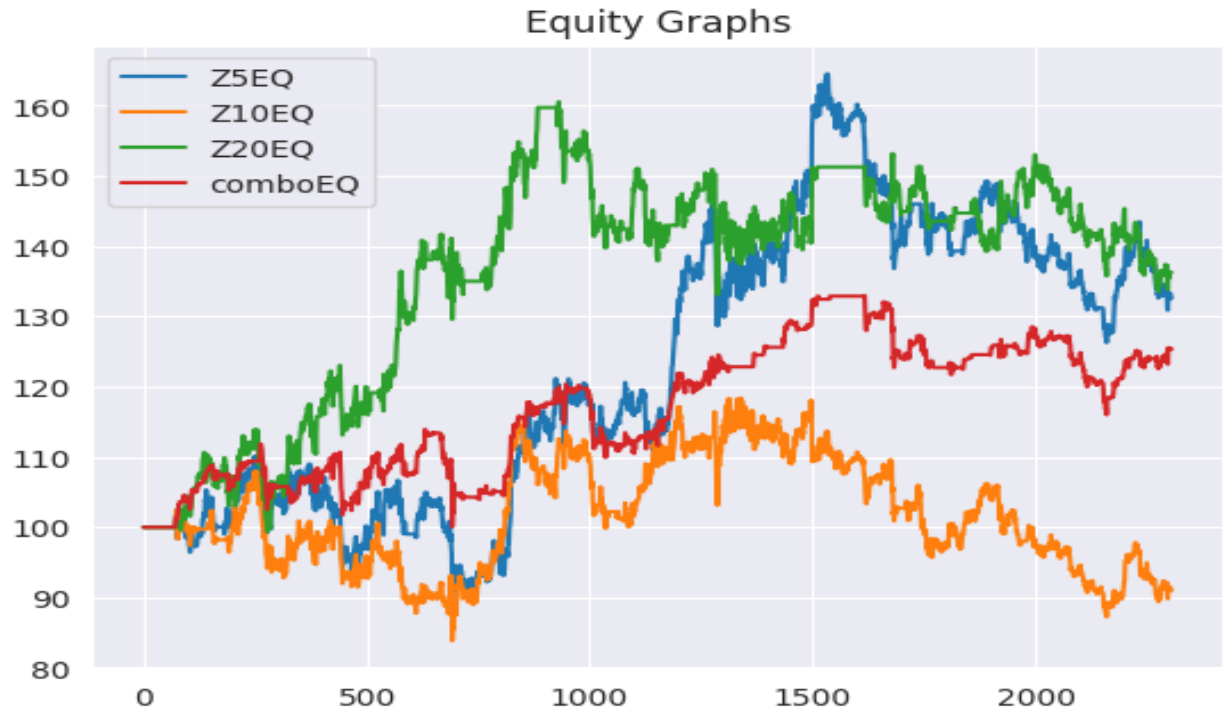
After completing the analysis, points for signal generation were marked on the z-score spreads of 5, 10, and 20 days. The y-axis was labeled with terms like neutral, buy (1-standard deviation) , strong buy (2 standard deviation), and corresponding sell signals.

Ultimately, for each z-score period (5, 10, and 20 days), the returns, equity graphs, and performance metrics were calculated and displayed in the table below.

