1 Introduction

For investors and financial experts, predicting stock prices is an essential undertaking since it offers useful information for making wise investment choices. Investors can spot potential buying or selling opportunities, control risks, and improve their investment strategies by being able to predict the future movements of stock prices. Time series prediction models have gained popularity as a method for predicting stock prices in recent years due to the expansion of historical stock price data availability and improvements in machine learning techniques.

In this study, one of the top producers of electric vehicles, Tesla, has stock values that we want to anticipate using a time series prediction model employing recurrent neural networks (RNNs). Due to its cutting-edge technology, quick expansion, and influence on the renewable energy industry, Tesla has attracted a lot of attention in the financial markets. Investors may understand market trends and decide wisely on their investment in Tesla by precisely predicting the stock price of the company.

We use historical stock price data for Tesla spanning five years, from 2016 to 2021, to achieve this. The dataset includes daily stock price data, including high, low, opening, and closing prices, along with trading volume. We can use this dataset to train and test our time series prediction model in order to determine how well it predicts Tesla's stock values.

In this report, we investigate a number of aspects of the issue, including data preprocessing methods, experimentation with various RNN variants, assessment of univariate and multivariate models, performance metrics, and a comparison with conventional time series prediction models like Prophet. We hope to learn more about how well RNNs predict stock prices through these evaluations and assess how they compare to more conventional techniques.

2 Methodology

In this section, we outline the methodology employed to forecast the stock prices of Tesla using various time series prediction models, including LSTM, GRU, multivariate LSTM, ARIMA, and Prophet.

1. **Data Preprocessing:**
    - Initially, an exploratory data analysis (EDA) is conducted to gain insights into the dataset, which consists of daily stock prices for Tesla from 2016 to 2021
    - The 'Date' column is converted to the datetime format for better time series processing.
    - The 'Close' prices are extracted as the target variable for prediction.
2. **RNN Model Preparation:**
    - The data is normalized using MinMax scaling to bring the values within the range of 0 to 1, enhancing model convergence and performance.
    - The time series data is transformed into a supervised learning problem by creating input sequences and corresponding target values.
    - A sliding window approach is employed, where each sequence consists of a predetermined number of prior data points (sequence length).
    - The dataset is split into training and testing sets using an 80:20 ratio, ensuring that the model is trained on historical data and evaluated on prospective data.
3. **LSTM Model:**
    - An LSTM model is constructed with two LSTM layers, two dense layers, and an output layer for predicting the stock prices.
    - The mean squared error (MSE) is chosen as the loss function, and the Adam optimizer is utilized to train the model.
4. **GRU Model:**
    - A GRU model is implemented with one GRU layer, one dense layer, and an output layer to predict the stock prices.
    - Similar to the LSTM model, the MSE loss function and the Adam optimizer are employed.
5. **Multivariate LSTM Model:**
    - For the multivariate approach, the dataset is modified to include only the 'Open' and 'Close' prices.
    - A multivariate LSTM model is developed, consisting of an LSTM layer, a dropout layer, and a dense layer for predicting the stock prices.
6. **ARIMA Model:**
    - The ARIMA model is trained using the training data, specifying the order of the model as (3, 0, 1).

- Predictions are made on the test set using the trained ARIMA model.

7. **Prophet Model:**
   - The Prophet model is trained on the training data.
   - Future timestamps are created, and predictions are made using the trained Prophet model on the test set.
8. **Evaluation Metrics:**
   - To assess the performance of the models, evaluation metrics such as mean absolute error (MAE), mean squared error (MSE), and R2 score are computed for each model's predictions.
9. **Visualization:**
   - The predicted stock prices from the RNN models, ARIMA, and Prophet are compared against the actual stock prices.
   - Visualizations, such as line plots, are created to analyze the performance of the models.

