Stock Price Prediction

Presented By:

1900520130019 Devang Gupta

1900520130046 Shashank Tripathi

1900520130064 Vivek Kumar Pandey

Supervised By:

Dr. Pawan Kumar Tiwari

Ms Sakshi Srivastava

Introduction

- The stock market is a marketplace where investors can buy and sell shares of publicly traded companies, allowing them to participate in the growth and success of these businesses.

- Stocks are often viewed as a barometer of the overall wealth of the economy, and fluctuations in the stock market can have significant impacts on businesses, governments, and individuals. Investors use the stock market as a tool for generating wealth, building their retirement portfolios, and making informed decisions about their financial future.

- The stock market is influenced by a wide range of factors, including economic indicators, company performance, global events, and political developments. It is a highly competitive environment where investors must constantly analyze and evaluate information in order to make informed decisions.

- Despite its complexities, the stock market plays a critical role in driving innovation and economic growth. By providing a platform for companies to raise capital and investors to participate in their success, the stock market fuels entrepreneurship and encourages businesses to take risks and innovate.

Problem Statement

- The Nifty 50 is a collection of India's top-performing companies and includes a diverse range of industries, including banking, energy, and technology. The stock prices of these companies are influenced by a variety of factors, including economic indicators, government policies, global events, and company-specific news.

- Traditional methods for predicting stock prices, such as technical analysis and fundamental analysis, have limitations in capturing the complex and dynamic nature of the stock market. Additionally, these methods may not effectively capture the unique characteristics of individual stocks and may not provide accurate predictions for all companies in the Nifty 50 index.

- The goal of this project is to develop a machine learning model that can accurately predict the stock prices of companies in the Nifty 50 index. The ultimate aim of this project is to provide investors with a valuable tool for making informed trading decisions in the Nifty 50 market.

Motivation

- Traditional methods for predicting stock prices, such as technical analysis and fundamental analysis, may not be sufficient for capturing the complex and dynamic nature of the stock market. Additionally, these methods may not effectively capture the unique characteristics of individual stocks and may not provide accurate predictions for all companies

- Machine learning techniques, particularly deep learning models such as Long Short-Term Memory (LSTM), have shown promise in predicting stock prices. These models are able to capture the non-linear relationships and temporal dependencies in the data, making them well-suited to the dynamic nature of the stock market.

- The development of an accurate machine learning model for predicting the stock prices of companies in the Nifty 50 index could have significant implications for investors and traders. By providing reliable predictions of stock prices, this model could help investors make informed decisions about buying and selling stocks, maximizing their returns and minimizing their risks. Ultimately, this project has the potential to make a valuable contribution to the field of stock market prediction and has practical applications for investors.

Literature Review

- In this paper researchers proved that financial markets, even though they are highly chaotic, are to some extent predictable. This paper and many other researchers have questioned the validity of EMH. This has led to the rise of different stock prediction techniques with methodologies from economy, statistics, data mining, AI etc [6].

- This published research reviewed and highlighted different share price prediction techniques and highlighted recent advancements in stock analysis and categorised different techniques under four categories—statistical, pattern recognition, machine learning (ML), and sentiment analysis. These categories mostly fall under the broader category of technical analysis, however, there are some machine learning techniques which also combine the broader categories of technical analysis with fundamental analysis approaches to predict the stock markets. The paper shows a taxonomy of popular stock prediction techniques [1].

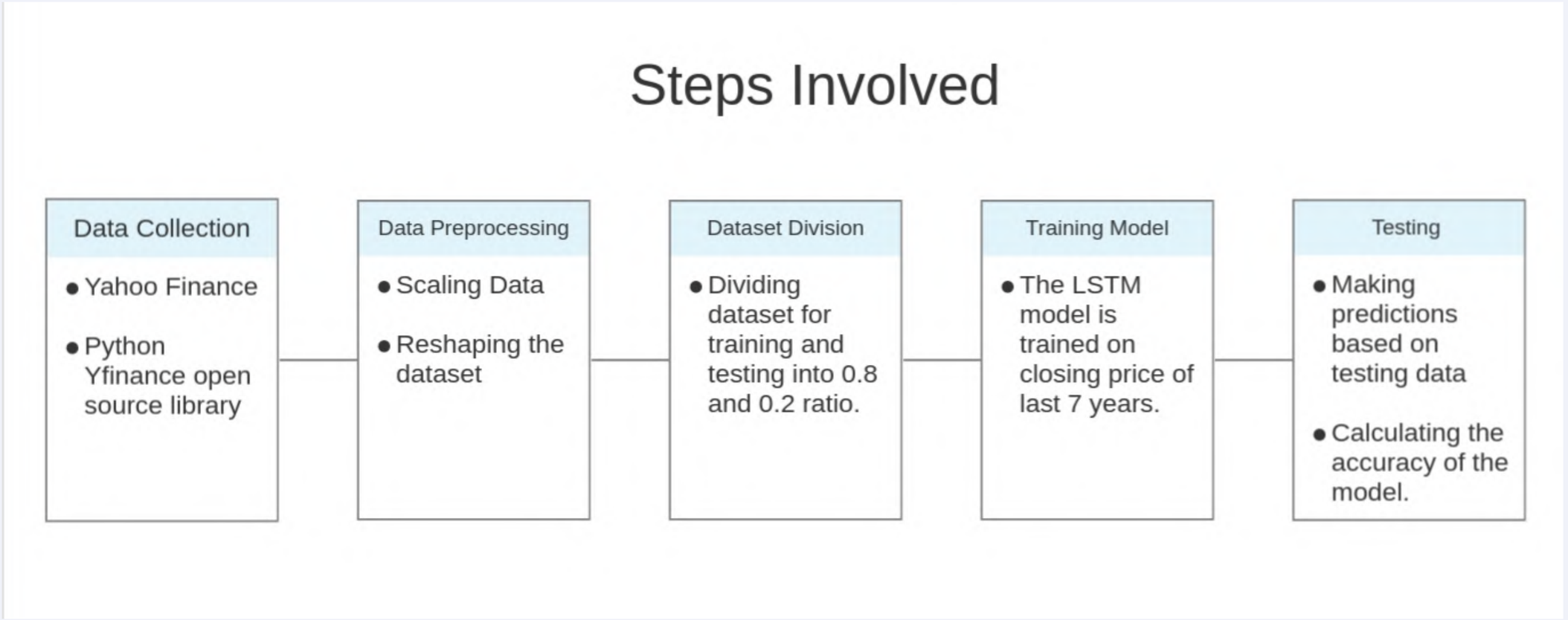
- This paper compared 3 different supervised learning techniques. ARIMA, ANN and LSTM. The paper also gave introduction to LSTMs and explained the need of models like LSTM as opposed to simple ANNs or RNNs. The paper concluded that the model with most accuracy in these three techniques is the LSTM one [2].