3.2 Performance Metrics

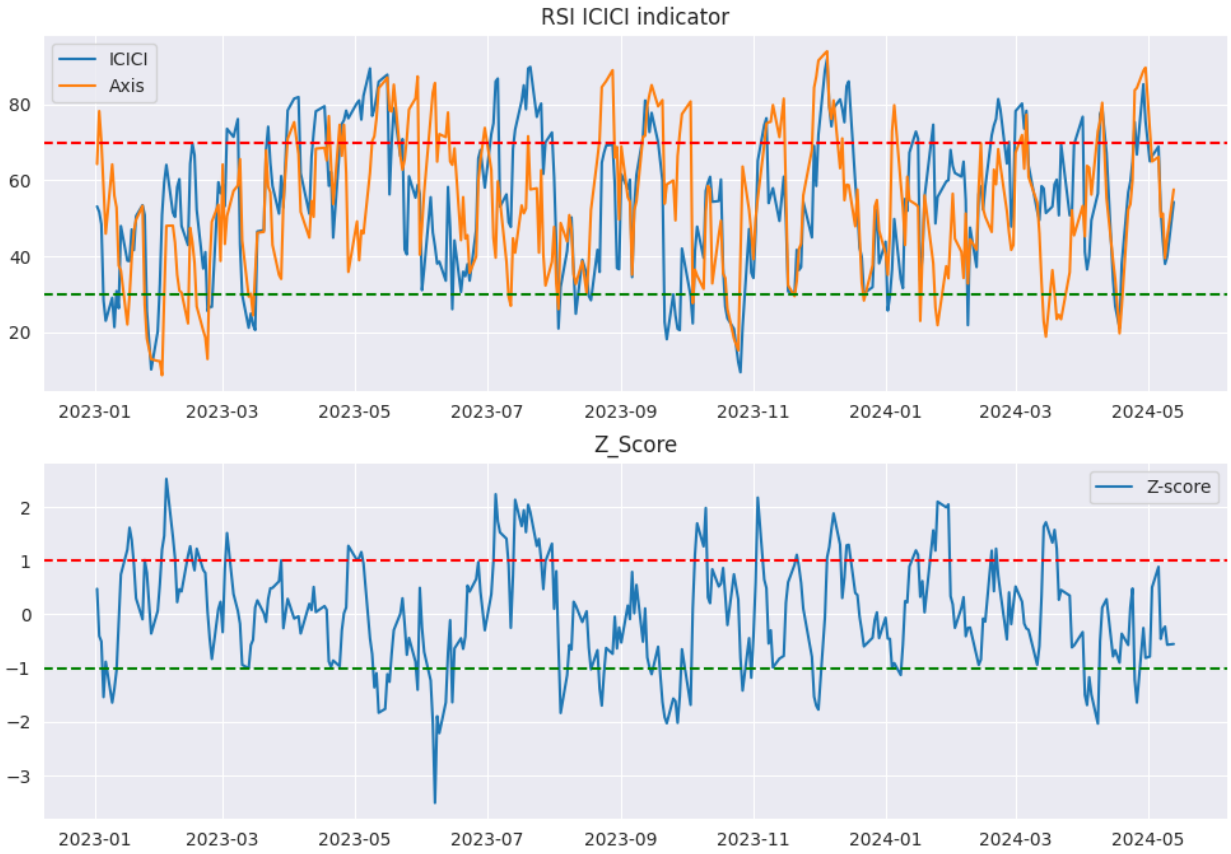
	Z_score_5_diff	Z_score_10_diff	Z_score_20_diff
Average Returns	0.000143	-0.000006	0.000189
Volatility	0.00571	0.0055	0.0057
Sharpe Ratio	**0.40**	-0.19	**0.52**
Cum Returns	0.34	-0.05	0.48

3.3 Equity Graphs:



Equity graphs for the z-scores over 5, 10, and 20 days were generated and displayed alongside graphs of combined z-scores. From these plots, it's evident that the 20-day z-score (Z20EQ) graph, which has a Sharpe ratio of 0.52, shows the highest performance compared to the combined z-score graphs. The graphs that combine different z-scores are more balanced compared to the individual z-score graphs.

3.4 Interpretation of **Z-score-spread-5 day** results along with RSI indicator:



I've plotted data from 2023 to 2024 to highlight trading opportunities for ICICI and Axis banks. The plot showcases the Relative Strength Index (RSI) and Z-score for both stocks within this period. The RSI for ICICI (blue line) and Axis (orange line) fluctuates between the overbought (70) and oversold (30) levels. ICICI frequently reaches overbought conditions, suggesting shorting opportunities, while Axis often hits oversold conditions, indicating potential buying opportunities. The Z-score-spread-5-day, which measures the standardized spread between the two stocks, hovers around zero but shows significant deviations. When the Z-score exceeds 1, ICICI is overperforming relative to Axis, recommending a strategy to short ICICI and buy Axis. Conversely, when the Z-score is below -1, ICICI is underperforming relative to Axis, suggesting a strategy to buy ICICI and short Axis. This combined analysis of RSI and Z-score provides a comprehensive approach to identifying trading opportunities based on the relative performance of these stocks.

