DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

Automated Stock Trading using Machine Learning

CS 354: COMPUTATIONAL INTELLIGENCE LAB

Name	**Roll Number**
Rishika Sharma	210002063
Niranjana R. Nair	210003049

Course Instructor:

Dr. Aruna Tiwari

April 6, 2024

Contents

1 Title and Problem Definition

Over the past decade, automated stock trading has emerged as a leading-edge innovation in the financial industry. It has successfully reshaped traditional trading practices by leveraging cutting-edge technologies. Fintech innovations, coupled with advancements in machine learning, have stimulated the growth of automated trading systems. This has offered investors unprecedented opportunities to capitalize on market fluctuations with enhanced efficiency and accuracy. Leveraging sophisticated algorithms and vast datasets, these systems can swiftly analyze market trends, identify profitable opportunities, and execute trades at lightning speed, often outperforming human traders.

In this project, we have employed ensemble learning techniques such as Random Forest Classifiers and Extra Trees Classifiers to train our models. Utilizing the finta library, we calculated essential technical/synthesised features from the input data, thereby enriching our dataset for model training and testing. Additionally, we employed SHAP (SHapley Additive exPlanations) summary plots to elucidate the contribution of features to the overall behavior of our models, providing valuable insights into their decision-making process. Furthermore, we have developed a comprehensive trading strategy designed to identify profitable trades and optimize portfolio management. This strategy relies on several parameters derived from sentiment scores obtained through web scraping financial websites, allowing us to make data-driven decisions in the dynamic landscape of stock trading.

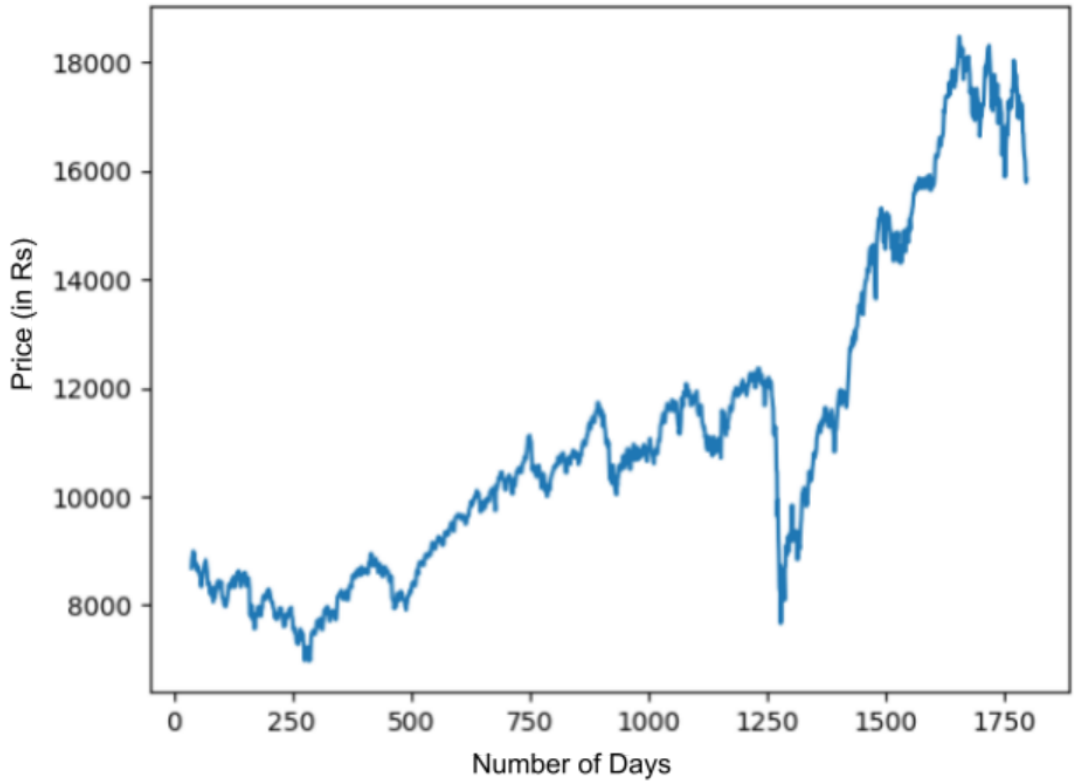
2 Analysis and Design

2.1 Data Collection/Preprocessing

The instrinsic working of any machine learning model is heavily dependent on the quality and relevance of the dataset used to train and test it.