Literature Review

- In this paper the writers implemented a LSTM network to predict Nifty prices with features like OHLC. Their results showed that the LSTM achieves an RMSE of 0.00859 for the test data in terms of daily percentage changes [4].

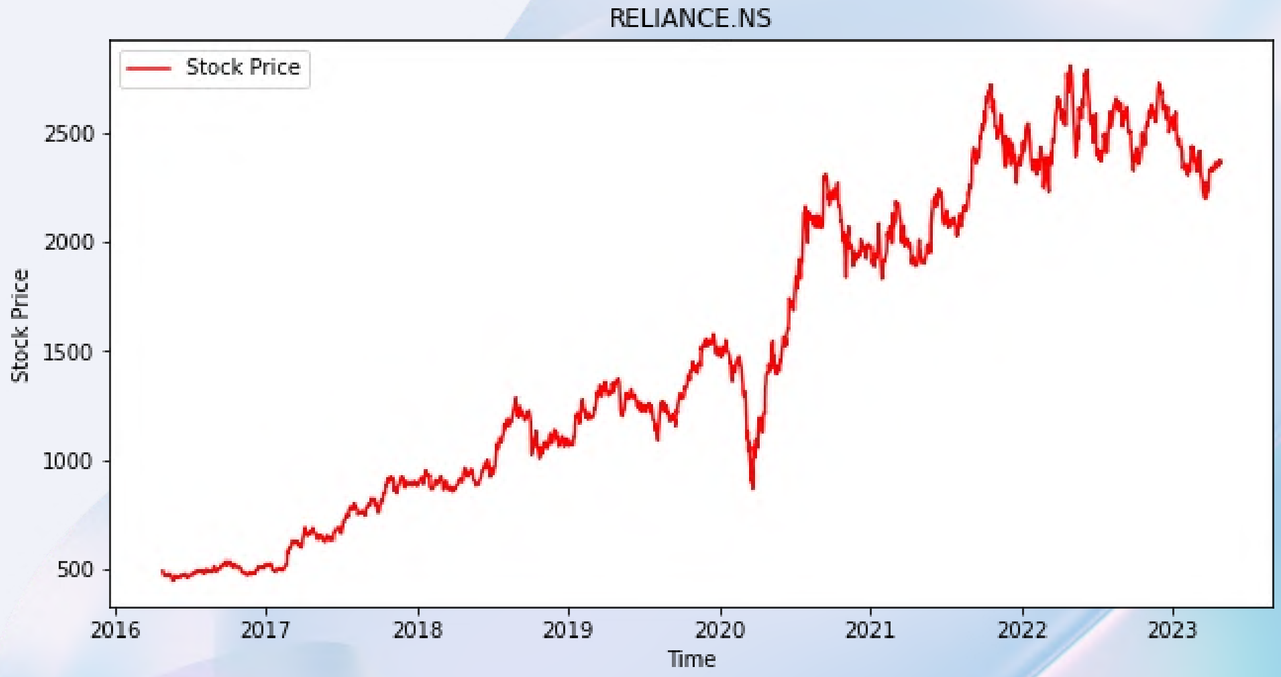
- In this research paper the writers applied three different Recurrent Neural Network models namely a basic RNN, the LSTM, and the Gated Recurrent Unit (GRU) on Google stock price to evaluate which variant of RNN performs better. It was evident from the results that the LSTM outperformed other variants with a 72% accuracy on a five-day horizon and the paper also explained and displayed the hidden dynamics of RNN [3].

Methodology



Data Collection

- The dataset has been downloaded from with the help of Yfinance, a open source python library.
- The Yfinance library scrapes stock price data from Yahooo Finance.
- The dataset contains stock prices of the last 7 years.



Date	Open	High	Low	Close	Volume	Dividends	Stock Splits
2016-04-27 00:00:00+05:30	489.788254	493.661722	488.874030	490.510010	6692966	0.0	0.0
2016-04-28 00:00:00+05:30	487.911653	490.413770	476.820566	479.587341	28589468	0.0	0.0
2016-04-29 00:00:00+05:30	478.023520	480.477520	470.998369	472.850891	10835868	0.0	0.0
2016-05-02 00:00:00+05:30	471.070523	477.518269	470.589352	474.895874	4901531	0.0	0.0
2016-05-03 00:00:00+05:30	475.401104	478.745284	468.063207	469.530792	8109395	0.0	0.0
...	...	...	...	...	...	...	...
2023-04-21 00:00:00+05:30	2350.649902	2361.000000	2336.399902	2349.000000	3529236	0.0	0.0
2023-04-24 00:00:00+05:30	2375.000000	2380.899902	2348.000000	2358.000000	5970048	0.0	0.0
2023-04-25 00:00:00+05:30	2366.000000	2380.600098	2350.500000	2376.050049	4262471	0.0	0.0
2023-04-26 00:00:00+05:30	2379.000000	2386.100098	2354.050049	2362.100098	3977129	0.0	0.0
2023-04-27 00:00:00+05:30	2375.000000	2378.699951	2364.000000	2371.149902	2674955	0.0	0.0

1732 rows × 7 columns

Preprocessing and Dataset Division

- Transformed the dataset to contain only the "Closing" price as a list of integers for all the companies
- Transformed the data using the MinMaxScaler to scale the data to a range of 0 to 1. This ensures that all features have the same scale and prevents the model from being biased towards features with larger values.
- The dataset is then divided into training and testing sets so as to evaluate. Here, the training values are taken as the more recent values.
- 80% of the intital dataset is being used for training and the rest will be used for testing.
- The dataset is then reshaped according to the input of model

```
trainData= {}
for k in symbols:
    trainData[k] = scaled_values[k][:training_data_len[k],:]
```

```
len(trainData[rel]) #len of Reliance's training data
```

```
1386
```

```
x_train,y_train=defaultdict(list),defaultdict(list) #x_train is the independent feature and y_train is the depend
```

```
days = 50
for k in companyPrice :
    for i in range(days,len(trainData[k])) :
        x_train[k].append(trainData[k][i-days:i,0])
        y_train[k].append(trainData[k][i,0])
```

```
for k in symbols :
    x_train[k],y_train[k]=np.array(x_train[k]),np.array(y_train[k])
```

Reshape train dataset because lstm model expects 3D data

```
for k in symbols :
    x_train[k] = np.reshape(x_train[k],(x_train[k].shape[0],x_train[k].shape[1],1))
```

```
x_train[rel].shape
```

```
(1336, 50, 1)
```

Model and Algorithms Used

Simple Moving Average

- SMA, short for Simple Moving Average, calculates the average of a range of stock (closing) prices over a specific number of periods in that range.
- The formula for SMA is:

$$SMA=(P1+P2+ \dots +Pn)/N$$

where Pn = the stock price at time point n, N = the number of time points.

