MINOR-2 PROJECT
Mid-Semester REPORT

For

Stock Forecasting and Visualization Using LSTM

Submitted By

Course	SAP ID	Name
B.Tech CSE-CCVT	500091657	Divyansh Jha
B.Tech CSE-CCVT	500091318	Rimjhim Gupta
B.Tech CSE-CCVT	500091835	Sanaya Bhardwaj

Department of Cloud Software and Operations

School Of Computer Science

UNIVERSITY OF PETROLEUM & ENERGY STUDIES

DEHRADUN- 248007. Uttarakhand

Dr. Arjun Arora
Project Guide

Dr Hitesh Kumar Sharma
Cluster Head

1. Project Title

Stock forecasting and visualization using LSTM

2. Abstract

This project explores the application of Long Short-Term Memory (LSTM) neural networks in the context of stock market analysis, specifically focusing on visualization and forecasting. The stock market is a complex system influenced by numerous factors, making accurate prediction challenging. Traditional methods often fail to capture the non-linear and dynamic nature of stock price movements. LSTM networks, a type of recurrent neural network (RNN), have shown promise in capturing temporal dependencies and patterns in sequential data, making them suitable for time series forecasting tasks. In this study, we propose a framework for visualizing stock data and forecasting future prices using LSTM networks. We demonstrate the effectiveness of our approach through empirical experiments on real-world stock market data, highlighting the potential for improved decision-making and risk management in financial markets.

3. Introduction

The stock market, as a cornerstone of the global economy, serves as a platform for companies to raise capital and for investors to allocate resources efficiently. However, the inherent volatility and complexity of financial markets pose significant challenges for accurate prediction and decision-making. Traditional time series forecasting techniques, such as autoregressive models and moving averages, often struggle to capture the dynamic and non-linear nature of stock price movements. Consequently, there is a growing interest in exploring alternative methodologies, particularly those rooted in machine learning and deep learning.

In recent years, deep learning models, and in particular, Long Short-Term Memory (LSTM) networks, have garnered attention for their ability to capture complex patterns and temporal dependencies in sequential data. LSTM networks, equipped with memory cells capable of retaining information over extended time intervals, offer promise in modeling and forecasting time series data, including stock prices. Leveraging the power of LSTM networks, our research aims to develop a comprehensive framework for visualizing historical stock data and forecasting future prices with improved accuracy and reliability.

In this paper, we propose a multi-faceted approach that begins with the visualization of historical stock data to gain insights into past trends and patterns. Subsequently, we introduce LSTM networks as the core forecasting engine, trained on historical data to predict future stock prices. We also discuss various preprocessing techniques, model architectures, and evaluation metrics employed in our framework. Through empirical experiments conducted on real-world stock market datasets, we evaluate the performance of our approach and compare it against baseline methods. Additionally, we discuss the implications of our findings for investors, traders, and financial analysts, emphasizing the potential for improved decision-making and risk management in the dynamic landscape of financial markets.

4. Literature Review

The first focus of our literature review was to evaluate different algorithms and models to determine whether stock price predictions could be made on real stock prices [1]. However, as we were unable to find any potential adaptation of these for stock price prediction, we then decided to look at the existing systems [2], analyze the major drawbacks of the same, and see if we could improve upon them. In [3], they used one of the most useful forecasting techniques, the use of recurrent neural network (RNN) and long-term and short-term memory (LSTM) unit to help investors, analysts, or any person who is interested in investing in the stock market, and to provide them with a good knowledge of the future status of the stock market. In [4], they proposed a method to predict the stock price with distributed representations of the reported information, and taking into account the interaction between multiple companies in the same industry. On their way to a regular network forecast changes, time-series fluctuations on the stock price. The experimental results show that distributed text information is far better than digital, data-only methods and the bag of text-based methods, LSTM can capture the time series more than other types of input data, and the company is an effective stock price forecast. LSTM is one of the most popular types of RNN. Graves et al. [5] and Pan et al. [6] proposed an LSTM method to obtain useful information and predict immature stock markets from financial time series.

5. Problem Statement

Given real-time stock market data fetched from an API, the task is to develop an LSTM recurrent neural network model capable of effectively visualizing historical stock data and accurately forecasting future stock prices.