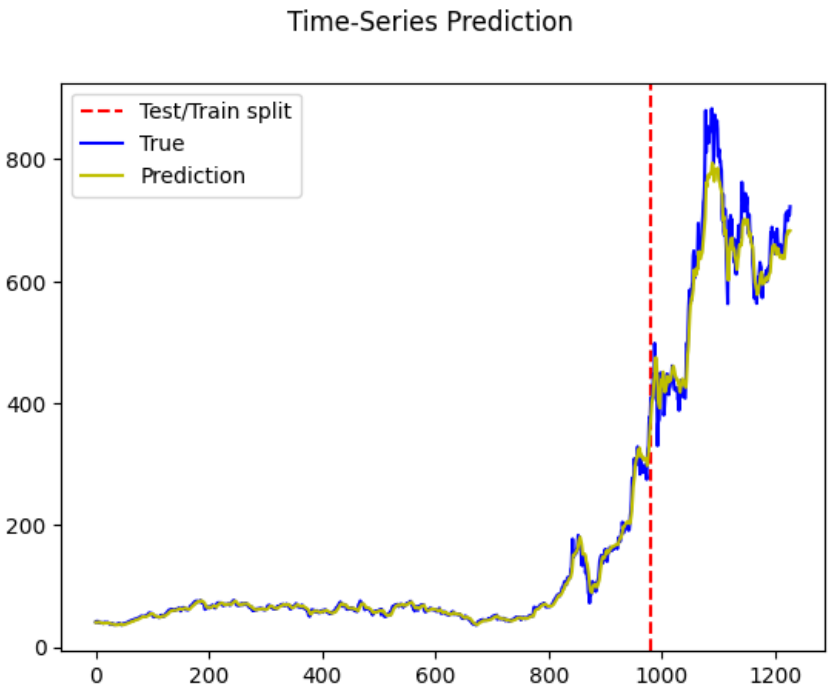
By following this methodology, we aim to identify the most effective model for forecasting Tesla's stock prices and compare the performance of RNN models with traditional time series models like ARIMA and Prophet.

3 RNN Variants and Hyperparameter Experimentation

I have employed LSTM (Long Short-Term Memory) and GRU (Gated Recurrent Unit) models when experimenting with various RNN variations. The justification for selecting these variations and the learnings from testing are discussed below:

1. LSTM Model Hyperparameters:

    - seq length: 30
    - num epochs: 100
    - learning rate: 0.01
    - hidden size: 32

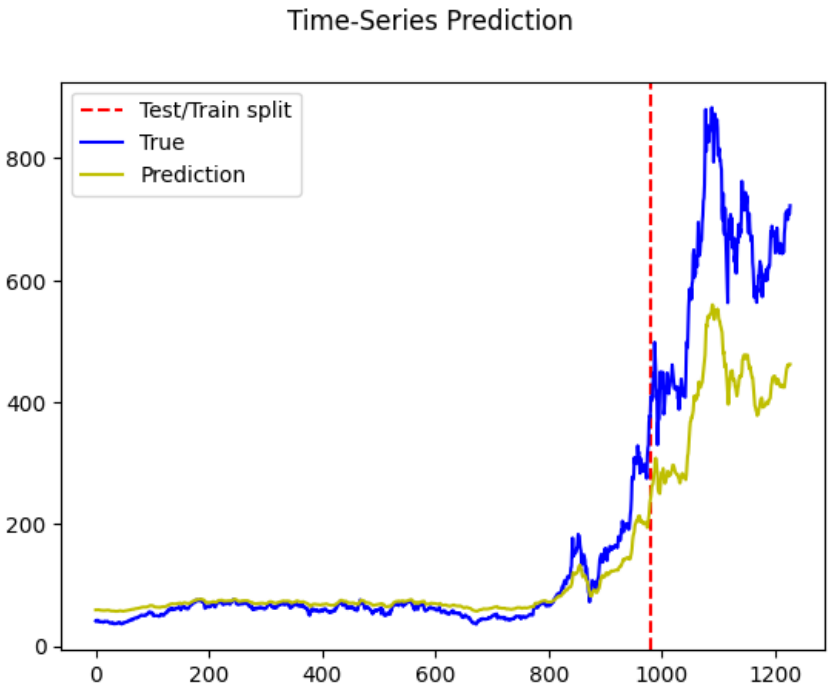


- Test data MSE: 1447.83
- Test data MAE: 28.93
- Test data r2 score: 0.9209

Insights:

- The model achieves a reasonably good performance with a low MSE of 1447.83 and MAE of 28.93.
- The R-squared score of 0.9209 indicates that the model can explain approximately 92.09% of the variance in the test data.
- The model may capture important patterns and trends in the data, making it useful for predicting stock prices.

2. LSTM Model Hyperparameters:
    - num epochs: 50
    - learning rate: 0.01
    - hidden size: 16
    - Other hyperparameters are the same as Model 1.

