BITS F464 Machine Learning

Members:

Hardaat Singh Baath	2021A7PS2662G
Sreekar Yadlapalli	2021A7PS2602G
Tanishq Gholap	2021A7PS2728G
Mohit Tiwari	2021A7PS2719G

Link to Colab Notebook:
Stock Market Prediction using ML.ipynb

Link to GitHub repository (Has the stocks CSV file):
GitHub Repository

1. Problem Statement and Hypothesis

The goal of this project is to explore the capabilities of machine learning algorithms for time series analysis and evaluate the performance of different hypothesis classes, including linear models, decision trees, random forests, k-nearest neighbours, support vector machines, and neural networks, in the context of time series forecasting for stock price data. The task is to predict the next value in a time series based on the previous k values of open, high, low, and close prices, formulated as:

$$x_{t-k}, x_{t-k+1}, \ldots, x_{t-1} \longrightarrow x_t$$

Hypothesis: Machine learning algorithms can be used for time series forecasting, and the project aims to identify the strengths and weaknesses of each hypothesis class for this problem.

2. Methodology

2.1 Data Preparation

The dataset used in this study is the historical stock data for TATASTEEL, a major Indian steel company. The dataset contains daily records of open, high, low, and close prices, and trading volume. The relevant features (open, high, low, close) were scaled using standard normalisation techniques to ensure consistent input scales for the machine learning models. The time series forecasting task's target variable was the closing price at time t, and the input features were the open, high, low, and close prices from the previous k time steps ago (t-k). The model will use the data of the past to predict the future.

2.2 Experimental Setup

The experimental setup involved the following key steps:

- Time-based splitting of the dataset into training + testing and prediction sets to preserve the temporal order of the data and ensure realistic evaluation of the models' forecasting abilities.
- Varying the forecast_out parameter from 1 to 100 to assess the models' performance across different prediction horizons and the effectiveness of the models in predicting future values.
- Exploring different test-size values (0.1 to 0.4) to investigate the impact of varying the proportion of data used for model evaluation.
- Training and evaluating the models using the coefficient of determination (R^2 score) as the performance metric, which measures the proportion of the variance in the target variable explained by the model.
- Visualising the actual and predicted values for different forecast_out values to qualitatively analyse the models' behaviour and identify potential strengths and weaknesses.

2.3 Model Implementation

The following hypothesis classes were implemented and evaluated:

- **Linear Regression:** A simple linear model that assumes a linear relationship between the input features and the target variable.
- **Decision Tree Regression:** A tree-based model that recursively partitions the input space into regions with similar output values.
- **Random Forest Regression:** An ensemble learning method that combines multiple decision trees to improve predictive accuracy and reduce overfitting.
- **K-Nearest Neighbors Regression:** A non-parametric method that predicts the target value based on the weighted average of the k-closest training examples.
- **Support Vector Machine Regression:** A kernel-based method that constructs a high-dimensional hyperplane to maximise the margin between different classes or values.
- **Multi-layer Perceptron Regression:** A neural network model with multiple layers of interconnected nodes that can learn complex non-linear relationships between inputs and outputs.

For each model, hyperparameter tuning was performed using cross-validation on the training set to optimise the model's performance on the validation set.

3. Experimental Results and Validation

3.1 Model Accuracies

The R^2 scores achieved by the different models on the held-out test set are as follows:

- **Linear Regression:** 0.982
- **Decision Tree:** 0.961

- **Random Forest:** 0.981
- **K-Nearest Neighbours:** 0.977
- **Support Vector Machine:** 0.901
- **Multi-layer Perceptron:** 0.907

These results indicate that linear regression, random forest, and k-nearest neighbours models best explain the variance in the stock price data. At the same time, support vector machines and multi-layer perceptrons exhibited relatively lower accuracy.

3.2 Training/Prediction Time

The time taken by different models for training and prediction are as follows:

Training:
- **Linear Regression:** 67.3ms
- **Decision Tree:** 92.7ms
- **Random Forest:** 292ms
- **K-Nearest Neighbours:** 74.8ms
- **Support Vector Machine:** 179ms
- **Multi-layer Perceptron:** 918ms

Prediction:
- **Linear Regression:** 6.27ms
- **Decision Tree:** 4.89ms
- **Random Forest:** 46ms
- **K-Nearest Neighbours:** 10.3ms
- **Support Vector Machine:** 212ms
- **Multi-layer Perceptron:** 4.1ms

3.3 Result validation

To validate the results, the following measures were taken:
- The train-test split was performed using a time-based approach to ensure the integrity of the temporal order and realistic evaluation of the models' forecasting abilities.
- To assess their generalisation performance, the models were evaluated on a held-out test set, separate from the training and validation data.
- The R^2 score, a widely accepted metric for regression problems, was used to quantify the models' performance in explaining the variance in the target variable.
- The visualisations of actual and predicted values were carefully analysed to identify potential issues or biases in the predictions, such as failure to capture sharp changes or trends in the time series.