6. Objectives

- To develop an LSTM recurrent neural network model capable of predicting stock prices based on real-time data fetched from an API.
- To enhance the understanding of stock market data patterns through effective visualization techniques.
- To assess the accuracy and reliability of the LSTM model in forecasting future stock prices.
- To explore the practical implications of the proposed approach for decision-making and risk management in financial markets.

7. Methodology

Sprint 1: Requirement Analysis

Goal: Review research papers and define project requirements based on insights gained.

- **Activities:**
    - Conduct a literature review of research papers on stock market prediction using LSTM or related techniques.
    - Summarize key findings and insights from relevant research.

Sprint 2: Data Prep & Deployment

Goal: Collect historical stock data, preprocess, and deploy databases on AWS.

- **Activities:**
    - Implement data collection using Yahoo Finance API.
    - Handle missing values and outliers in the dataset.
    - Deploy the database on AWS RDS.

Sprint 3: Train LSTM Model

Goal: Design, train, and save the LSTM model for stock prediction.

- **Activities:**
    - Split data into training and testing sets(80/20).
    - Design LSTM models and specify hyperparameters.
    - Train the model with historical stock data.

Sprint 4: Setup Django & Deploy Web App

Goal: Establish the Django framework and deploy the web application on AWS EC2.

- **Activities:**
    - Set up the Django framework for the web application.
    - Develop user interfaces for user interaction.
    - Deploy the web application on AWS EC2 for accessibility.

Sprint 5: User Input, Prediction & Visualization

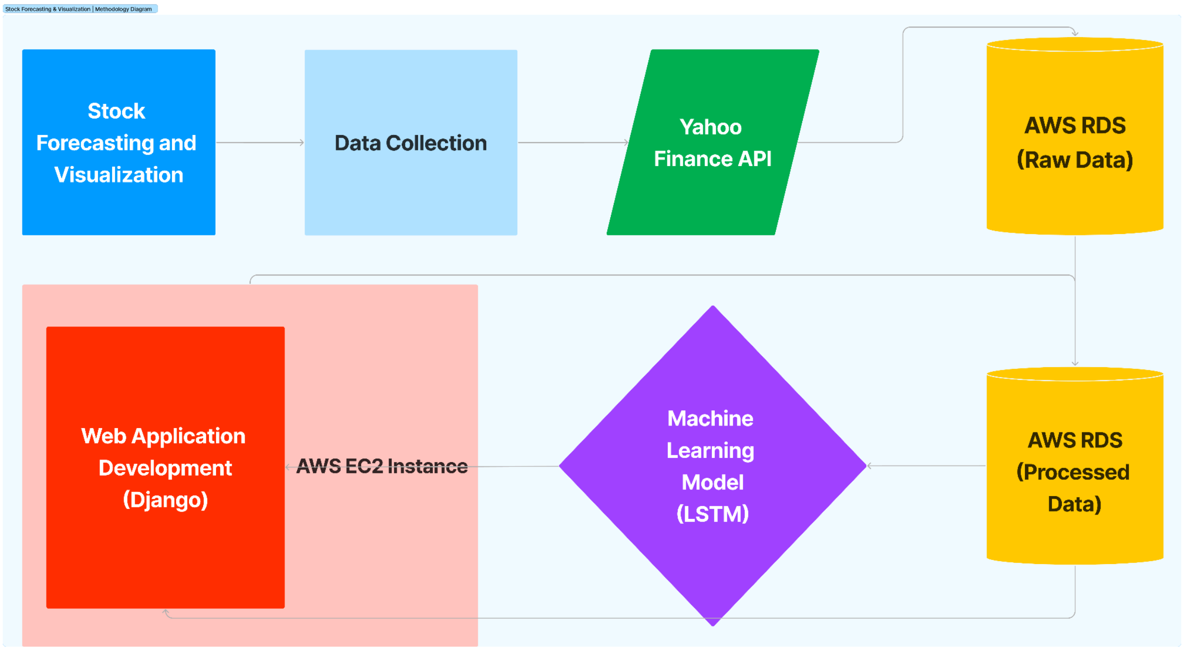
Goal: Enable user input, make stock predictions, and visualize using Plotly.