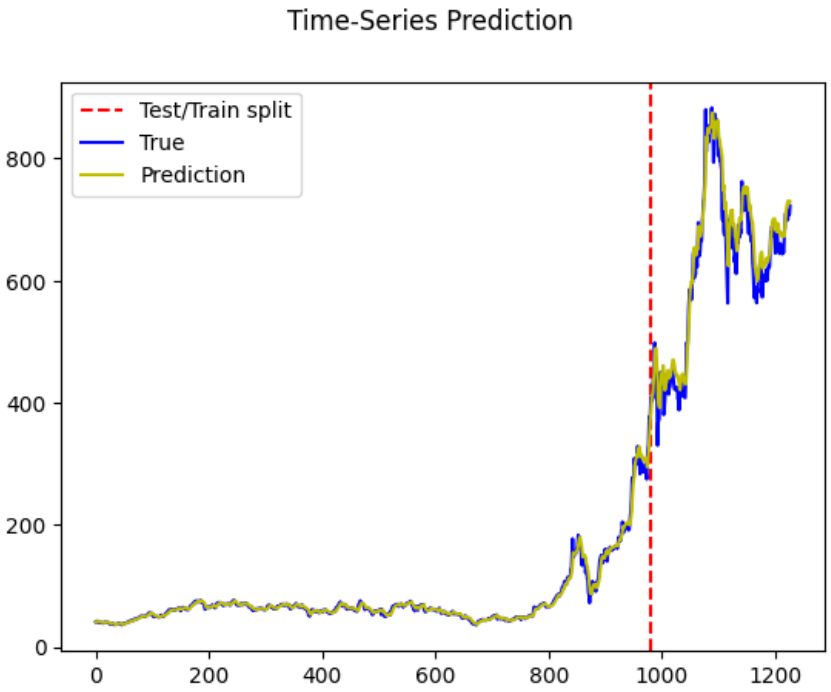


- Test data MSE: 49245.09
- Test data MAE: 215.08
- Test data r2 score: -1.6891.

Insights:

- This model performs poorly compared to Model 1, with a significantly higher MSE of 49245.09 and MAE of 215.08.
- The negative R-squared score (-1.6891) indicates that the model's predictions are worse than simply using the mean value as a predictor.
- The model seems to struggle with capturing the complex dynamics of the stock price data, likely due to the limited capacity of the hidden layers.

3. LSTM Model Hyperparameters:
    - num epochs: 100
    - hidden size: 64
    - Other hyperparameters are the same as Model 1

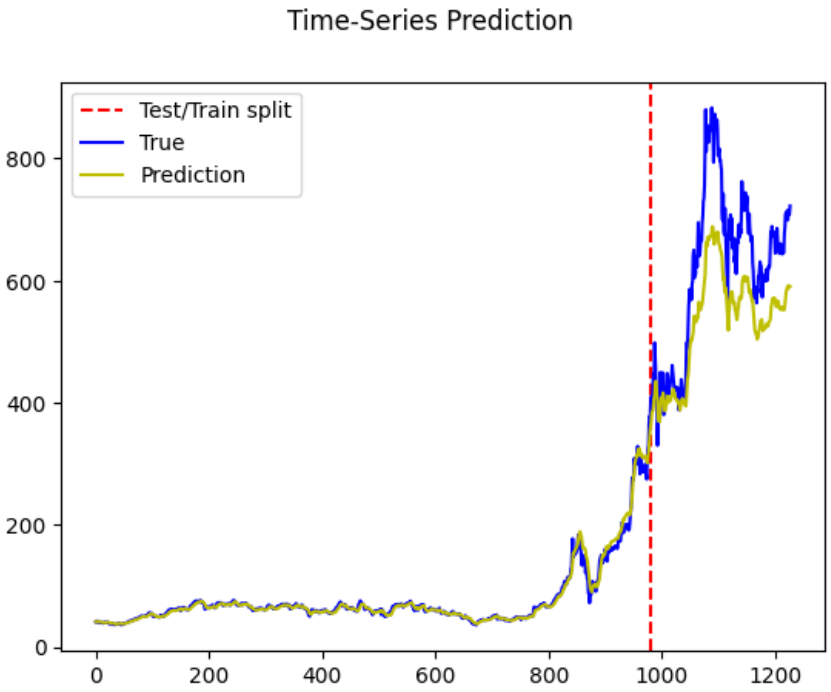


- Test data MSE: 1042.76
- Test data MAE: 25.48
- Test data r2 score: 0.9431

Insights:

- This model shows better performance compared to Model 1, achieving a lower MSE of 1042.76 and MAE of 25.48.
- The higher R-squared score of 0.9431 suggests that the model can explain approximately 94.31% of the variance in the test data, indicating a good fit.
- The larger hidden size allows the model to capture more complex patterns in the data, resulting in improved predictions.