3.4 Prediction Visualisation

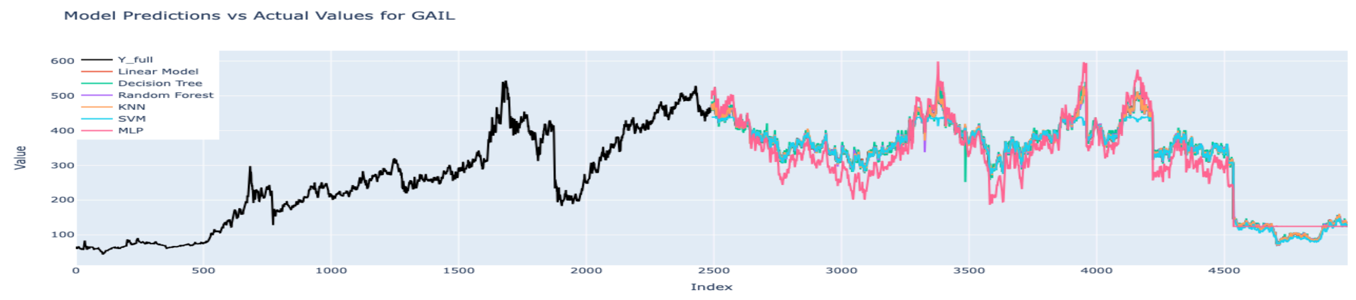
The visualisations of the actual and predicted values for different forecast_out values provide insights into the models' behaviour and strengths and weaknesses. The linear regression and random forest models capture the overall trend in the stock prices reasonably well, with their predictions closely following the actual values. However, they may struggle to accurately predict the time series's sharp changes or sudden spikes.

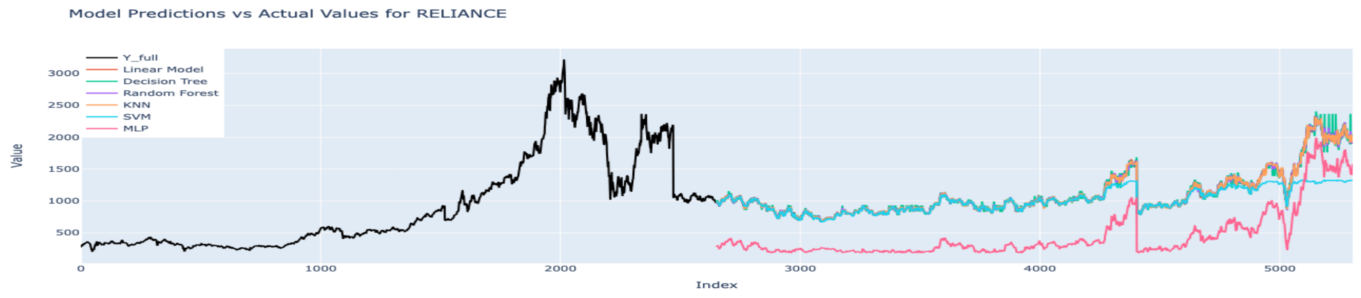
On the other hand, decision trees and k-nearest neighbours models tend to exhibit more volatile predictions, potentially overfitting to local patterns in the data. Support vector machines and neural networks, while capable of capturing non-linear relationships, have difficulty accurately forecasting stock prices, possibly due to the complexity and noise inherent in financial time series data.

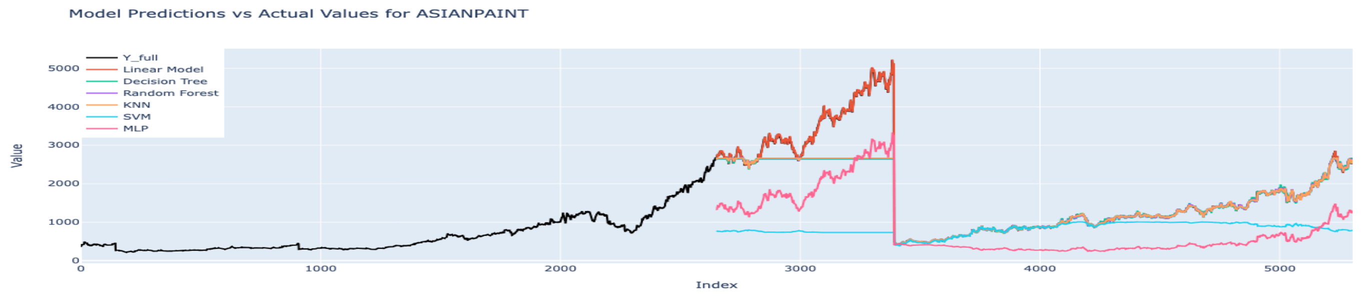
Overall, it was observed that the linear model captured the details of the trend well. At the same time, decision trees, random forests, and KNN faced issues with spikes and tended to flatten out when the stock prices increased, exposing their inability to capture spikes in the trend. SVM tend to be bad, but it did show some accurate predictions when the stock movement came close to its position. Multi-layer perceptrons performed the worst, being away from the stock value's current position, even though they showed some tendency to capture the general trend of the stock.