In this project, we initiated the data collection process by acquiring 23 years (2000-2023) of NIFTY-50 data from the official website of the National Stock Exchange (NSE). The collected dataset underwent preprocessing to ensure compatibility with subsequent analysis steps. Initially, we renamed the columns to align with the expected column names by the finta library. Subsequently, we eliminated

redundant columns such as 'Index Name', as our dataset solely focused on the NIFTY-50 index. Moreover, we performed a crucial step of reversing the dataset to guarantee the ascending order of dates, which is vital for time-series analysis. Following this, we embarked on generating technical features using the finta library.



2.1.1 Input Data Features

The input data features include:

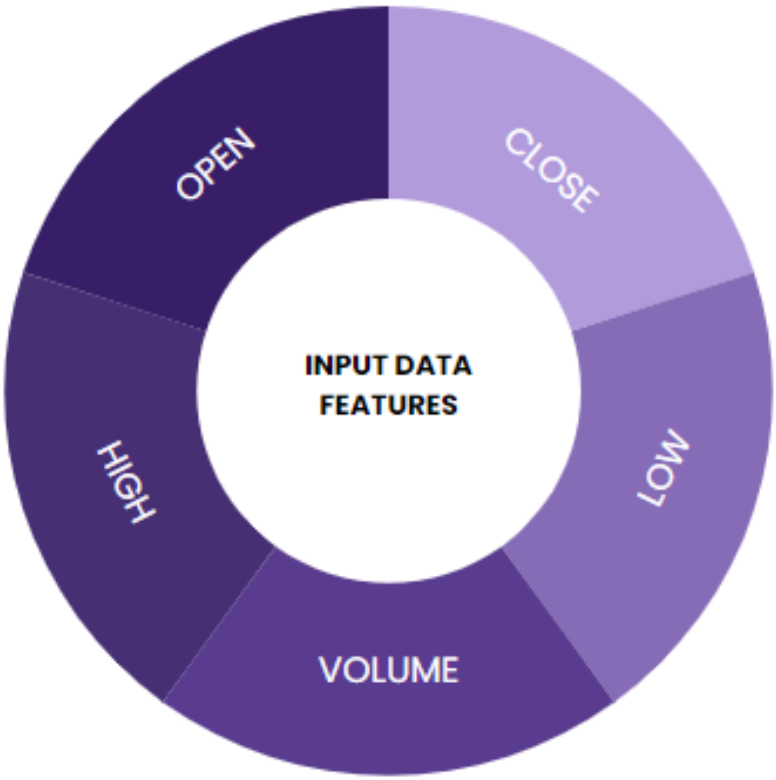
Open Price: The price of a financial instrument at the beginning of a trading period.

High Price: The highest price reached by a financial instrument during a trading period.

Low Price: The lowest price reached by a financial instrument during a trading period.

Close Price: The price of a financial instrument at the end of a trading period.

Volume (optional): The total number of shares or contracts traded during a specific period.



2.1.2 Synthesised Data Features

The synthesised data features are as follows:

Simple Moving Averages (SMA): SMA calculates the average closing price over a specified period, smoothing out price fluctuations and providing insights into the overall trend direction.

Exponential Moving Averages (EMA): EMA assigns more weight to recent closing prices, making it more responsive to short-term price changes and providing a clearer indication of trend direction.

Bollinger Bands: Bollinger Bands consist of a middle band (SMA) and upper and lower bands determined by standard deviations. They help identify volatility and potential price reversal points when the bands contract or expand.

Kaufman Adaptive Moving Averages (KAMA): KAMA adjusts its sensitivity to price changes based on market volatility, providing smoother trend signals and reducing false signals during choppy market conditions.

Parabolic Stop and Reverse (SAR): SAR identifies potential trend reversals by plotting points above or below the price, indicating when to buy or sell assets based on changes in direction.