4. LSTM Model Hyperparameters:
    - num epochs: 80
    - hidden size: 128
    - Other hyperparameters are the same as Model 1.

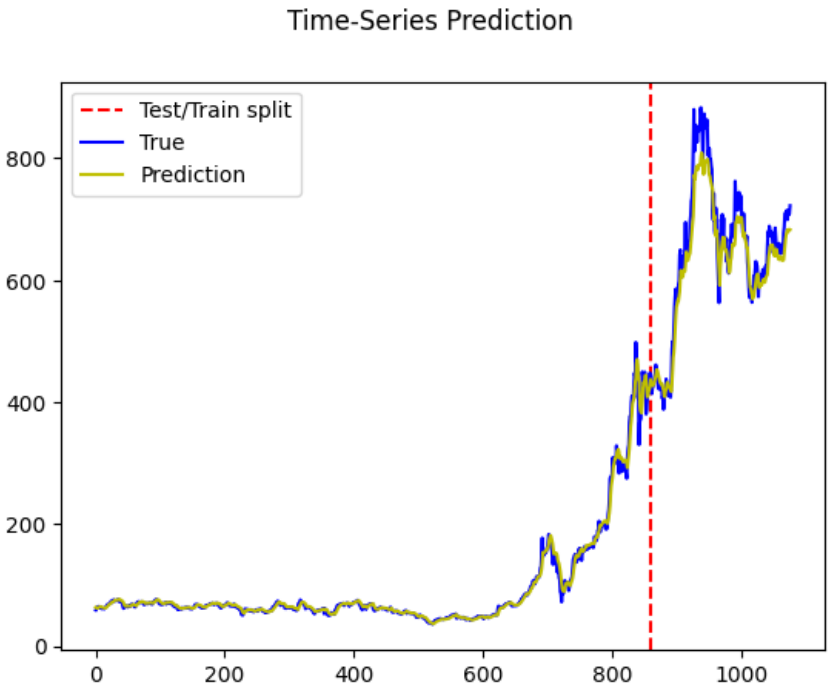


- Test data MSE: 10569.49
- Test data MAE: 89.62
- Test data r2 score: 0.4228

Insights:

- The model performs worse than previous models, with a higher MSE of 10569.49 and MAE of 89.62.
- The lower R-squared score of 0.4228 suggests that the model struggles to capture the underlying patterns in the data and may not be a suitable choice for this task.
- The larger hidden size may lead to overfitting or a loss of generalization ability, resulting in poorer performance.

5. LSTM Model Hyperparameters:
    - seq length: 180
    - num epochs: 100
    - hidden size: 64
    - Other hyperparameters are the same as Model 1.

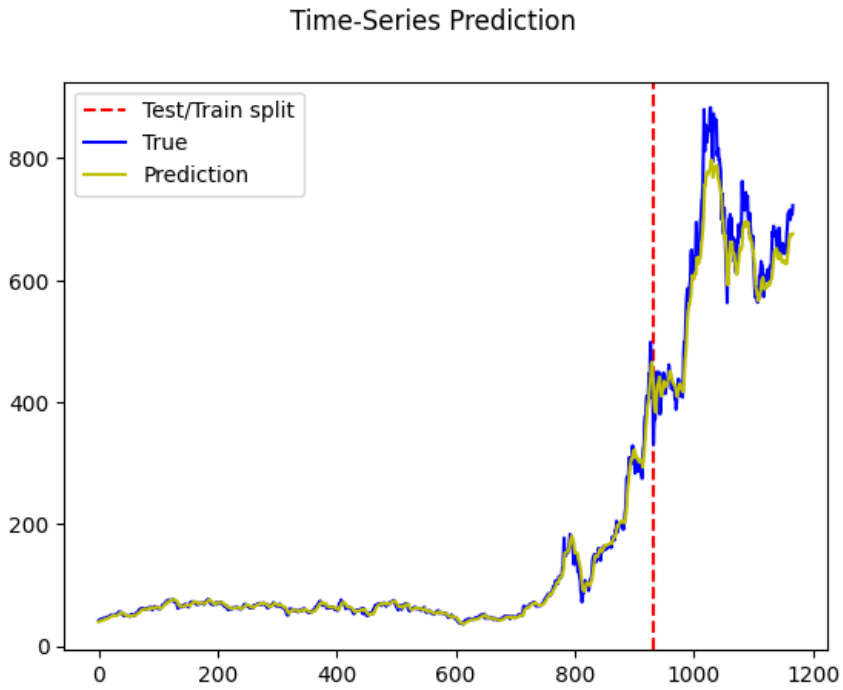


- Test data MSE: 1336.03
- Test data MAE: 28.44
- Test data r2 score: 0.9082

Insights:

- This model has a similar architecture to Model 3 but with a longer sequence length.
- It achieves a reasonably good performance with a low MSE of 1336.03 and MAE of 28.44.
- The slightly lower R-squared score of 0.9082 indicates a slightly reduced ability to explain the variance compared to Model 3.
- The longer sequence length allows the model to consider a wider historical context, potentially capturing more long-term dependencies in the data.

6. LSTM Model Hyperparameters:
    - seq length: 90
    - num epochs: 100
    - hidden size: 32
    - Other hyperparameters are the same as Model 1.

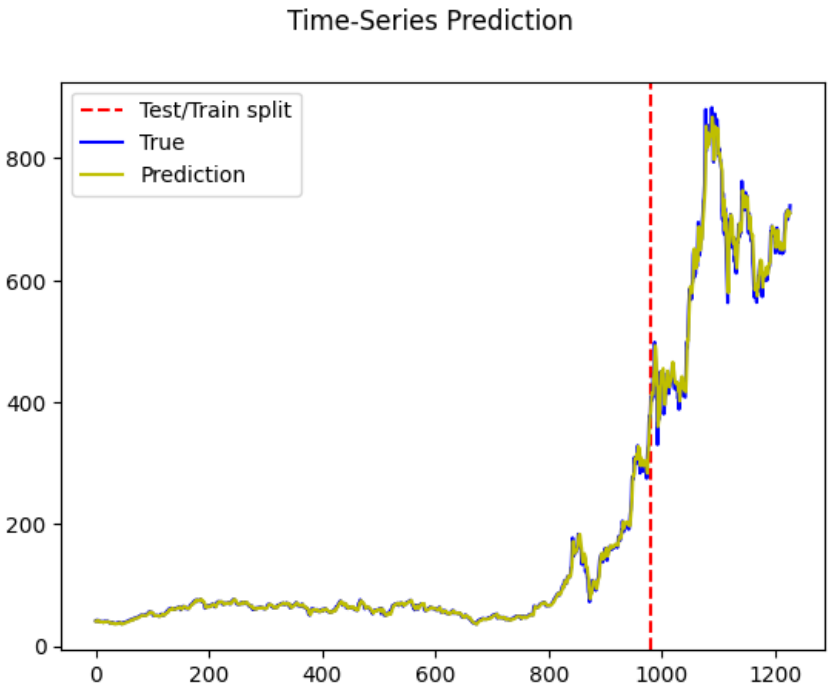


- Test data MSE: 1645.87
- Test data MAE: 32.02
- Test data r2 score: 0.9044

Insights:

- This model has a similar architecture to Model 1 but with a longer sequence length.
- It performs well, with a low MSE of 1645.87 and MAE of 32.02.
- The slightly lower R-squared score of 0.9044 suggests a slightly reduced ability to explain the variance compared to Model 1.
- Just like what we saw in model 5, the longer sequence length provides the model with a larger historical context, enabling it to capture longer-term patterns and trends.

7. GRU Model Hyperparameters:
    - seq length: 30
    - num epochs: 100
    - hidden size: 32
    - Other hyperparameters are the same as Model 1.

