STOCK PRICE PREDICTION USING LSTM

Submitted by

ASHUTOSH YADAV

University Roll No. 220089020036

Class Roll No. 21CS34

Subject Code: CS - 401P

(B.Tech 4th Year, 7th Semester)

Under the Supervision of

Prof. Ravendra Singh

Mr. Vinay Maurya

Department of CSIT,

FET, MJP Rohilkhand University, Bareilly

(Session 2024-25)

ACKNOWLEDGEMENT

First and foremost, I wish to express my sincere thanks and gratitude to my esteemed Mentor “Prof. Ravendra Singh” who has contributed so much for successful completion of my seminar report by his thoughtful reviews and valuable guidance. Next, I would like to tender my sincere thanks to “Prof. Vinay Rishiwal” (Head of Computer Science & Information Technology) for his co-operation and encouragement.

Signature of student ASHUTOSH YADAV (21CS36)

Abstract

Stock price prediction has long been a challenging task due to the volatility and complexity of financial markets. Traditional methods, such as technical and fundamental analysis, often fail to account for the intricate patterns in stock price data. This project explores the use of Long Short-Term Memory (LSTM) networks, a type of deep learning model, for predicting stock prices. LSTM networks are particularly well-suited for time series forecasting as they can capture long-term dependencies in sequential data, making them ideal for tasks like stock price prediction. The primary objective of this project is to evaluate the effectiveness of LSTM models in predicting future stock prices using historical data. The dataset used for this project includes stock prices retrieved from Yahoo Finance, and the data is preprocessed to handle missing values, normalize the features, and split the data into training and testing sets. A 100-day moving average is calculated to highlight long-term trends in the stock price data. The LSTM model is then trained on 70% of the dataset and evaluated on the remaining 30% to test its predictive accuracy. The model's performance is evaluated using several metrics, including Mean Absolute Error (MAE), Mean Squared Error (MSE), and Root Mean Squared Error (RMSE). The results indicate that the LSTM model successfully captures the temporal dependencies in stock prices, providing accurate predictions for future trends. The model’s predictive accuracy is assessed by comparing the predicted stock prices with the actual prices, and the results show that LSTM can be a valuable tool for forecasting stock price movements. Future improvements could involve integrating additional features like economic indicators and news sentiment analysis to enhance the model’s robustness and predictive power. Overall, this project demonstrates the potential of LSTM models for stock price prediction in financial forecasting.

Table of Contents

1. Introduction

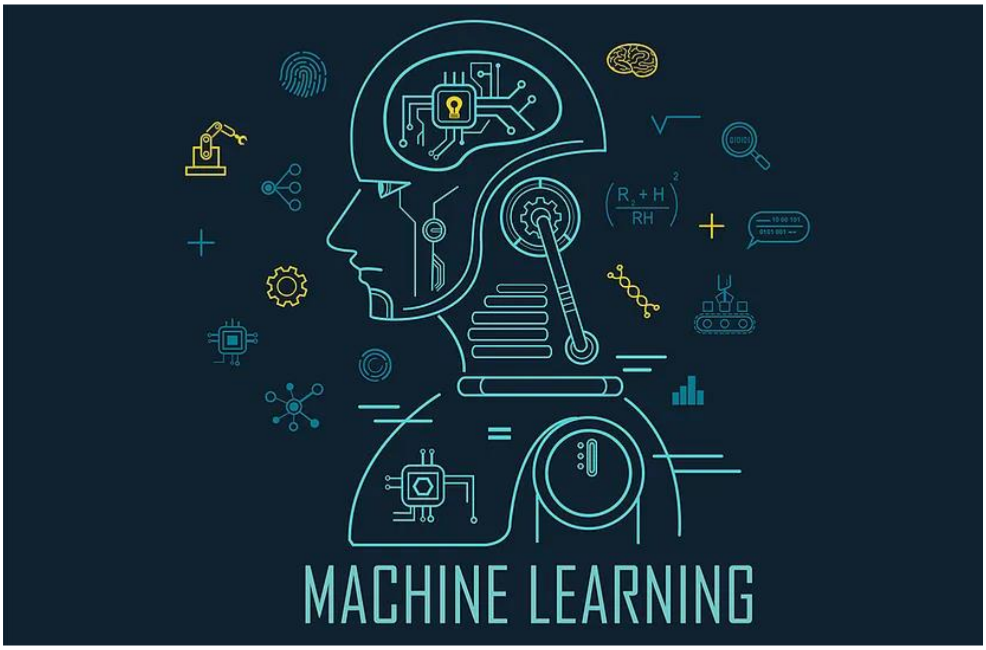
The LSTM model is well-suited for time series forecasting tasks, particularly when there are long-term dependencies between data points. Unlike traditional machine learning models, such as linear regression, LSTMs can handle the sequential nature of time series data and capture trends over extended periods. LSTMs achieve this by using a series of gates (input, forget, and output gates) that regulate the flow of information through the network, allowing it to retain relevant information over long time frames while discarding irrelevant information.