Exponential Moving Average

- Different from SMA, which assigns equal weights to all historical data points, Exponential Moving Average, applies higher weights to recent prices.
- The magnitude of the weighting factor depends on the number of time periods.
- The formula to calculate EMA is:

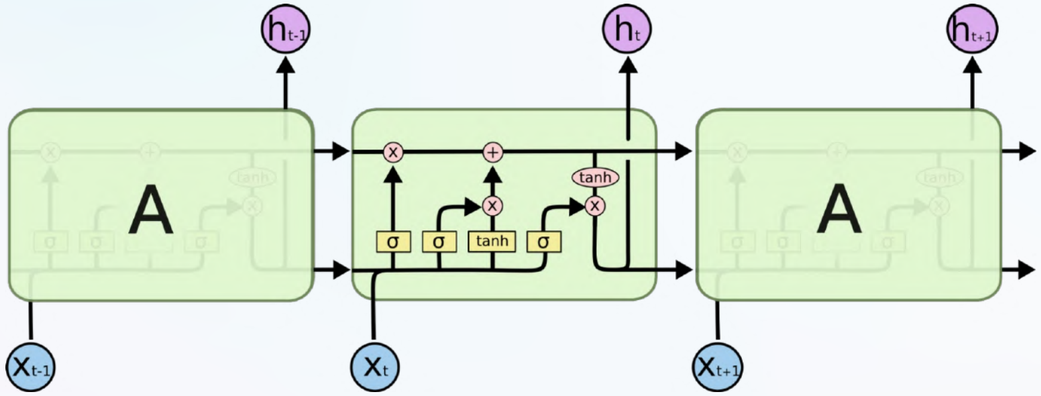
$$EMA_t = P_t * k + EMA_{t-1} * (1-k)$$

where P_t = the price at time point t, EMA_{t-1} = EMA at time point t−1,
N = number of time points in EMA, and weighting factor $k = 2/(N+1)$.

- One advantage of the EMA over SMA is that EMA is more responsive to price changes, which makes it useful for short-term trading.

LSTM

Long short-term memory (LSTM) is an artificial neural network used in the fields of artificial intelligence and deep learning. Unlike standard feedforward neural networks, LSTM has feedback connections. LSTMs are a type of recurrent neural networks (RNN). To understand LSTM we must first understand RNNs.



Structure of a LSTM cell [7]

A recurrent neural network can be thought of as multiple copies of the same network, each passing a message to a successor. This type of backpropagation allows the network to retain information which helps them to identify patterns. RNN have a problem of long term dependency i.e. retaining context and patterns for longer sequences of data. This is where LSTMs come into picture.

Need of LSTM

LSTMs are explicitly designed to avoid the long-term dependency problem. Remembering information for long periods of time is practically their default behaviour, not something they struggle to learn.

LSTMs work in a three-step process -

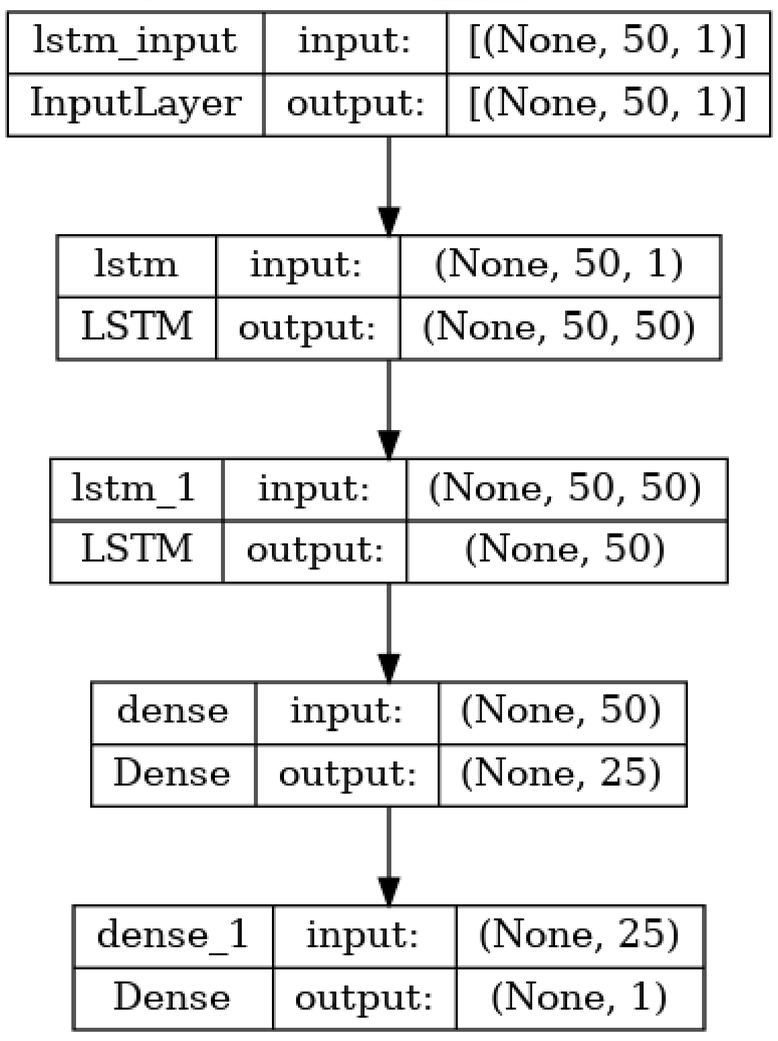
• The first step in LSTM is to decide which information to be omitted from the cell in that particular time step. It is decided with the help of a sigmoid function. It looks at the previous state (h t -1) and the current input xt and computes the function.

• There are two functions in the second layer. The first is the sigmoid function, and the second is the tanh function. The sigmoid function decides which values to let through (0 or 1). The tanh function gives the weightage to the values passed, deciding their level of importance from -1 to 1.

• The third step is to decide what will be the final output. First, you need to run a sigmoid layer which determines what parts of the cell state make it to the output. Then, you must put the cell state through the tanh function to push the values between -1 and 1 and multiply it by the output of the sigmoid gate.

In our project we will train LSTMs on the closing price of stock data.