Triangular Moving Averages (TRIMA): TRIMA assigns more weight to the middle portion of a data

Out[3]:

	close	SMA5	SMA10	SMA15	SMA20	EMA5	EMA10	EMA15	EMA20
27	1711.20	1659.96	1618.050	1613.740000	1615.1650	1667.812402	1640.849374	1629.280509	1623.017866
28	1756.00	1691.22	1633.740	1624.066667	1621.8950	1697.208268	1661.785851	1645.120445	1635.682831
29	1744.50	1712.80	1653.570	1632.326667	1627.9850	1712.972179	1676.824787	1657.542889	1646.046371
30	1702.50	1720.76	1668.870	1638.253333	1632.5300	1709.481452	1681.493008	1663.162528	1651.422907
31	1711.10	1725.06	1681.180	1646.566667	1637.7500	1710.020968	1686.876097	1669.154712	1657.106440
...	...	...	...	...	...	...	...	...	...
6016	22493.55	22421.58	22270.100	22233.693333	22132.8100	22407.111876	22311.166990	22233.443597	22165.859841
6017	22332.65	22412.43	22291.160	22247.720000	22168.6400	22382.291251	22315.072992	22245.844398	22181.744618
6018	22335.70	22398.45	22304.895	22256.970000	22198.2625	22366.760834	22318.823357	22257.076348	22196.407036
6019	21997.70	22326.73	22309.550	22253.146667	22206.1450	22243.740556	22260.437292	22224.654304	22177.482556
6020	22146.65	22261.25	22325.935	22248.426667	22217.9400	22211.377037	22239.748693	22214.903766	22174.546122

5994 rows × 47 columns

series, offering smoother trend signals while reducing lag compared to traditional SMAs.

Average Directional Index (ADX): ADX measures the strength of a trend rather than its direction, helping traders determine whether a trend is gaining or losing momentum.

Commodity Channel Index (CCI): CCI identifies cyclical trends by measuring deviations from the average price, indicating potential overbought or oversold conditions.

Moving Average Convergence Divergence (MACD): MACD identifies trend changes by comparing short-term and long-term moving averages, signaling potential buy or sell opportunities when the MACD line crosses the signal line.

Momentum indicators (MOM): MOM quantifies the rate of price change over a specified period, helping traders identify the strength and direction of trends.

Rate of Change (ROC): ROC measures the percentage change in price over a specified period, indicating the momentum of a trend and potential reversal points.

Percentage Price Oscillator (PPO): PPO compares short-term and long-term moving averages as a percentage, providing insights into the strength and direction of trends.

Relative Strength Index (RSI): RSI measures the magnitude of recent price changes to determine overbought or oversold conditions, indicating potential trend reversals.

Stochastic Oscillator: Stochastic Oscillator identifies potential trend reversals by comparing the current closing price to the price range over a specified period, signaling overbought or oversold conditions.

Ultimate Oscillator (ULTOSC): ULTOSC captures momentum across three different timeframes, providing a comprehensive view of trend strength and potential reversals.

William's %R (WILLR): WILLR measures overbought or oversold conditions by comparing the current closing price to the highest high and lowest low over a specified period.

Average True Range (ATR): ATR measures market volatility by calculating the average range between high and low prices over a specified period.

True Range (TR): TR represents the maximum of price changes within a period, providing insights into market volatility and potential trend reversals.

Typical Price (TYPPRICE): TYPPRICE calculates the average price of a financial instrument, providing a clearer picture of market sentiment and potential price movements.

Vortex indicators: Vortex indicators measure the strength and direction of trends by comparing the difference between high and low prices over a specified period.

Money Flow Volume (MFV): MFV calculates the money flow into or out of a financial instrument based on volume and typical price, indicating buying or selling pressure in the market.

2.1.3 Data Pipeline

Our data pipeline involved the following steps:

2.1.4 Data Split

We have split our dataset into training and testing sets. The training set was used to train the models and the testing set was used to test the performance of the models. We have made use of of 80-20 split where 80% of the data was used for training and 20% was used for testing.

2.1.5 Feature Generation

We synthesised technical features from input data features to provide additional information to our models.

2.1.6 Label Generation

The labels for our dataset were generated by comparing the current price of a stock with its future price. If the price increased, the data was labelled as '1'. If the price decreased, the data was labelled as '0'.

2.1.7 Smoothening

We used smoothening techniques to make the data noise-free. Smoothening helped in removing sudden spikes/drops in the stock prices and made the data more consistent. In our project, we have made

use of exponential smoothening to smooth out the input and sythesised data features.

2.1.8 Feature Scaling

We scaled the features to ensure that they had similar ranges and that no feature dominated the others. Feature scaling is necessary because it ensures that the machine learning models give equal importance to all features. We used Min-Max scaling to scale our features, which scales the values between 0 and 1.