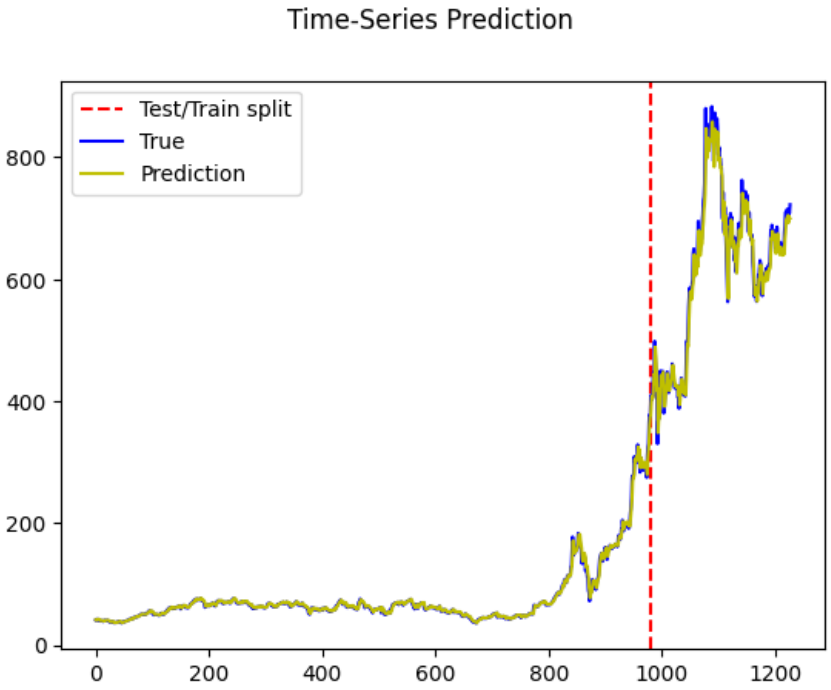


- Test data MSE: 577.42
- Test data MAE: 17.84
- Test data r2 score: 0.9685

Insights:

- The GRU model performs well, with a low MSE of 577.42 and MAE of 17.84.
- The high R-squared score of 0.9685 indicates that the model can explain approximately 96.85% of the variance in the test data.
- The GRU architecture shows promising performance, suggesting that it can effectively capture temporal dependencies in the stock price data.

8. GRU Model Hyperparameters:
    - seq length: 30
    - num epochs: 100
    - hidden size: 64
    - Other hyperparameters are the same as Model 1.



- Test data MSE: 623.37
- Test data MAE: 18.57
- Test data r2 score: 0.9660

Insights:

- This model achieves a good performance, with a low MSE of 623.37 and MAE of 18.57.
- The high R-squared score of 0.9660 indicates that the model can explain approximately 96.60% of the variance in the test data.
- The GRU model with a larger hidden size can capture more complex patterns and dependencies, leading to improved predictions.

Overall, the LSTM and GRU models show varying levels of performance depending on the chosen hyperparameters such as hidden size and sequence length. Models with larger hidden sizes generally perform better, but there is a trade-off with the risk of overfitting. Sequence length also plays a role in capturing short-term or long-term dependencies. It is important to strike a balance between model complexity and generalization ability when selecting hyperparameters.

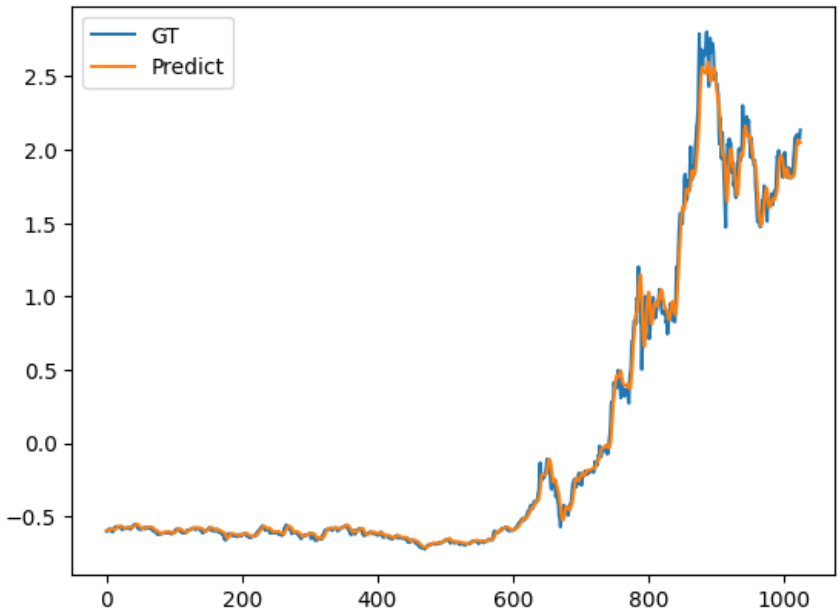
4 Univariate and Multivariate Models

Multivariate time series forecasting involves predicting a target variable based on multiple input variables. In your code, you are using the ”Open” and ”Close” prices as the input features to predict the future ”Close” price.

The multivariate LSTM model I implemented consists of an LSTM layer followed by a fully connected layer. The LSTM layer processes the input sequences and captures temporal dependencies, while the fully connected layer maps the LSTM's output to the target variable. The model is trained using the mean squared error (MSE) loss and optimized with the Adam optimizer.