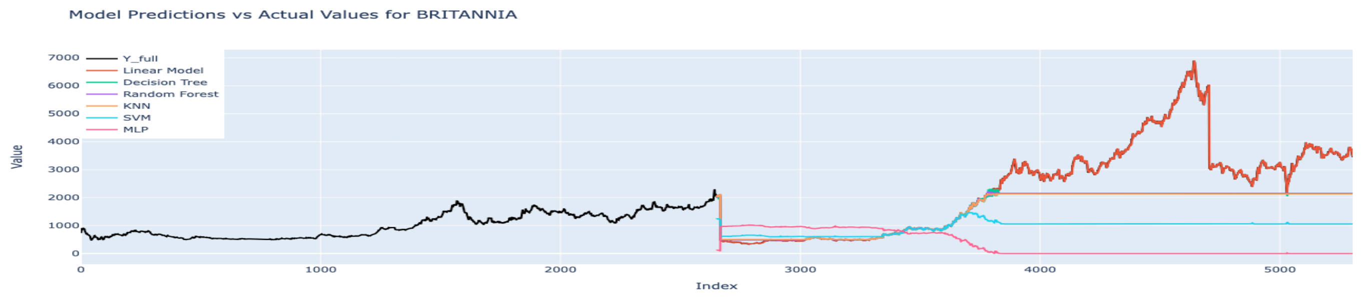
The images for the stock market for GAIL, Reliance, Asian Paints, Britannia and Kotak Bank have been shared below, along with the predictions given by various models.

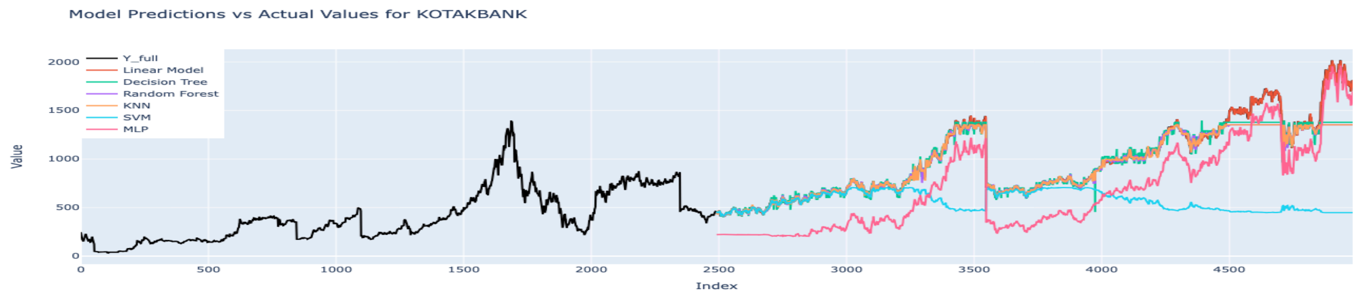
(The Colab Notebook has interactive plots.)











4. Conclusion and Future Work

Based on the experimental results, linear regression appears well-suited for time series forecasting problems, as it exhibited high accuracy and effectively captured the overall trend. Random Forest, K-nearest neighbours and Decision trees also performed reasonably well, while support vector machines and multi-layer perceptrons struggled to achieve comparable performance. It is also important to realise that time series data depend heavily on past information that ML models don't have, which plays an important role in their poor performance.

It is important to note that the performance of these models may vary depending on the characteristics of the time series data, such as the presence of non-linear patterns, seasonality, or other complex structures. Additionally, the choice of hyperparameters and the feature engineering process can significantly impact the model's performance.

Future work could explore more advanced time series forecasting techniques, such as autoregressive integrated moving average (ARIMA) models, long short-term memory (LSTM) networks, or transformer-based architectures. Additionally, incorporating domain-specific knowledge and feature engineering tailored to the time series data could improve the performance of the evaluated models. Exploring ensemble methods, such as combining predictions from multiple hypothesis classes, to improve overall predictive performance.

Furthermore, evaluating the models on a diverse range of time series datasets from different domains would provide a more comprehensive understanding of the strengths and weaknesses of each hypothesis class in the context of time series forecasting.