By following this data pipeline, we ensured that our dataset was well-prepared and processed, and that our machine learning models would be trained on high-quality and relevant data, leading to accurate predictions.

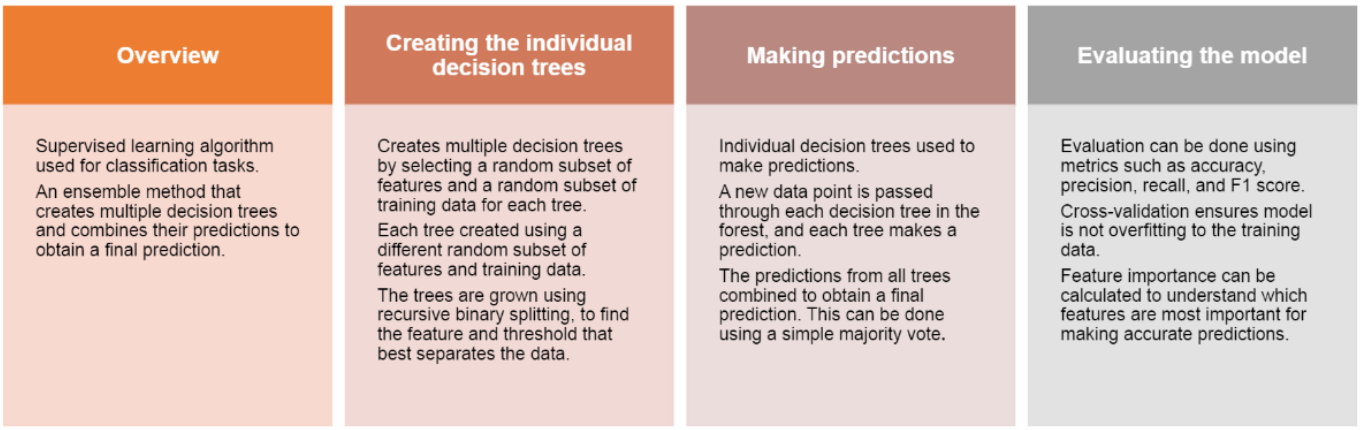
2.2 Models used

In our project, we have made use of two machine learning models to make stock predictions- Random Forest and Extra Trees Classifier

2.2.1 Random Forest Classifier

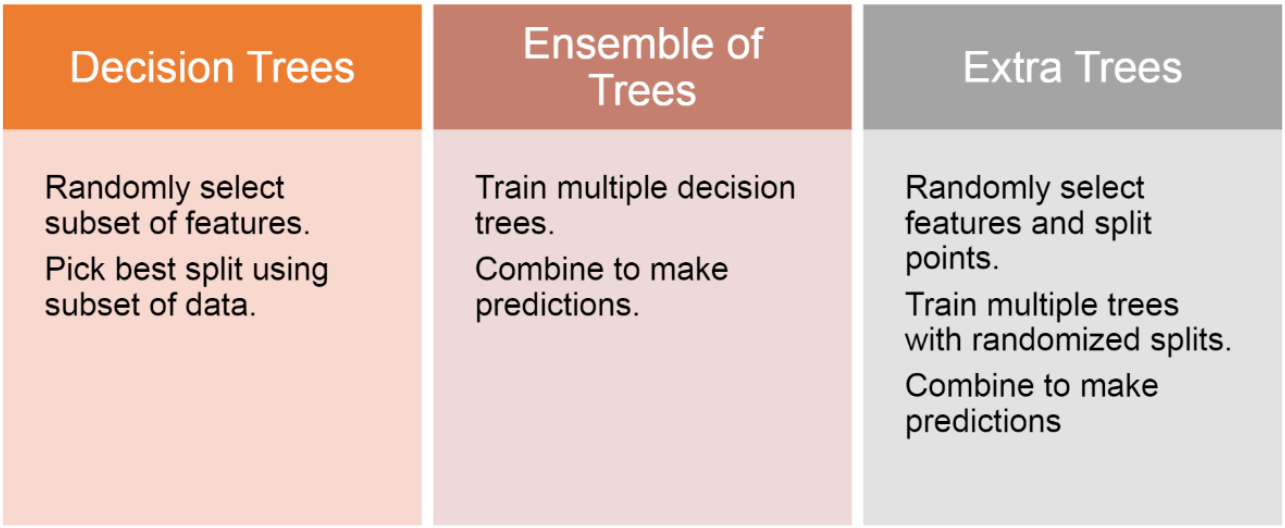
Random Forest Classifier is a supervised learning algorithm used for classification tasks. It is an ensemble method that creates multiple decision trees and combines their predictions to obtain a final prediction. The process for creating the individual decision trees involves selecting a random subset of features and a random subset of training data for each tree. Each tree is created using a different random subset of features and training data. The trees are grown using recursive binary splitting to find the feature and threshold that best separates the data.