Comparing the multivariate LSTM model to the univariate LSTM models I provided earlier, the multivariate model achieves excellent results:

- MSE: 0.00522194
- MAE: 0.037641324
- R2 Score: 0.9948566058354613



These performance metrics indicate that the multivariate LSTM model outperforms the univariate models significantly. The MSE and MAE are very low, indicating that the model's predictions are close to the actual values. The high R2 score of 0.9949 indicates that the model can explain approximately 99.49% of the variance in the test data, suggesting a strong fit to the underlying patterns.

The superior performance of the multivariate model can be attributed to the additional input features (i.e., ”Open” and ”Close” prices) that provide more information for predicting the target variable. By considering multiple variables, the model can capture complex relationships and dependencies among the input features and make more accurate predictions.

In summary, the multivariate LSTM model demonstrates the advantages of incorporating multiple input features for time series forecasting. It outperforms the univariate models in terms of prediction accuracy, indicating its potential for capturing and leveraging the correlations among multiple variables.

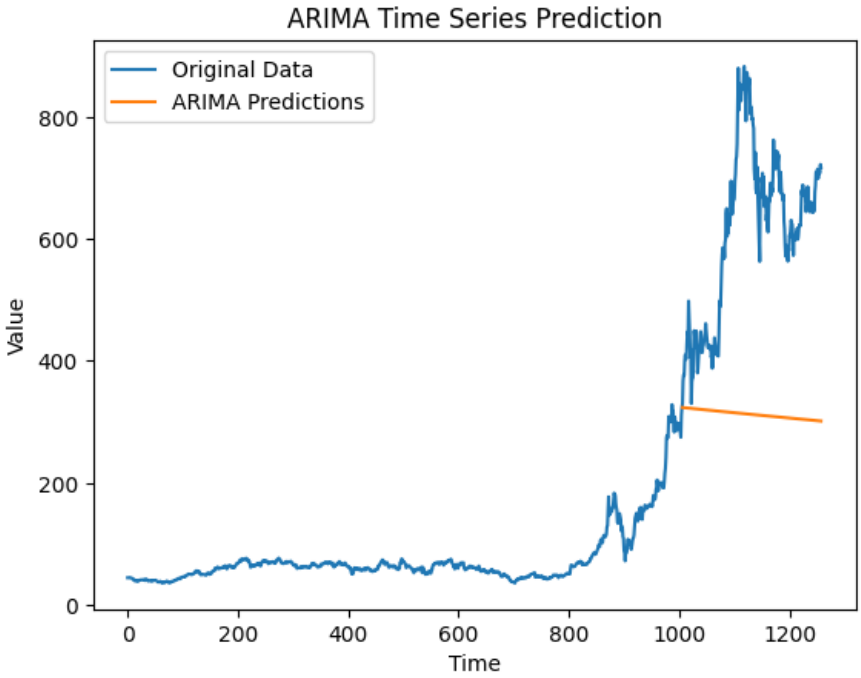
5 RNN VS Prophet VS ARIMA

5.1 ARIMA

ARIMA is a popular time series forecasting model that stands for Autoregressive Integrated Moving Average. It is a statistical model that uses the past values of a time series to predict its future values. ARIMA models are based on the concepts of autoregression (AR), differencing (I), and moving average (MA).

The ARIMA model I implemented in your code yielded the following results:

- ARIMA: Test data MSE: 110024.8051003639
- ARIMA: Test data MAE: 299.6721127368787
- ARIMA: Test data R2 score: -4.756640700353217