Model Architecture




```
model = Sequential()
model.add(LSTM(50,activation='relu',return_sequences=True,input_shape=(days,1)))
model.add(LSTM(50,return_sequences=False))
model.add(Dense(25))
model.add(Dense(1))
model.compile(optimizer='adam', loss='mean_squared_error')
plot_model(model, show_shapes=True, show_layer_names=True)
```

Tools Used

Python

Tensorflow

FastAPI

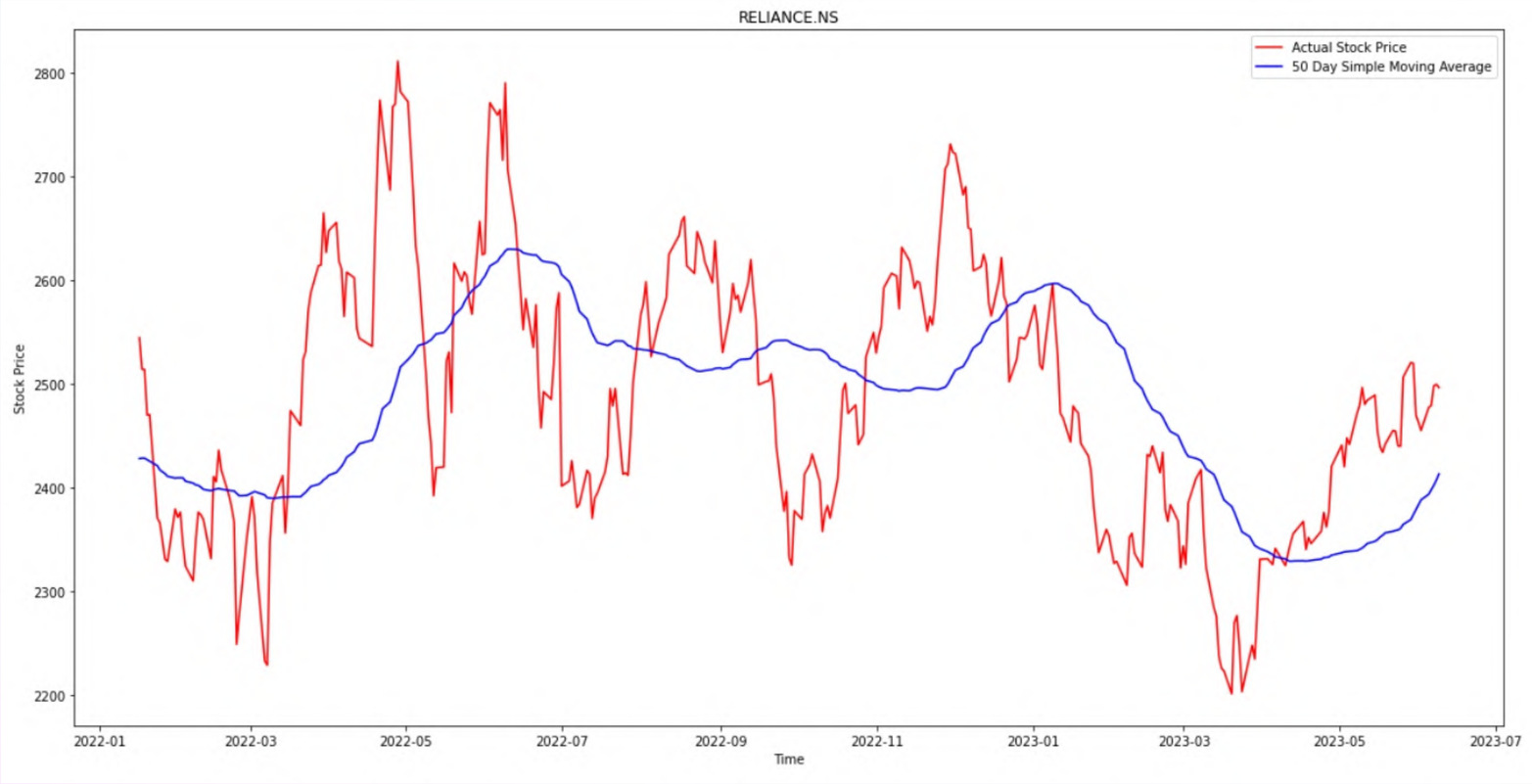
Evaluation Metric - Root Mean Square Error

Root Mean Square Error (RMSE) is a commonly used metric for evaluating the accuracy of a predictive model, particularly in regression tasks. It measures the average magnitude of the residuals or prediction errors between the predicted values and the actual values. RMSE is calculated by taking the square root of the mean of the squared differences between the predicted and actual values.