The individual decision trees are used to make predictions. A new data point is passed through each decision tree in the forest, and each tree makes a prediction. The predictions from all trees are combined to obtain a final prediction, which can be done using a simple majority vote. Evaluation can be done using metrics such as accuracy, precision, recall, and F1 score. Cross-validation ensures the model is not overfitting to the training data. Feature importance can be calculated to understand which features are most important for making accurate predictions.



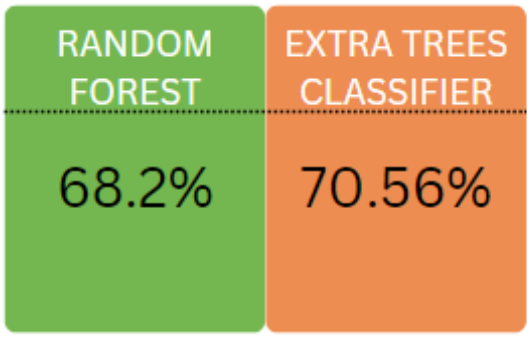
2.2.2 Extra Trees Classifier

Extra Trees Classifier is similar to Random Forest Classifier in that it also involves creating multiple decision trees to make a final prediction. However, Extra Trees Classifier takes randomization to the next level. It randomly selects features and split points and trains multiple trees with randomized splits. The trees are then combined to make predictions. The idea behind this is that the more randomization, the less likely the trees will overfit to the training data. Extra Trees Classifier also provides feature importance, which can be used to understand which features are most important for making accurate predictions.



For our project, we experimented with both models to see which one performed better on our dataset.

RESULTS:



2.3 Performance Metrics

Returns:

The Annual Return measures the total percentage change in an investment's value over a year, including capital gains and losses. It is calculated by dividing the current value of the investment by its

original value, subtracting 1, and multiplying by 100 to get the percentage return.

Sharpe Ratio:

The Sharpe Ratio is a measure of risk-adjusted return. It is calculated by subtracting the risk-free rate of return from the investment's average return and dividing the result by the standard deviation of the investment's return. The higher the Sharpe ratio, the better the risk-adjusted performance of the investment.

Maximum Drawdown:

Maximum Drawdown is the maximum percentage decline in an investment's value from a previous high. It is a measure of the investment's risk and is used to calculate the Calmar ratio.

Calmar Ratio:

The Calmar Ratio measures risk-adjusted return that compares an investment's average annual return to its maximum drawdown, the largest percentage decline in its value from a previous high. The higher the Calmar ratio, the better the risk-adjusted performance of the investment.

Win-Loss Ratio:

It is calculated by dividing the total number of winning trades by the total number of losing trades over a specific period. A win-loss ratio greater than 1 indicates that the strategy generates more winning trades than losing trades.

3 Experimentation and Results

3.1 Algorithm

To train our machine learning models, we have used an algorithm that involves splitting the dataset into training and testing sets. We have used the training set to train our models and the testing set to evaluate their performance. We have also used cross-validation to ensure that our models are not overfitting the training data.

3.1.1 Training

To ensure a fair and accurate fitting result, we have used a sliding window approach to training and testing.

The default window size taken is of 40 days, reflecting two financial months of the dataset. We have used both Random Forest and Extra Trees Classifier to predict the labels.

The default lookahead of the labels is taken as 10 days (2 weeks), which means that our models will be trained to predict whether the stock will go up or down after two weeks.

3.1.2 Trading Strategy

We have used the Extra Trees Classifier's trained model as an input to our algorithm. The dataset that is passed is unlabeled but contains all the synthesized features. The principal amount denotes

the amount of money available to purchase the shares. The trading window size denotes the days after which the predicted label is obeyed. Maximum volume is set such that any major fault in the price would not lead to a large drawdown value. Confidence is an essential indicator that determines the stock volume that needs to be traded in a position. We have updated the confidence level using the following rule:

confidence=(1-α) $\times$ confidence + $\alpha\times$result

Here, the α parameter denotes the weightage given to the current prediction, and the result signifies if the current prediction was correct or not. This approach ensures that the confidence level is updated dynamically, based on the algorithm's accuracy.