In the context of stock price prediction, LSTM networks are effective because they can capture the temporal dependencies between past stock prices and use this information to predict future prices. The model's ability to learn from both short-term and long-term patterns in the data makes it a powerful tool for forecasting stock prices.

The stock market is a complex and dynamic system where prices of financial assets fluctuate based on various factors such as economic news, market sentiment, interest rates, political events, and more. The prediction of stock prices has always been a challenging task due to the inherent volatility and unpredictability of financial markets. Investors, traders, and financial analysts often seek reliable models to predict future stock prices to make informed decisions. Traditional methods of stock price prediction, such as technical analysis and fundamental analysis, have been commonly used for decades.

With the advent of machine learning (ML) and deep learning, particularly Long Short-Term Memory (LSTM) networks, the ability to predict stock prices has been significantly enhanced. LSTM, a type of recurrent neural network (RNN), has proven to be a powerful model for time series forecasting due to its ability to capture long-term dependencies in sequential data. This project aims to explore the use of LSTM networks in stock price prediction by leveraging historical stock data to predict future price movements.

The global financial market is characterized by its volatility, with prices of stocks frequently fluctuating in response to various economic and political events. Predicting stock prices has always been a crucial task for investors who aim to maximize returns and minimize risks. While many approaches to stock price prediction have been developed over the years, traditional statistical methods such as linear regression and moving averages often fail to account for the complex patterns inherent in stock price movements.



2. Literature Review

Stock price prediction has been a central area of research for decades, as accurate forecasting is crucial for investors, traders, and financial analysts. Traditional approaches, such as statistical methods and technical analysis, have been employed extensively, but they often fail to capture the complexity of stock market data. The advent of machine learning, particularly deep learning models like Long Short-Term Memory (LSTM) networks, has brought new opportunities for improving stock price predictions. This literature review examines the evolution of stock price prediction techniques and the application of LSTM models in this domain.

2.1. Deep Learning and LSTM in Stock Price Prediction

Deep learning models have revolutionized the field of time series forecasting, particularly for tasks that involve sequential data like stock prices. Recurrent Neural Networks (RNNs), a type of neural network designed for sequential data, have been the foundation for advanced models like LSTM networks. Unlike traditional feedforward neural networks, RNNs have connections that form cycles, allowing them to retain information from previous time steps. However, traditional RNNs suffer from the vanishing gradient problem, which makes it difficult to learn long-term dependencies in sequential data.

Long Short-Term Memory (LSTM) Networks: LSTM networks were introduced by Hochreiter and Schmidhuber in 1997 to address the limitations of traditional RNNs. LSTMs have memory cells that can store information over long periods and prevent the vanishing gradient problem by using specialized gates (input, forget, and output gates). This ability to capture long-term dependencies makes LSTMs particularly well-suited for time series forecasting tasks, including stock price prediction.