$$RMSE = \sqrt{\sum_{i=1}^{n} \frac{(\hat{y}_i - y_i)^2}{n}}$$

Result

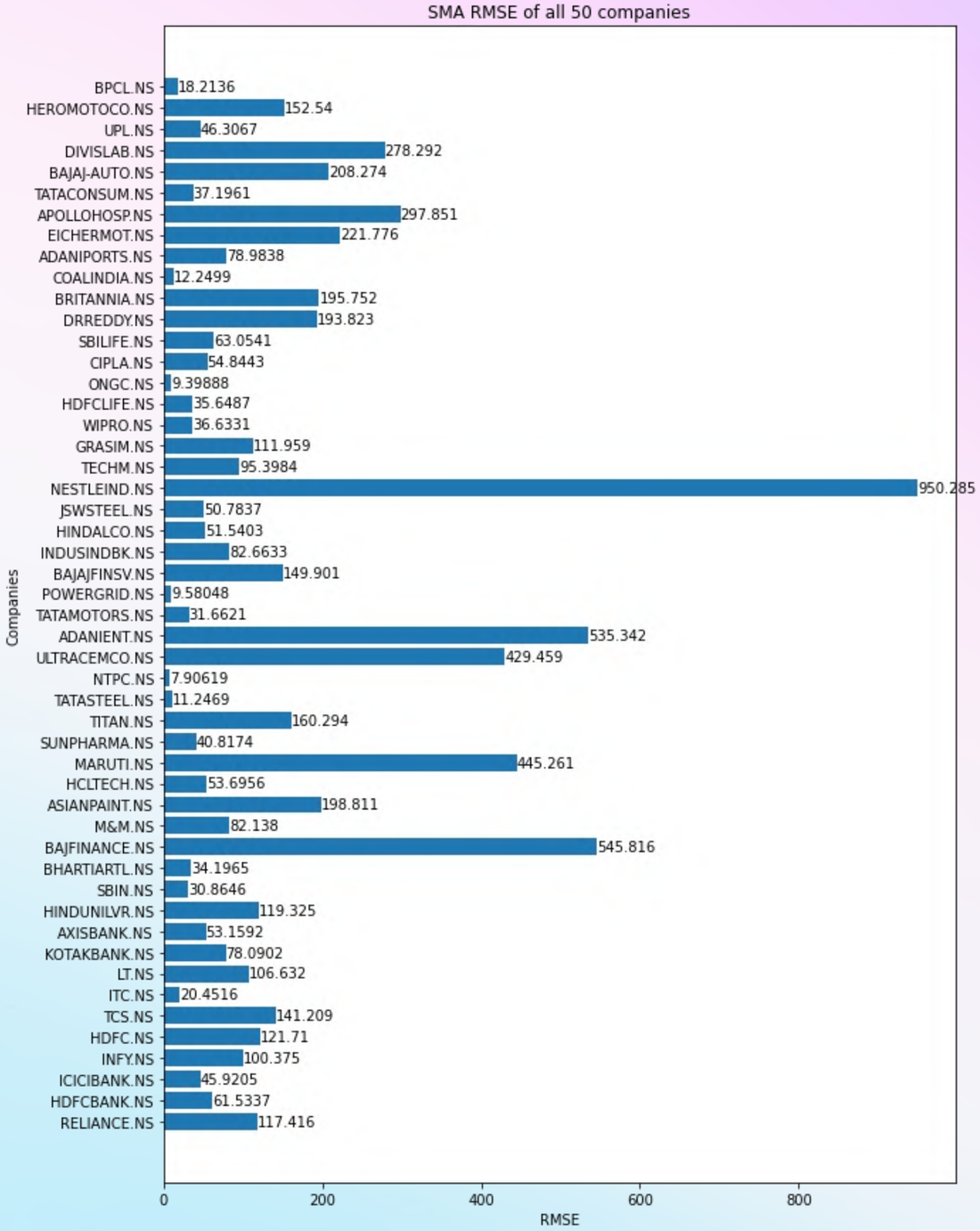
Simple Moving Average



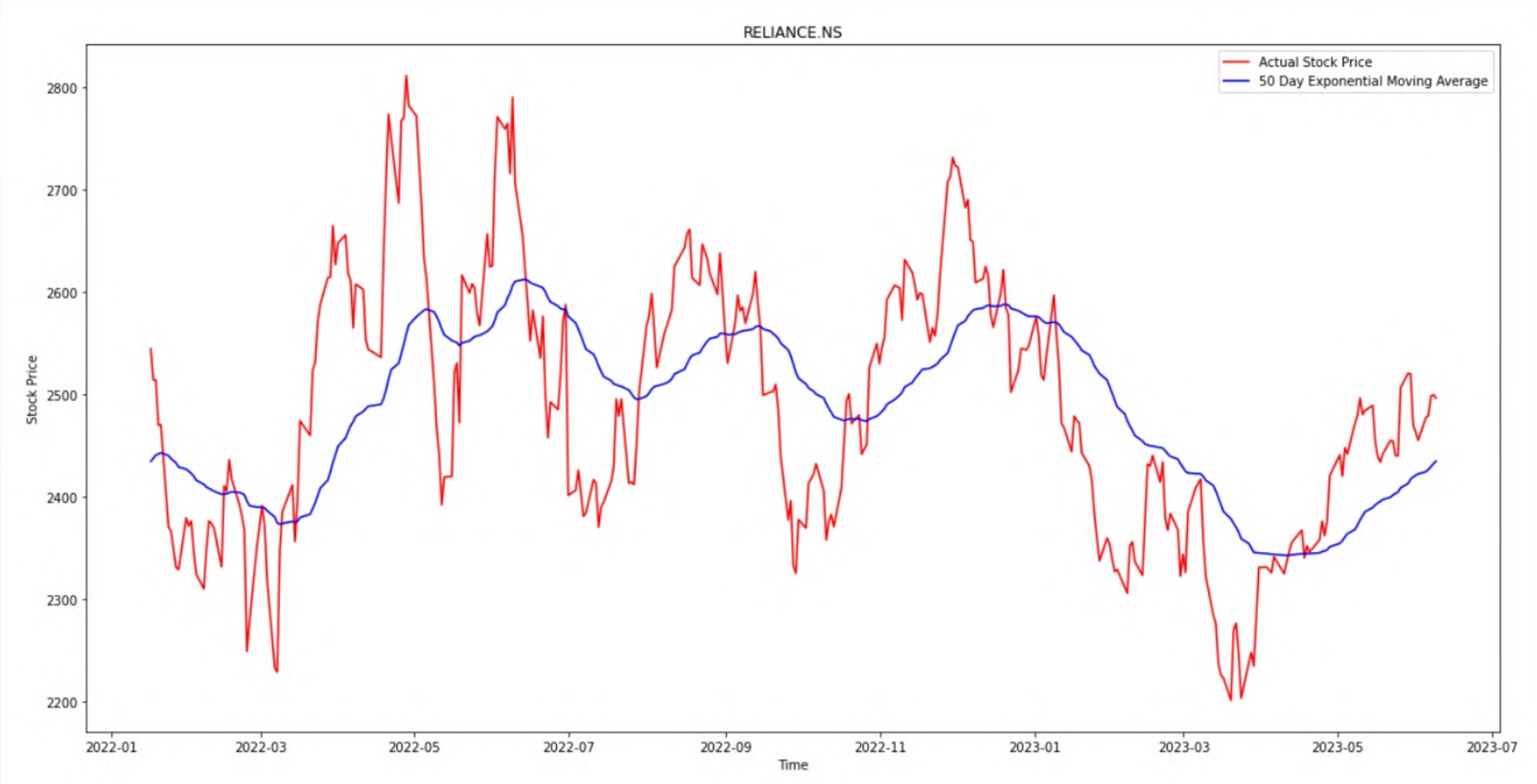
The root mean square error value of SMA for Reliance is 117.416429

RMSE of SMA

The mean and median of root mean square error values for SMA for all companies are 140.3256190501452 and 80.56092503251921 respectively.