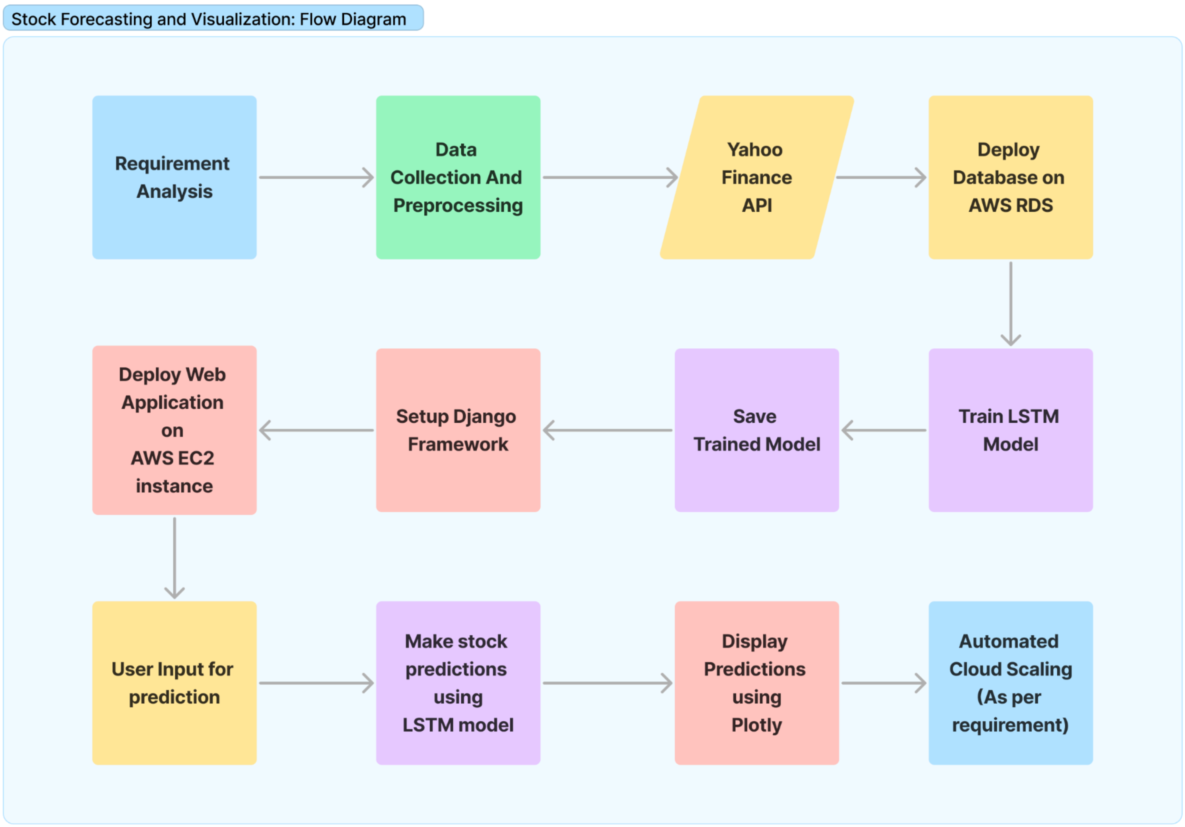
- **Activities:**
    - Implement user input functionality for predicting specific stocks.
    - Utilize the trained LSTM model to make accurate stock predictions.
    - Implement interactive visualization using Plotly for user-friendly representation.