The max shares that can be traded on a day are determined by the rule:

max_shares = volume $\times$confidence^n

Here, the n parameter denotes the number of days since the last trading activity. Hence, the number of shares that can be bought or sold is determined by the balance available and the above value.

3.2 Backtesting

To validate the performance of our machine learning models, we have conducted backtesting. Backtesting involves testing a trading strategy on historical data to see how it would have performed in the past. We have used our models to generate trading signals based on their predictions and have tested these signals on historical data to see how profitable they would have been.

Back testing is an important step in evaluating the effectiveness of a trading strategy. It involves testing the strategy on historical data that the model has not been trained on to assess its performance and potential profitability.

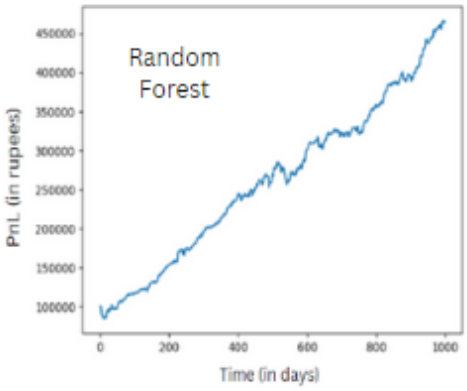
In our case, we back test our trading strategy on four years of unseen data using the both the Random Forest and Extra Trees Classifier models. The strategy is trained on 19 years of data (2000-19) to make predictions on the unseen data.

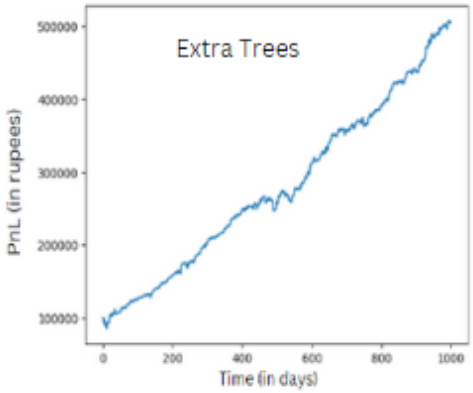
As an example test case, we consider an investment amount of INR 100000, a maximum volume of 40 shares, a window size of 5 days, a confidence rank of 0.5, alpha value equal to 0.5 and an n value of 2.4.

These parameter values are calculated based on the sentiment score obtained by performing sentiment analysis on the data scraped from Google finance website.

We also take into account a risk-free rate of 15% while calculating the Sharpe Ratio. This rate represents the average increment in the NIFTY 50 index price annually and serves as a benchmark to evaluate the performance of our trading strategy.

By testing our strategy on unseen data, we can obtain a more accurate estimate of its potential profitability and risk-adjusted performance. This helps us identify any weaknesses or areas for improvement in our model and trading strategy.





Model	Annual Returns	Cumulative Returns	Max Drawdown	Sharpe Ratio	Calmar Ratio	Win-Loss Ratio
Random Forest	50.47	406.02	-14.56	1.95	-2.88	1.71
Extra Trees	47.26	364.56	-15.41	1.81	-2.58	1.67

4 Conclusion

In conclusion, our algorithm is designed to make profitable trading decisions automatically, based on the predicted labels obtained from the Random Forest and Extra Trees Classifier. Our training approach and evaluation metrics ensure that the model is accurate and robust, and the trading strategy is based on sound principles of money management. However, it is essential to note that the financial market is unpredictable and subject to various risks, and our algorithm is not guaranteed to achieve profits in all situations. Nonetheless, we believe that our algorithm can provide valuable insights and increase the chances of achieving higher profits in this dynamic and complex domain.

References

[1] https://towardsdatascience.com/predicting-future-stock-market-trends-with-pytho

[2] https://towardsdatascience.com/deep-reinforcement-learning-for-automated-stock

[3] https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForest

[4] https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.ExtraTreesClas

[5] https://blog.quantinsti.com/mean-reversion-time-series/

[6] https://irep.ntu.ac.uk/id/eprint/32787/1/PubSub10294_702a_McGinnity.pdf