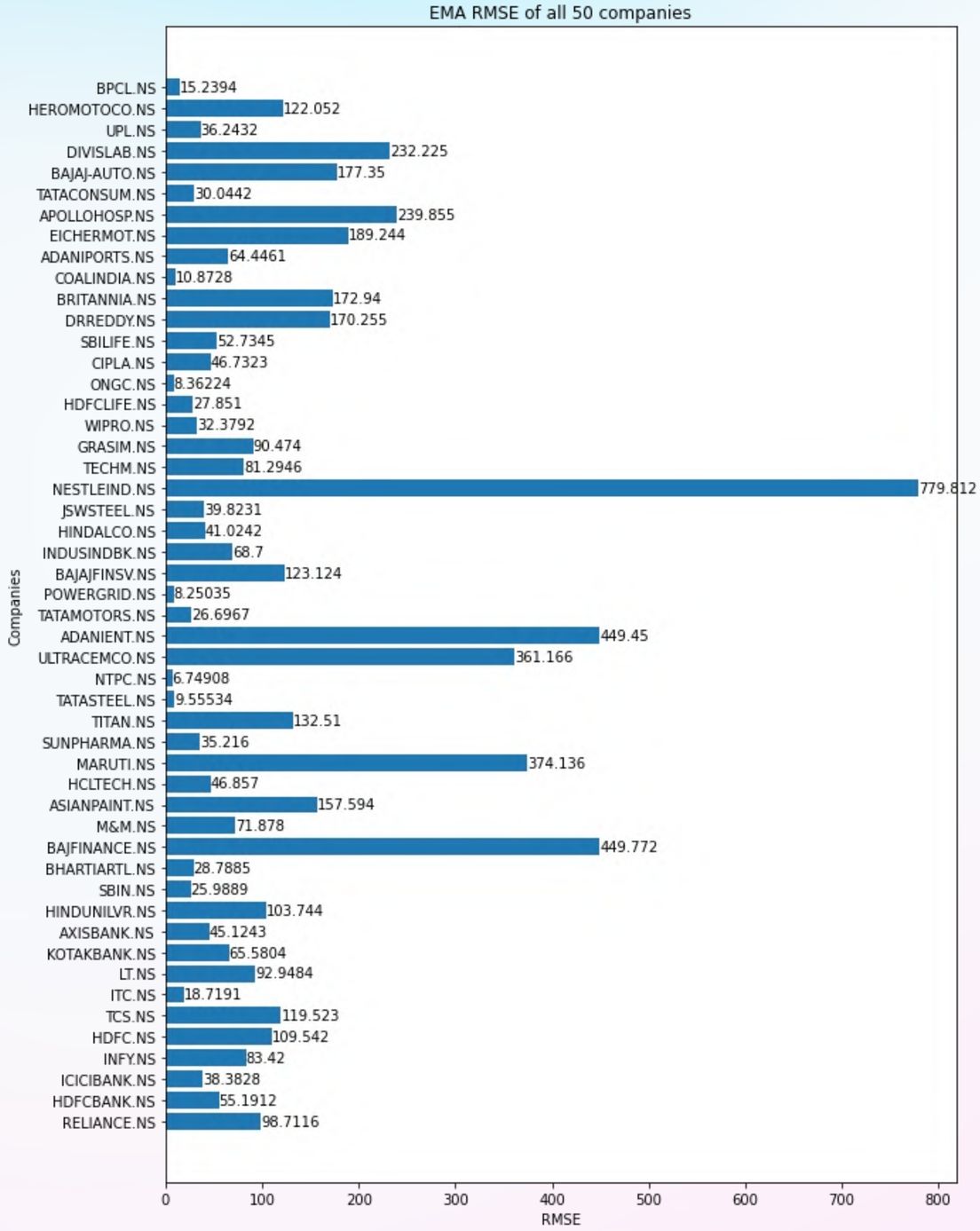
Exponential Moving Average



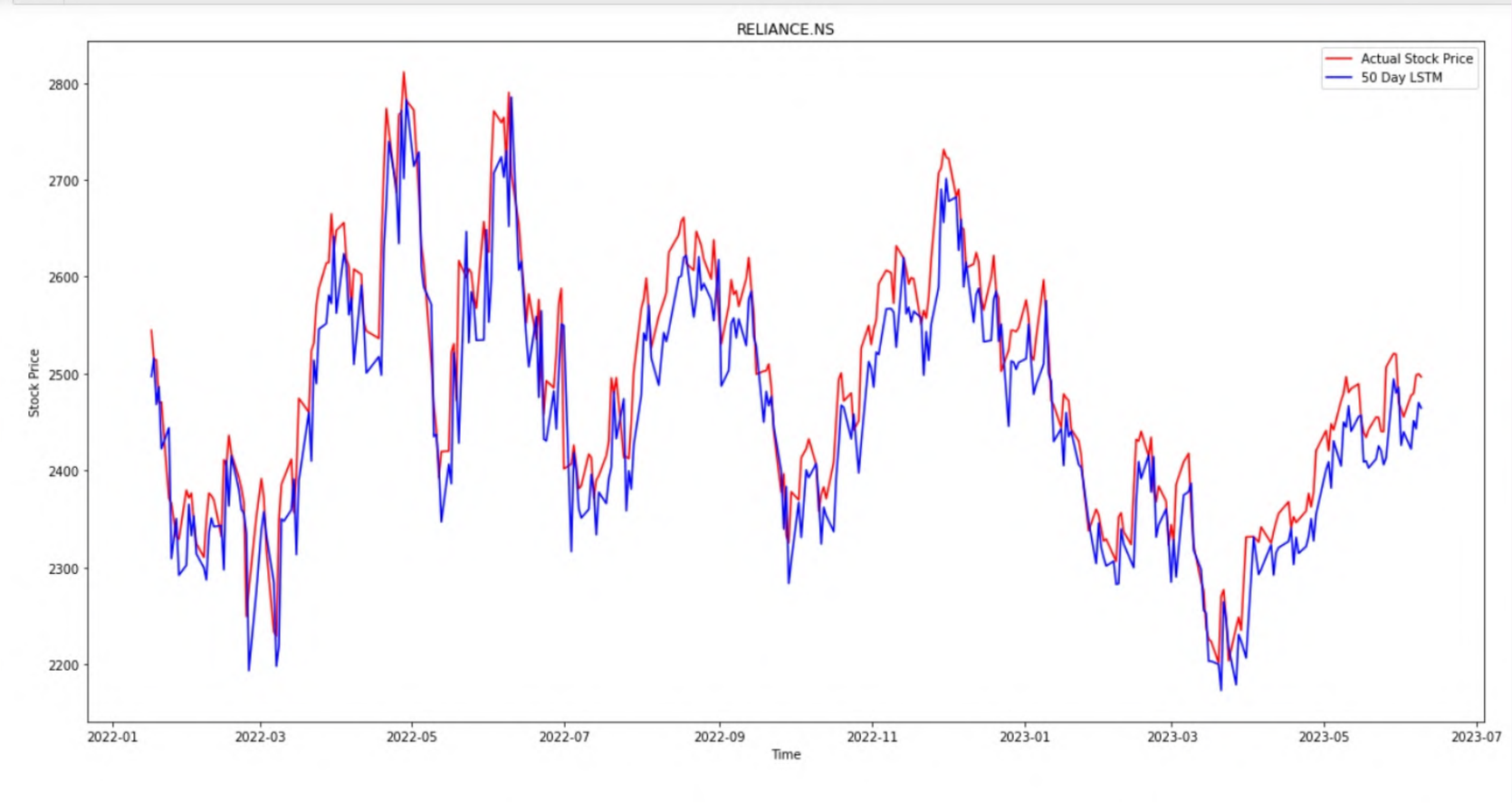
The root mean square error value of EMA for Reliance is 98.71159

RMSE of EMA

The mean and median of root mean square error values for EMA for all companies are 117.37143403375343 and 67.14018271891325 respectively.