Sprint 6: Automated Cloud Scaling

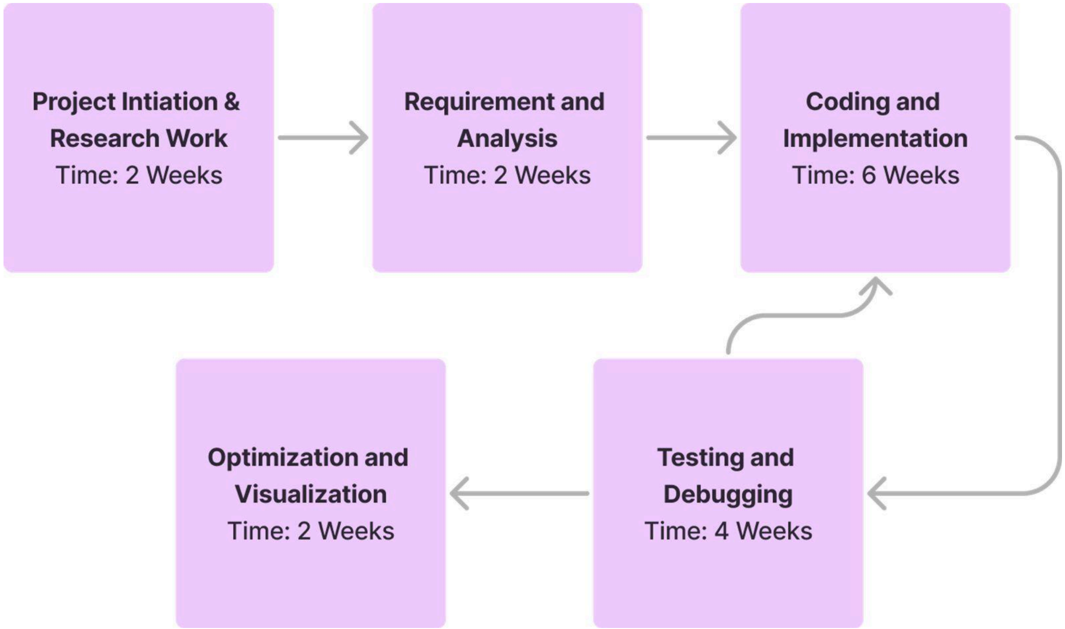
Goal: Implement automated cloud scaling based on system requirements.

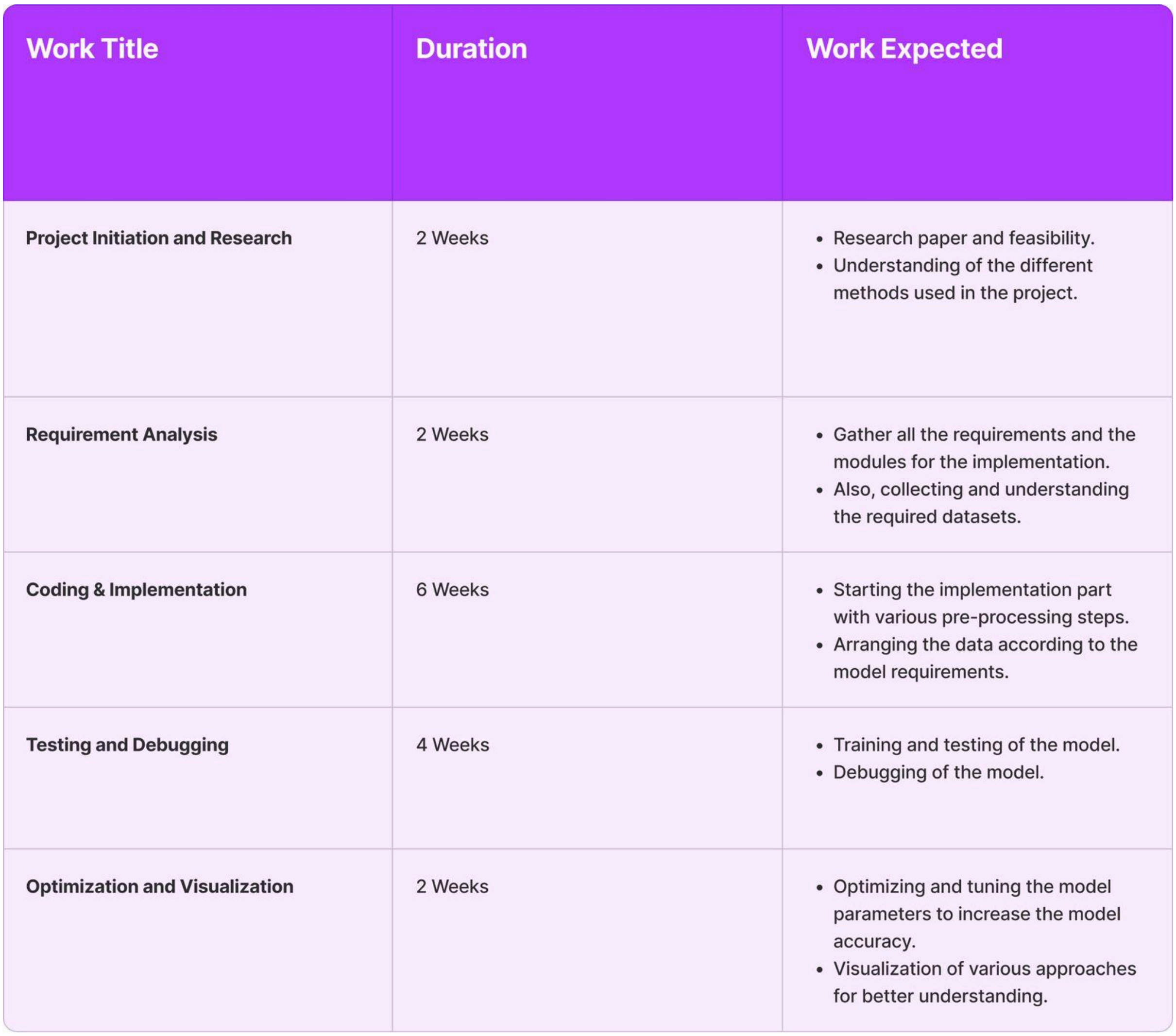
- **Activities:**
  - Analyze system requirements for scaling.
  - Implement automated scaling mechanisms on AWS for enhanced performance.
  - Test and validate the effectiveness of automated scaling.