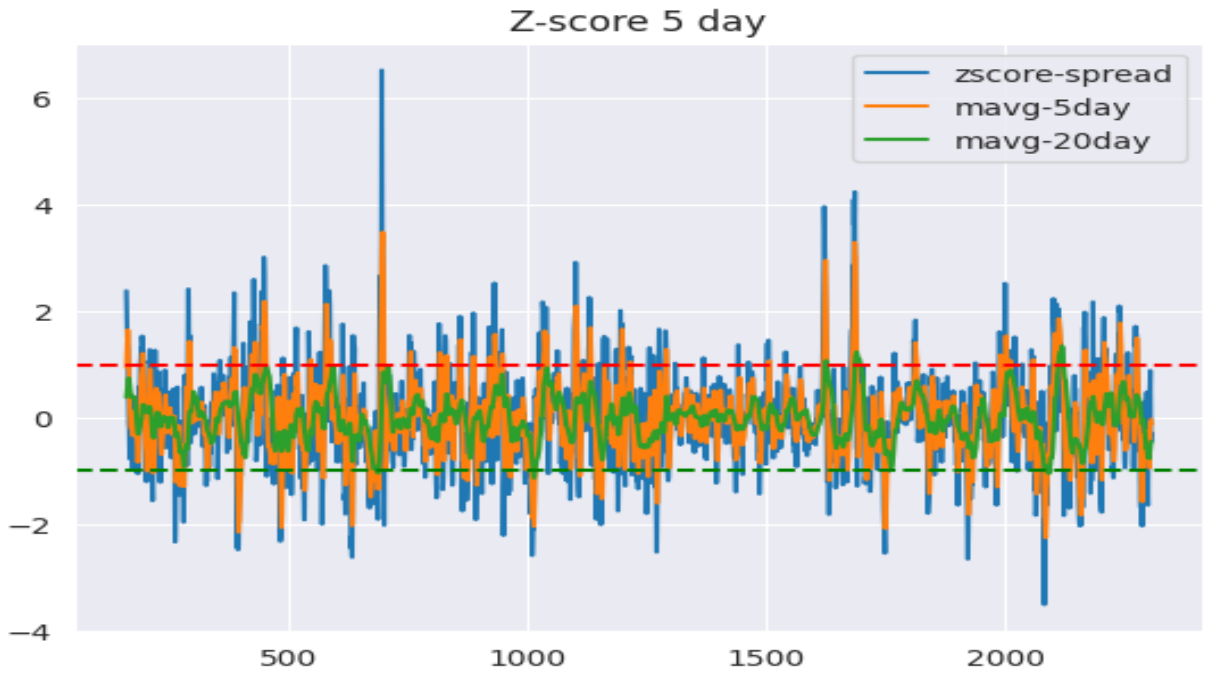
4. Choice of Machine Learning Model

The DecisionTreeClassifier is a popular machine learning algorithm used for classification tasks, where the goal is to predict the category of an instance based on its attributes. It constructs a decision tree by splitting the dataset into subsets based on the feature values that result in the largest information gain, aiming to have the purest possible subsets at

each node. This process continues recursively until a stopping criterion is met, such as reaching a maximum specified depth of the tree or achieving a minimum number of samples in a node. Decision trees are favored for their ease of interpretation and visualization, as each decision node in the tree represents a clear decision rule, making the logic of predictions easy to follow and understand.