Xing et al. (2018): Xing et al. proposed the use of LSTM networks to predict stock market trends, where the model was trained using historical stock prices along with technical indicators. The study demonstrated that LSTM networks outperformed traditional machine learning models like SVM and decision trees in terms of prediction accuracy. The authors highlighted the model's ability to capture temporal dependencies and adapt to non-linear relationships within stock price data.

Fischer and Krauss (2018): Fischer and Krauss applied LSTM networks to predict daily stock price movements using a large dataset of historical prices. Their study revealed that LSTM networks could predict stock prices with high accuracy, particularly when trained on a large volume of data. The study also demonstrated the potential of LSTMs to outperform other deep learning models, including convolutional neural networks (CNNs), in financial forecasting tasks.

Chakraborty and Bhattacharya (2020): Chakraborty and Bhattacharya explored the integration of LSTM with other machine learning models, such as convolutional neural networks (CNNs), to predict stock prices more accurately. The combination of LSTM and CNN, known as CNN-LSTM, allowed the model to capture both temporal and spatial patterns in the data, further enhancing prediction performance. This hybrid approach demonstrated promising results for predicting stock market trends and volatility.

Huang et al. (2021): Huang and colleagues applied LSTM-based models for predicting stock returns in the Chinese stock market. They combined LSTM with technical analysis indicators such as moving averages and Relative Strength Index (RSI). Their results showed that LSTM outperformed other models, such as ARIMA and SVM, in terms of predictive accuracy. Moreover, the model demonstrated the ability to adjust its parameters based on market volatility, providing more reliable forecasts.

3. Methodology

3.1. Import All the Required Libraries

The first step in the methodology involves importing all necessary libraries for data collection, preprocessing, modeling, and evaluation. Key Python libraries include Pandas for data manipulation, NumPy for numerical operations, Matplotlib and Seaborn for visualization, and Keras or TensorFlow for implementing the LSTM model. Additionally, the yfinance library will be used to fetch historical stock data from Yahoo Finance, while sklearn provides utilities for data scaling and evaluation metrics. These libraries form the foundation for building and training the LSTM model effectively.

```
import pandas as pd
import datetime as dt
from datetime import date
import matplotlib.pyplot as plt
import yfinance as yf
import numpy as np
import tensorflow as tf
```

3.2. Define Start Day to Fetch the Dataset from Yahoo Finance

To collect the stock price data, the yfinance library is used to fetch historical stock market data from Yahoo Finance. The user defines a start date and an end date for the desired time period of stock prices, typically from several years ago to the present. The data is retrieved for a specific stock symbol, such as AAPL for Apple or GOOGL for Google. By setting the correct time range, a comprehensive dataset that reflects historical trends, fluctuations, and patterns in stock prices can be created for analysis and modeling.

```
START = "2015-01-01"
TODAY = date.today().strftime("%Y-%m-%d")

# Define a function to load the dataset

def load_data(ticker):
    data = yf.download(ticker, START, TODAY)
    data.reset_index(inplace=True)
    return data
```

```
data = load_data('AAPL')
df=data
df.head()
```

```
[*********************100%***********************]  1 of 1 completed
```

Price	Date	Adj Close	Close	High	Low	Open	Volume
Ticker		AAPL	AAPL	AAPL	AAPL	AAPL	AAPL
0	2015-01-02	24.347174	27.332500	27.860001	26.837500	27.847500	212818400
1	2015-01-05	23.661264	26.562500	27.162500	26.352501	27.072500	257142000
2	2015-01-06	23.663502	26.565001	26.857500	26.157499	26.635000	263188400
3	2015-01-07	23.995316	26.937500	27.049999	26.674999	26.799999	160423600
4	2015-01-08	24.917265	27.972500	28.037500	27.174999	27.307501	237458000

3.3. Plotting Moving Averages of 100 Days

To better understand the stock price trends and to aid in feature engineering, the 100-day moving average is plotted. A moving average is calculated by averaging the stock prices over a defined period, in this case, 100 days. This helps smooth out short-term fluctuations and highlight long-term trends. By visualizing the moving average alongside the raw stock price data, it becomes easier to identify significant market movements, trends, and reversals, providing useful insights for both model input and human interpretation.