Stock Forecasting & Visualization | Methodology Diagram
Stock Forecasting and Visualization
Data Collection
Yahoo Finance API
AWS RDS (Raw Data)
Web Application Development (Django)
AWS EC2 Instance
Machine Learning Model (LSTM)
AWS RDS (Processed Data)


Stock Forecasting and Visualization: Flow Diagram
Requirement Analysis
Data Collection And Preprocessing
Yahoo Finance API
Deploy Database on AWS RDS
Train LSTM Model
Save Trained Model
Setup Django Framework
Deploy Web Application on AWS EC2 instance
User Input for prediction
Make stock predictions using LSTM model
Display Predictions using Plotly
Automated Cloud Scaling (As per requirement)

8. PERT Chart





Work Title	Duration	Work Expected
Project Initiation and Research	2 Weeks	• Research paper and feasibility. • Understanding of the different methods used in the project.
Requirement Analysis	2 Weeks	• Gather all the requirements and the modules for the implementation. • Also, collecting and understanding the required datasets.
Coding & Implementation	6 Weeks	• Starting the implementation part with various pre-processing steps. • Arranging the data according to the model requirements.
Testing and Debugging	4 Weeks	• Training and testing of the model. • Debugging of the model.
Optimization and Visualization	2 Weeks	• Optimizing and tuning the model parameters to increase the model accuracy. • Visualization of various approaches for better understanding.