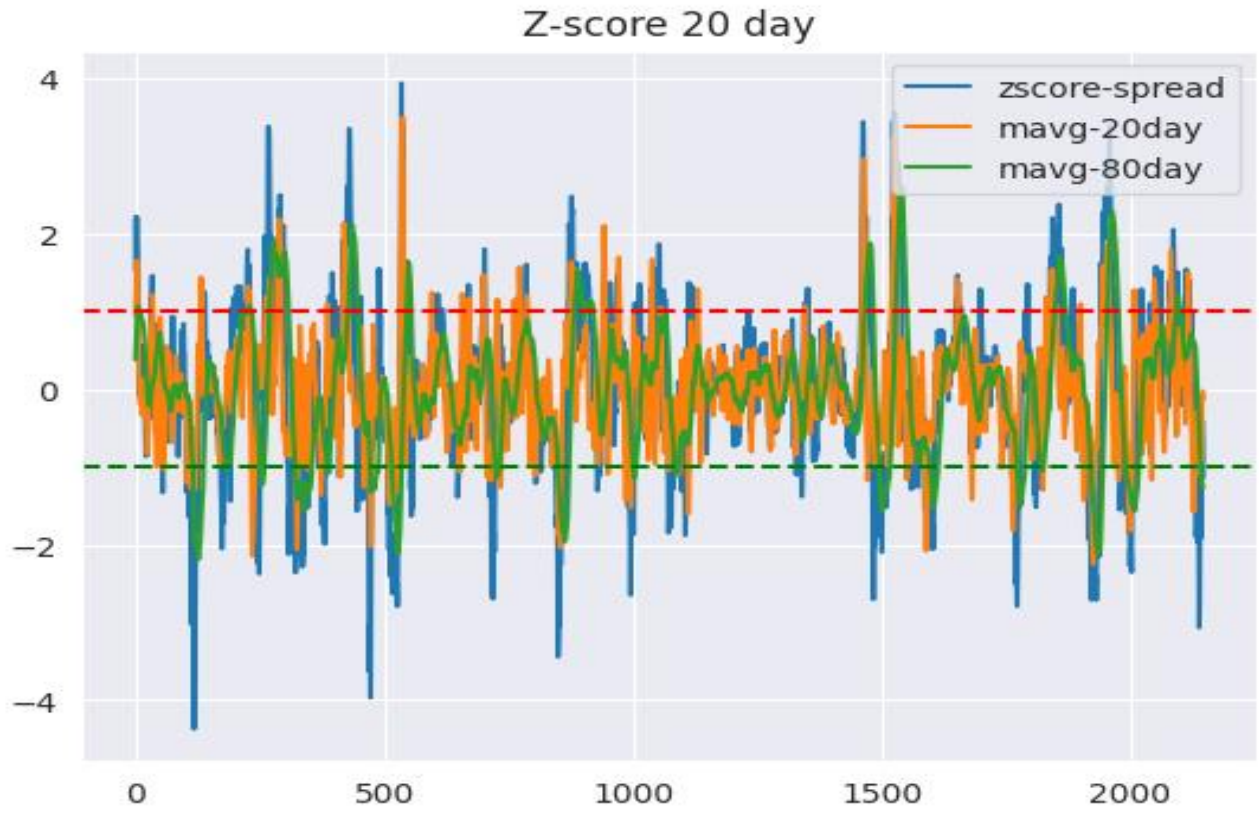
4.1 Features

I have selected and employed 9 features for the machine learning model, which include the normalized spread difference between two stocks, along with their 5-day and 20-day moving averages for the 5-day z-score model, and the 20-day and 80-day moving averages for the 20-day z-score model. Additionally, the model incorporates the standard deviation, returns, and means of ICICI and Axis Bank to enhance its predictive accuracy and reliability.



The provided plot displays three time series: the z-score of a 5-day spread in blue, a 5-day moving average in green, and a 20-day moving average in orange, plotted against a time axis spanning from 0 to over 2000+ days. The z-score spread shows notable volatility with sharp peaks reaching up to 6 and dips down to -4. In contrast, the moving averages exhibit smoother behaviors; the 5-day average oscillates primarily between -2 and 2, while the 20-day average maintains a steadier trend, closely hugging the zero line. Two horizontal

dashed red lines at y=0 and y=1 serve as reference points, indicating normal or threshold levels of interest.