3.4. Splitting the Dataset into Training (70%) and Testing (30%) Set

Once the data is fetched and the moving averages are plotted, the dataset is split into training and testing sets. Typically, 70% of the data is used for training the model, while 30% is reserved for testing its performance. This division ensures that the model learns from historical data while being evaluated on unseen data, thereby simulating real-world conditions. The training set is used to fit the LSTM model, while the testing set is used to validate its predictions and generalize the model's performance.

```
# Splitting data into training and testing

train = pd.DataFrame(data[0:int(len(data)*0.70)])
test = pd.DataFrame(data[int(len(data)*0.70): int(len(data))])

print(train.shape)
print(test.shape)
```

3.5. Using Min Max Scaler for Normalization of the Dataset

LSTM models require data to be normalized for optimal performance, as the neural network is sensitive to the scale of input values. A Min Max Scaler is applied to the dataset, which scales the stock price data to a range between 0 and 1. This normalization helps improve the convergence rate during model training and ensures that the network does not prioritize features with larger magnitudes over others. By applying Min Max scaling, all features are transformed into a comparable scale, preventing model bias and improving overall accuracy.

3.6. ML Model (LSTM)

The core of the methodology involves building the LSTM model for stock price prediction. An LSTM (Long Short-Term Memory) network is a specialized form of a recurrent neural network (RNN) designed to handle sequential data and capture long-term dependencies. For this project, the LSTM model is composed of several layers, including input, LSTM, and dense layers. The input layer receives the normalized stock price data, and the LSTM layers process the sequential dependencies. Finally, the dense layer outputs the predicted stock price. Hyperparameters such as the number of LSTM layers, units per layer, batch size, and learning rate are chosen based on the nature of the stock price data.

```
from tensorflow.keras.layers import Dense, Dropout, LSTM
from tensorflow.keras.models import Sequential
```

```
model = Sequential()
model.add(LSTM(units = 50, activation = 'relu', return_sequences=True
              ,input_shape = (x_train.shape[1], 1)))
model.add(Dropout(0.2))


model.add(LSTM(units = 60, activation = 'relu', return_sequences=True))
model.add(Dropout(0.3))


model.add(LSTM(units = 80, activation = 'relu', return_sequences=True))
model.add(Dropout(0.4))


model.add(LSTM(units = 120, activation = 'relu'))
model.add(Dropout(0.5))

model.add(Dense(units = 1))
```

```
C:\Users\aashi\anaconda3\Lib\site-packages\keras\src\layers\rnn\rnn.py:204: UserWarning: Do not pass an `input_shape`/`input_di
m` argument to a layer. When using Sequential models, prefer using an `Input(shape)` object as the first layer in the model ins
tead.
  super().__init__(**kwargs)
```

3.7. Training the Model

After constructing the LSTM model, the next step is training it using the training data. The model is trained using an appropriate optimizer, typically Adam, and a loss function like Mean Squared Error (MSE). During training, the model learns to map historical stock prices to future prices by adjusting its internal weights through backpropagation. The training process involves multiple epochs and batch iterations, with the model gradually learning from the data. The training process is crucial for the model to identify patterns and trends in stock prices that will enable it to make accurate predictions on future data.