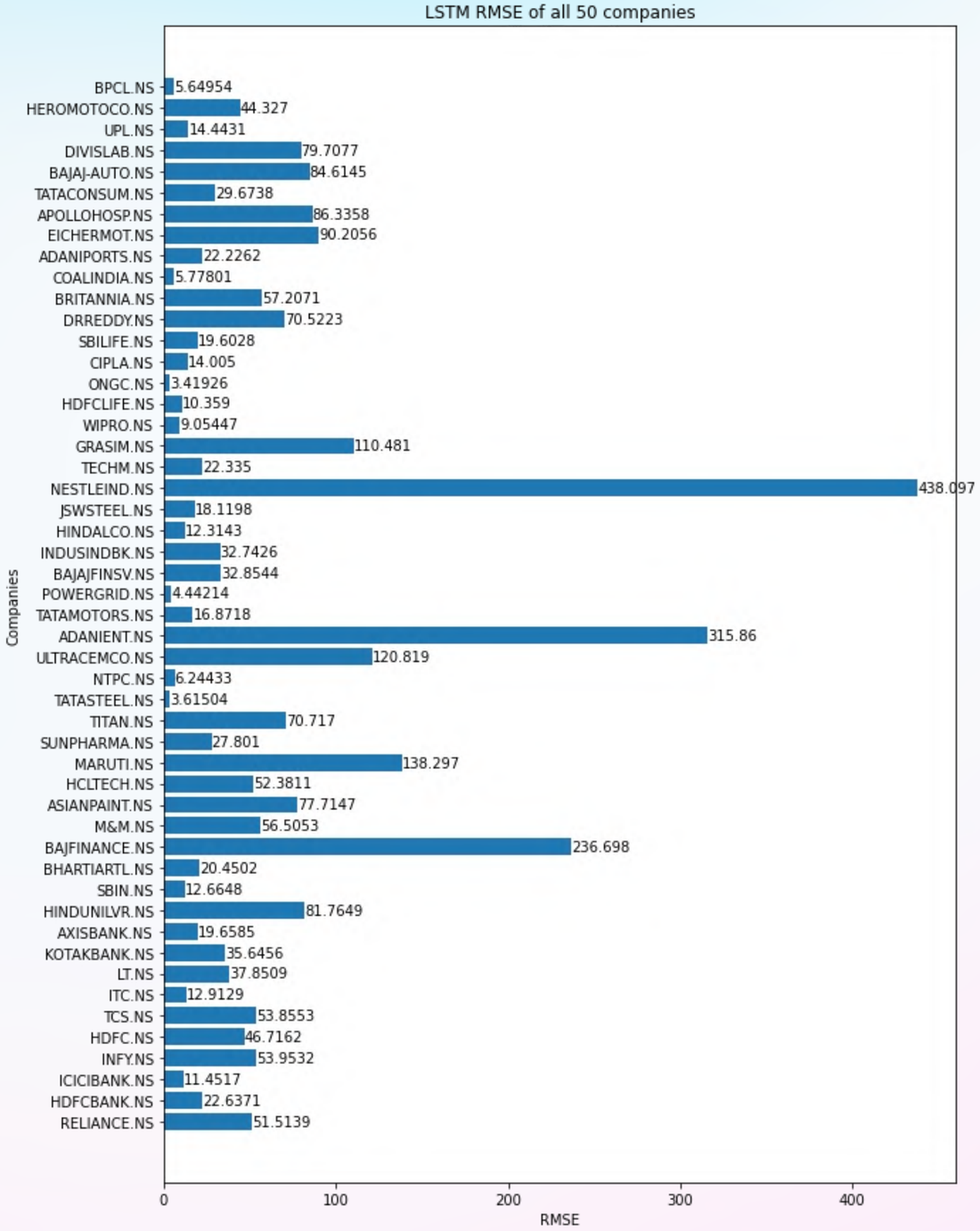
LSTM



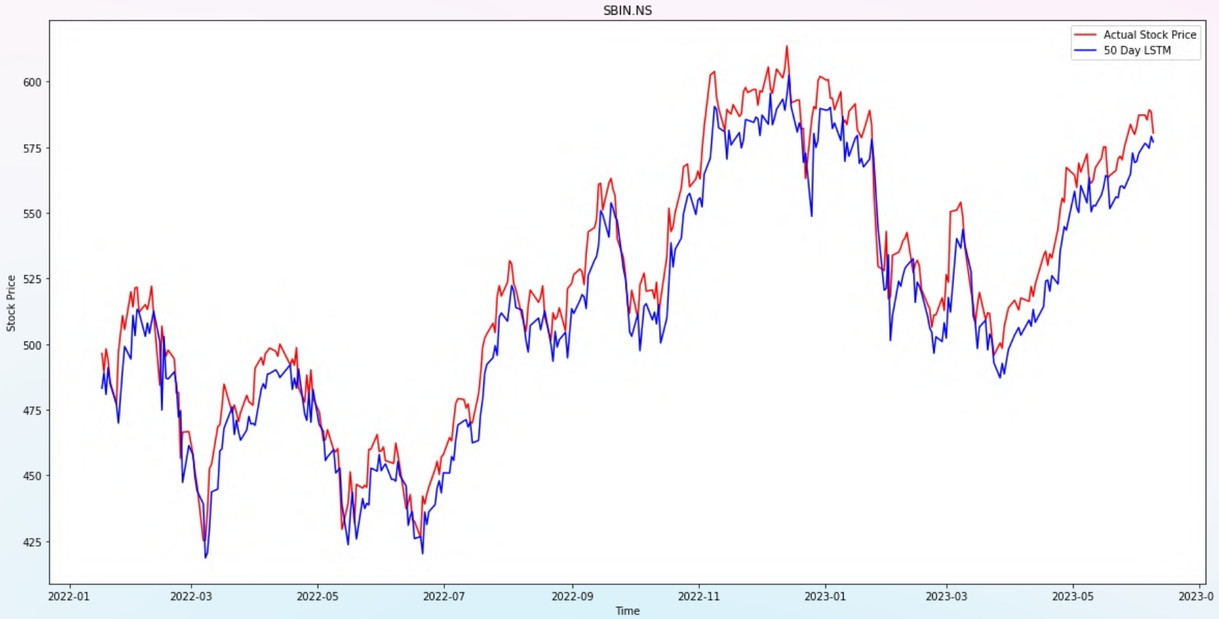
The root mean square error value of LSTM for Reliance is 51.5139