4.2 Label Generation:

I used the signals generated from the analysis of pairs trading for the 5-day and 20-day z-score models. Since the goal is to predict these signals using machine learning model, the outputs from both the 5-day and 20-day z-score models are utilized as the target features (y-feature) in the model.

```
Y = train['Signal5_label']
X = train[['z_score_5_diff_x', 'zscore_diff_mavg_5', 'zscore_diff_mavg_20','std_20','ICICI_ret','AXIS_ret','ICIC_mean',
'AXIS_mean']]
```

4.3 Training, Validation and Test datasets:

Firstly, all the NaN values in any columns were removed to facilitate computation and learning. Then, 70% of the dataset was allocated for training, 15% was used for validation,

and the remaining 15% was used for testing. Z-score-10 day were dropped because it gave the significant lower sharpe ratio when compared to z-scores days.

```
train = data_mlF_clean[0:1667]
val = data_mlF_clean[1667:1917]
test = data_mlF_clean[1917:]
```

4.4 Model Setup

A DecisionTree Classifier machine learning model was used for classifying and predicting the target features. The hyperparameters were selected based on conducting a grid search to find the optimal values. After completing the grid search, the hyperparameters such as `max_depth = 10`, `min_samples_split = 12`, and `random_state = 1` were used to train the model. Once the model was trained, the same classifier, `clf`, was utilized to predict the validation and test datasets. A similar process was used for the z-score 20 model.

```
# Training data classifier
clf = DecisionTreeClassifier(random_state = 1, min_samples_split = 12, max_depth = 10)#, min_samples_split = 10)
clf.fit(X, Y)
print(confusion_matrix(Y, clf.predict(X)))
print(accuracy_score(Y, clf.predict(X)))
```

The confusion matrix and accuracy score for the z-score-5 and z-score-20 are presented below for the reference:

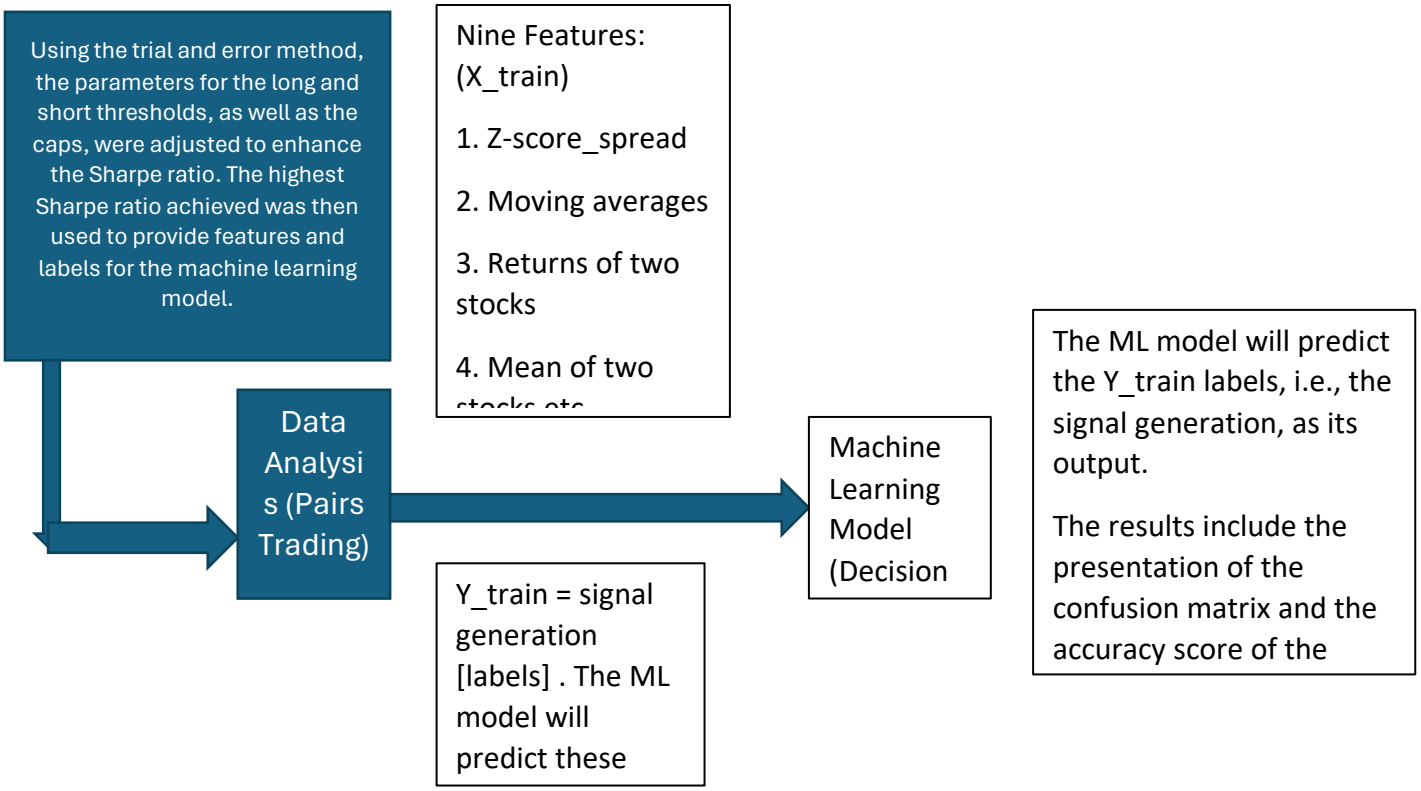
Accuracy, confusion matrix	Traning	Validation	Testing
z-score_5	**Confusion Matrix:** [[740 9 42] [17 76 12] [54 21 696]] **Accuracy Score:** 0.90	**Confusion Matrix:** [[63 6 15] [19 0 4] [28 9 106]] **Accuracy Score:** 0.67	**Confusion Matrix:** [[100 10 15] [1 3 11] [10 4 77]] **Accuracy Score:** 0.77