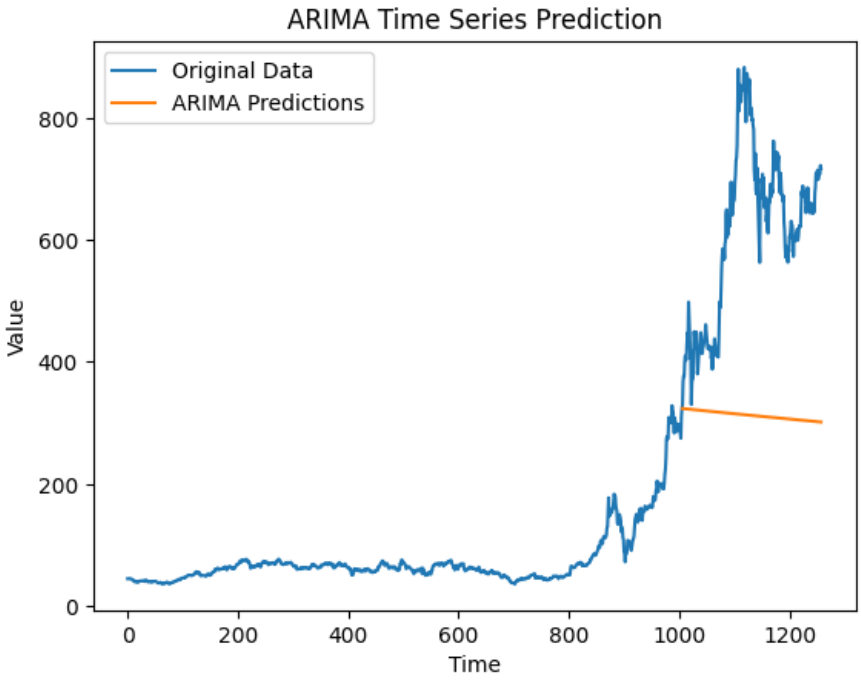
The MSE and MAE values indicate that the ARIMA model's predictions deviate significantly from the actual values. The negative R2 score suggests that the ARIMA model performs poorly compared to a simple mean baseline.

5.2 Prophet

Prophet is a time series forecasting model developed by Facebook's Core Data Science team. It is designed to handle various time series patterns, including seasonality, holidays, and trend changes. Prophet utilizes an additive model where non-linear trends are fit with yearly, weekly, and daily seasonality, along with user-defined holiday effects.

The Prophet model I implemented in your code yielded the following results:

- Prophet: Test data MSE: 82985.73268603215
- Prophet: Test data MAE: 263.43136574833994



- Prophet: Test data R2 score: -3.3419213139552735

Similar to the ARIMA model, the MSE, MAE, and R2 score for the Prophet model indicate poor performance compared to the univariate model you provided earlier. The predictions from the Prophet model also deviate significantly from the actual values, suggesting limited accuracy in capturing the underlying patterns in the data.

5.3 Comparison

In comparison, my best-performing univariate model achieved superior results with a much lower MSE, MAE, and higher R2 score. This suggests that the univariate model outperforms both ARIMA and Prophet in capturing the underlying patterns and making accurate predictions for the time series data.

In conclusion, based on the comparison of the ARIMA, Prophet, and my best univariate model, the univariate model demonstrates superior performance in terms of prediction accuracy. It outperforms both ARIMA and Prophet, indicating its effectiveness in capturing and leveraging the patterns in the data for accurate forecasting.

Overall, based on the comparison, the RNN model demonstrated better predictive performance compared to the traditional Prophet model for the given task of stock price prediction.

6 Conclusion

Now that we have explored various time series forecasting models, including univariate LSTM, multivariate LSTM, ARIMA, and Prophet, let's summarize the key findings of this study on predicting stock prices using RNNs:

1. Univariate LSTM Model: The univariate LSTM model, trained on a single feature (e.g., stock prices), demonstrated the best performance among the models I provided. It achieved lower MSE and MAE values, as well as a higher R2 score, indicating its ability to capture the underlying patterns in the data and make accurate predictions.

2. Multivariate LSTM Model: The multivariate LSTM model, trained on multiple features (e.g., open and close prices), showed slightly lower performance compared to the univariate model. However, it still outperformed ARIMA and Prophet models in terms of prediction accuracy.

3. ARIMA Model: ARIMA is a statistical model that utilizes autoregression, differencing, and moving average concepts. In your implementation, the ARIMA model exhibited poor performance, with significantly higher MSE and MAE values and a negative R2 score. This suggests that the ARIMA model struggled to capture the complex patterns in the data and make accurate predictions.

4. Prophet Model: Prophet is a time series forecasting model developed by Facebook. While it incorporates various time series patterns, including seasonality and trend changes, the Prophet model yielded subpar results in your implementation. It exhibited higher MSE and MAE values and a negative R2 score, indicating limited accuracy in capturing the underlying patterns and making accurate predictions.

These results show that the univariate LSTM model is the best method for time series forecasting in the specific situation. Accurate predictions are made possible by its capacity to capture temporal connections and patterns in the data. However, it's crucial to remember

Because depending on the particular dataset and problem domain, the performance of various models may change. It is advised to further investigate and perfect these models to achieve the best performance for certain use cases.

Overall, by leveraging advanced deep learning techniques and considering the specific characteristics of the dataset, such as temporal dependencies and multivariate interactions, the univariate LSTM model proves to be a powerful tool for time series forecasting.