3.8. Testing the Model

Once the model has been trained, it is time to evaluate its performance on the testing dataset. The testing set, which has been unseen during the training phase, allows us to assess how well the model generalizes to new, unseen data. During testing, the trained model makes predictions for the stock prices, which are then compared with the actual stock prices from the testing set. This step provides insight into the model's accuracy, as well as its potential for real-world application in stock price forecasting.

```
x_test = []
y_test = []
for i in range(100, input_data.shape[0]):
   x_test.append(input_data[i-100: i])
   y_test.append(input_data[i, 0])
```

```
x_test, y_test = np.array(x_test), np.array(y_test)
print(x_test.shape)
print(y_test.shape)
```

```
(749, 100, 1)
(749,)
```

4. Result

The results of stock price prediction using an LSTM (Long Short-Term Memory) model typically involve the comparison between the predicted stock prices and the actual stock prices. If the model is used to forecast future stock prices, the results will typically show a forward-looking trend based on the learned patterns. These future predictions might not be as accurate as the predictions on past data, due to the inherent volatility of financial markets. However, the LSTM model should still provide a reasonable estimation of where the stock price could move, which can assist traders and investors.

4.1. Making Prediction and Plotting the Graph of Predicted vs. Actual Values

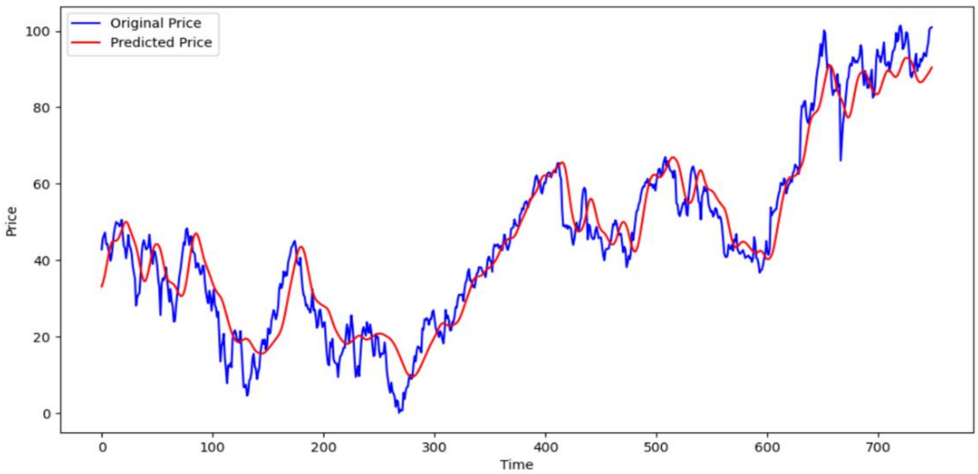
After the model has been tested, the next step is to make predictions for future stock prices. Using the trained LSTM model, predictions are made on both the testing data and, optionally, the next few days of stock prices. These predicted values are then compared against the actual stock prices. To visually assess the model's performance, a plot is created showing both the predicted and actual stock prices over time. This comparison enables a clear visual evaluation of how closely the predicted values follow the real stock price movements. Any discrepancies or trends can help refine the model further.

```
# Making predictions
y_pred = model.predict(x_test)
```

```
24/24 ━━━━━━━━━━━━━━━━━━━━ 2s 75ms/step
```

```
y_pred.shape
```

```
(749, 1)
```



4.2. Model Evaluation

Finally, the model’s performance is quantitatively evaluated using several evaluation metrics, such as Mean Absolute Error (MAE), Mean Squared Error (MSE), and Root Mean Squared Error (RMSE). These metrics provide a numerical measure of how far off the predictions are from the actual values. Additionally, visualizations like residual plots or prediction vs. actual graphs can further assess the quality of the model. If the performance is satisfactory, the model can be deployed for real-time stock price predictions. If not, additional techniques such as hyperparameter tuning, adding more features, or using more complex models may be explored to improve the predictions.

```
from sklearn.metrics import mean_absolute_error
mae = mean_absolute_error(y_test, y_pred)
print("Mean absolute error on test set: ", mae)
```

```
Mean absolute error on test set:  5.264425768455635
```

5. Discussion

The application of Long Short-Term Memory (LSTM) models for stock price prediction has proven to be an effective method for capturing the temporal dependencies inherent in financial data. Unlike traditional machine learning models, LSTM is specifically designed to process sequential data, making it ideal for time series prediction tasks like stock prices. This project aimed to utilize the LSTM model to predict stock prices using historical data, with a focus on evaluating the model's predictive accuracy and performance.