z-score-20	**Confusion Matrix:** [[734 29 6] [113 265 20] [28 17 455]] **Accuracy Score:** 0.87	**Confusion Matrix:** [[88 44 19] [32 8 2] [3 20 34]] **Accuracy Score:** 0.52	**Confusion Matrix:** [[108 11 6] [18 6 2] [0 1 79]] **Accuracy Score:** 0.83

Both models demonstrate strong performance on the training data, with the Z-Score 5 model achieving an accuracy of 0.90 and the Z-Score 20 model achieving 0.87. However, their performance significantly drops on the validation data, particularly for the Z-Score 20 model, which sees a decrease to an accuracy of 0.52. This suggests potential overfitting, as the models are not generalizing well to unseen data. The testing results are better than the validation results for both models, indicating that the validation set may not be fully representative or has specific challenges. Additionally, the second category consistently shows weaker performance across all datasets, implying that feature representation or model complexity may need adjustment to better capture the nuances of this category. Overall, the noticeable performance differences across datasets highlight the need for improvements to reduce overfitting and enhance model generalization.

5. Execution Flow:

The assignment focuses on pairs trading analysis, specifically generating signals for long and short entries to maximize investment returns, measured by the Sharpe ratio. Following the steps provided, I generated these signals and then set up a machine learning model using a DecisionTree classifier.



This model utilized nine parameters as features, with the signal generation columns serving as the target for predicting labels. The goal was to evaluate how accurately the machine learning model could predict signals compared to the original signal generation labels. After training the model, I used the trained classifier to predict signals on the test dataset. The results of this prediction are presented in the previous section.

6. Conclusion

Pairs trading analysis was successfully conducted for two banking stocks, ICICI and Axis, after confirming that the datasets were co-integrated and checking for stationarity. The spread was computed, normalized, and used for signal generation over 5-day, 10-day, and 20-day periods. Subsequently, the performance metrics were presented along with equity graphs. A Decision Tree Classifier machine learning model was then employed to predict target features such as signals. The model's accuracy and the confusion matrix for all datasets were presented and analyzed.