9. Comparison Analysis

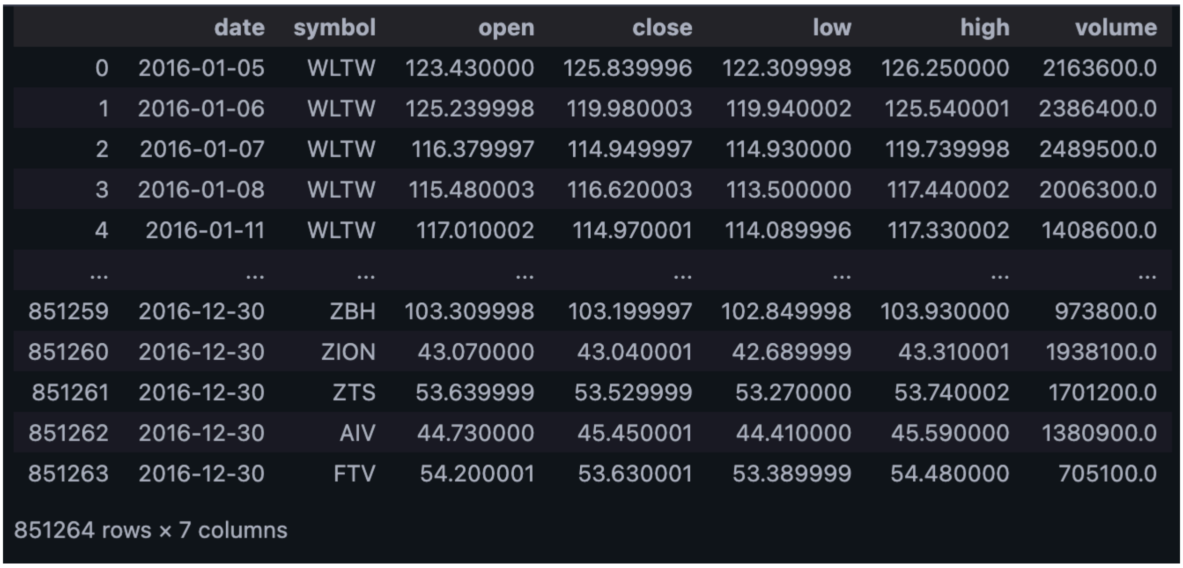
- This work comparatively analyzes the implementations of LSTM (Long Short Term Memory) and three kernels of SVR (Support Vector Regressor), namely linear kernel, RBF (Radial Basis Function) kernel and polynomial kernel. The LSTM model presented here works very well although since the only feature here is the opening price of the stocks, the model cannot predict the actual price of the next day. But, LSTM does find a reasonable pattern as can be seen in the paper which helps it closely follow the line of actual stock prices. (Paper 1)
- A comparative study of algorithms: Scaled Unscaled LR, SVM, LSTM are discussed. In machine learning, LR is a fundamental approach by which a linear trend can be obtained. But SVM has modern features such as high exactness and certainty. LSTM is a kind of recurrent neural network (RNN)and uses an appropriate gradient algorithm. The factors considered for choosing the best algorithm were opening price, closing price, adjacent close, volume. Overall, LSTM performs better, this is due to its ability to recollect or forget the information in an efficient manner. (Paper 2)
- To train and predict stock price and stock price sub correlation, the ARIMA (autoregressive integrated moving average model) and LSTM (long short-term memory) neural network models are used. LSTM model predicts better than ARIMA. (Paper 3)
- A comparative study of two very promising artificial neural network models namely a Long Short-Term Memory (LSTM) recurrent neural network (RNN) and a deep neural network (DNN) in forecasting the daily and weekly movements of the Indian BSE Sensex index. It seems that both the DNN and the LSTM RNN are suitable for this type of task. With the help of strategies to avoid overfitting, both models were able to generalize well to a new, more volatile data set. Finally, the LSTM RNN outperformed the DNN in making weekly predictions. With the inclusion of more attributes, the LSTM RNN shows promise for finding underlying trends and making longer term predictions on volatile stock data sets. (Paper4).
- The proposed deep LSTMP RNN architecture outperforms standard LSTM networks and DNNs and makes more effective use of the model parameters by addressing the computational efficiency needed for training large networks. We also show for the first time that LSTM RNN models can be quickly trained using ASGD distributed training. (Paper5).
- The evaluation focused on the task of sequence modeling on a number of datasets including polyphonic music data and raw speech signal data.The evaluation clearly demonstrated the superiority of the gated units; both the LSTM unit and GRU,over the traditional tanh unit. This was more evident with the more challenging task of raw speech signal modeling. However, we could not make a concrete conclusion on which of the two gating units was better. (Paper6).

10. Backend Implementation

Modules used:

- **pandas**: Used for data manipulation and analysis.
- **MinMaxScaler** from **sklearn.preprocessing**: Used to scale feature values to a range between 0 and 1, which is often beneficial for neural network models like LSTM.
- **train_test_split** from **sklearn.model_selection**: Used to split the dataset into training and testing sets.
- **numpy**: Used for numerical operations.
- **tensorflow**: Deep learning library for building and training neural networks.
- **Sequential** from **tensorflow.keras.models**: Used to create a sequential model, where layers are stacked sequentially.
- **LSTM** and **Dense** from **tensorflow.keras.layers**: Used to define the LSTM and fully connected (dense) layers in the neural network architecture.
- **EarlyStopping** from **tensorflow.keras.callbacks**: Used for early stopping during model training to prevent overfitting.
- **matplotlib.pyplot**: Used for data visualization, especially for plotting the model's performance metrics.
- **mean_squared_error**, **mean_absolute_error**, **r2_score** from **sklearn.metrics**: Used to evaluate the performance of the model.