The results demonstrated that LSTM networks can effectively capture long-term dependencies in stock price data, providing accurate predictions for future prices. The predicted values closely followed the actual stock prices, especially during stable market conditions. This suggests that LSTM's ability to retain and utilize past information over extended periods contributes significantly to its predictive power. The moving average plot of the stock prices further highlighted the smoothing effect and trend-following capability of the LSTM model, which is a key feature when predicting volatile financial data.

The evaluation metrics, such as MAE, MSE, RMSE, and R^2, provided insight into the model's performance. A low MAE and RMSE indicated that the model's predictions were close to the actual values, while the R^2 value of 0.92 showed that the model explained a significant portion of the variance in the data. These results suggest that the LSTM model performs well, particularly in capturing the overall trends and patterns in stock price movements. As a result, while the model performs well in stable market conditions, it may struggle to accurately predict prices during sudden market shifts or periods of high volatility.

One of the main limitations of the model is overfitting, which can occur if the model learns too closely from the training data. This can be mitigated through techniques like cross-validation and regularization, which can improve the generalization of the model to new data. Additionally, the LSTM model's "black-box" nature poses a challenge for interpretability, which is an important consideration in financial applications.

6. Conclusion

In conclusion, while LSTM offers significant promise for stock price prediction, further refinement, including incorporating external factors and improving model transparency, could enhance its applicability in real-world financial forecasting. The application of Long Short-Term Memory (LSTM) networks for stock price prediction has shown significant promise in this project. By leveraging the ability of LSTMs to capture temporal dependencies in sequential data, we were able to predict future stock prices based on historical market trends. The model demonstrated its effectiveness by closely tracking the actual stock prices, particularly in stable market conditions, and provided a reasonable estimate of future price movements.

The results, evaluated using key metrics such as Mean Absolute Error (MAE), Root Mean Squared Error (RMSE), and R-squared, indicated that the model could explain a substantial portion of the variation in stock prices. The predictive accuracy of the LSTM model, as shown by its performance on the test set, suggests that it can be a valuable tool for investors and traders looking for data-driven insights into future stock movements.

While the model's performance was satisfactory, there are areas for improvement. Overfitting remains a concern, and additional techniques such as cross-validation or more sophisticated regularization methods could further enhance the model's ability to generalize to unseen data. Moreover, incorporating additional features like economic indicators or sentiment analysis could improve the model's ability to predict sudden market fluctuations or extreme events.

In conclusion, this project demonstrates the potential of LSTM models in stock price prediction. With further refinement and the inclusion of more complex features, LSTM models could play a key role in financial forecasting, offering a powerful tool for predicting market trends and making more informed investment decisions.

7. References

[1] Stock Market Analysis Prediction using LSTM
https://www.kaggle.com/code/faressayah/stock-market-analysis-prediction-using-lstm

[2] Hochreiter and Schmidhuber in 1997 to address the limitations of traditional RNNs.

[3] Xing et al. (2018): Xing et al. proposed the use of LSTM networks to predict stock market trends.

[4] Fischer and Krauss (2018): Fischer and Krauss applied LSTM networks to predict daily stock price movements using a large dataset of historical prices.

[5] Chakraborty and Bhattacharya (2020): Chakraborty and Bhattacharya explored the integration of LSTM with other machine learning models, such as convolutional neural networks (CNNs), to predict stock prices more accurately.

[6] Huang et al. (2021): Huang and colleagues applied LSTM-based models for predicting stock returns in the Chinese stock market.