RMSE of LSTM

The mean and median of root mean square error values for LSTM for all companies are 58.062322070154494 and 32.79851883180621 respectively



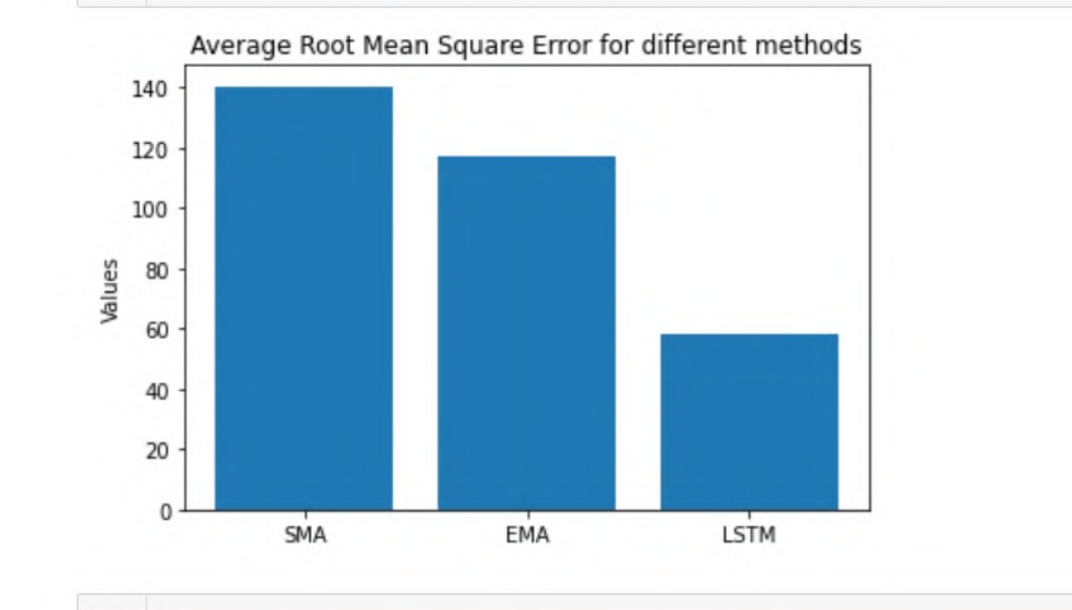
LSTM

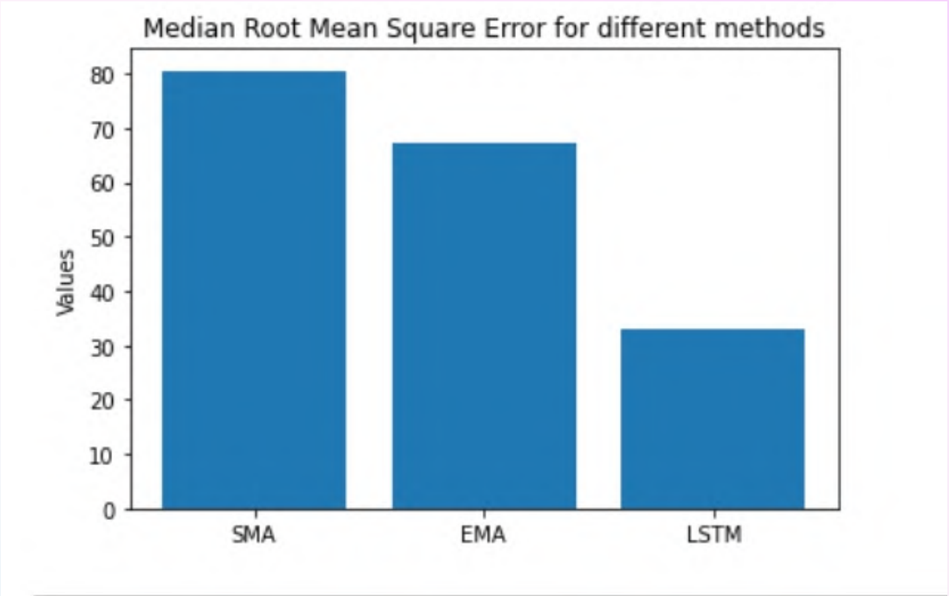


SBI, RMSE- 12.6648

ONGC, RMSE- 3.4193

Result





In conclusion, our analysis indicates that LSTM (Long Short-Term Memory) models outperform simple and exponential moving averages for forecasting task

Conclusion

In conclusion, the stock price prediction using LSTM project demonstrated the effectiveness of using deep learning techniques for predicting stock prices. By processing historical price data with an LSTM neural network, we were able to capture complex patterns in the data and use them to make accurate predictions about future prices. Our model achieves an average mean squared error of 58.062 on the test data and gives visually convincing predictions.

The accuracy of our model could be further improved the training with a higher epoch value and smaller lag value. However, it is important to note that stock prices are influenced by a wide range of factors beyond historical price data, such as company news, macroeconomic events, and investor sentiment. As such, our model should be used as a tool to support decision-making, rather than as a standalone predictor of future stock prices.

References

1. Dev Shah, Haruna Isah and Farhana Zulkernine. 2019. Stock Market Analysis: A Review and Taxonomy of Prediction Techniques, International Journal of Financial Studies. 2019, 7(2), 26

2. Qihang Ma. Comparison of ARIMA, ANN and LSTM for Stock Price Prediction,E3S Web of Conferences 218, 01026 (2020)

3. Di Persio, Luca, and Oleksandr Honchar. 2017. Recurrent Neural Networks Approach to the Financial Forecast of Google Assets. International Journal of Mathematics and Computers in simulation 11: 7–13.

4. Roondiwala, Murtaza, Harshal Patel, and Shraddha Varma. 2017. Predicting Stock Prices Using Lstm. International Journal of Science and Research (IJSR) 6: 1754–56.

5. Fama, Eugene F. 1970. Efficient Capital Markets: A Review of Theory and Empirical Work. The Journal of Finance 25: 383–417

6. Chong, Eunsuk, Chulwoo Han, and Frank C. Park. 2017. Deep learning networks for stock market analysis and prediction: Methodology, data representations, and case studies. Expert Systems with Applications 83: 187–205

7. About LSTM, http://colah.github.io/posts/2015-08-Understanding-LSTMs/