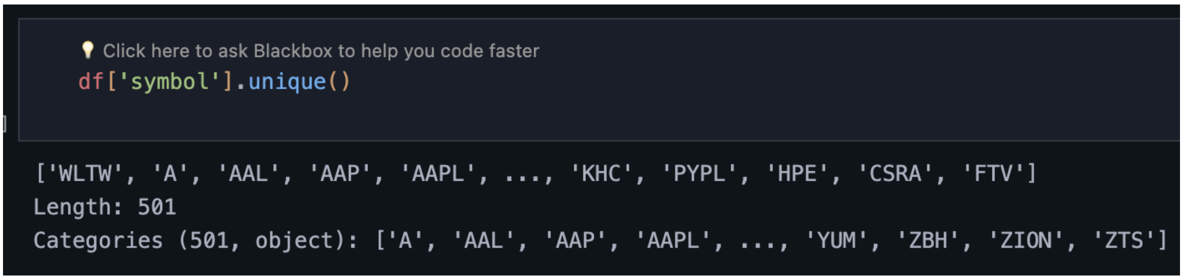
Dataset Used: NYSE (2010-2017)



	date	symbol	open	close	low	high	volume
0	2016-01-05	WLTW	123.430000	125.839996	122.309998	126.250000	2163600.0
1	2016-01-06	WLTW	125.239998	119.980003	119.940002	125.540001	2386400.0
2	2016-01-07	WLTW	116.379997	114.949997	114.930000	119.739998	2489500.0
3	2016-01-08	WLTW	115.480003	116.620003	113.500000	117.440002	2006300.0
4	2016-01-11	WLTW	117.010002	114.970001	114.089996	117.330002	1408600.0
...	...	...	...	...	...	...	...
851259	2016-12-30	ZBH	103.309998	103.199997	102.849998	103.930000	973800.0
851260	2016-12-30	ZION	43.070000	43.040001	42.689999	43.310001	1938100.0
851261	2016-12-30	ZTS	53.639999	53.529999	53.270000	53.740002	1701200.0
851262	2016-12-30	AIV	44.730000	45.450001	44.410000	45.590000	1380900.0
851263	2016-12-30	FTV	54.200001	53.630001	53.389999	54.480000	705100.0

851264 rows × 7 columns

501 unique company tickers taken into account:



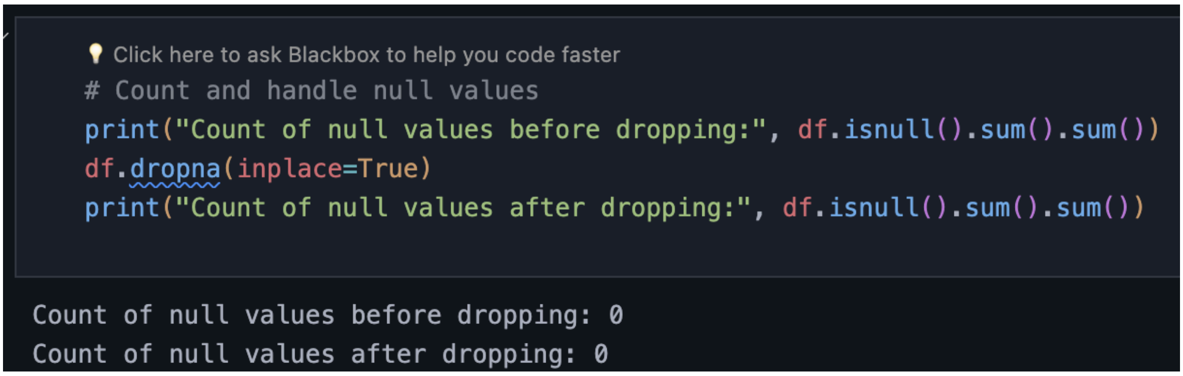
💡 Click here to ask Blackbox to help you code faster

```
df['symbol'].unique()
```

```
['WLTW', 'A', 'AAL', 'AAP', 'AAPL', ..., 'KHC', 'PYPL', 'HPE', 'CSRA', 'FTV']
Length: 501
Categories (501, object): ['A', 'AAL', 'AAP', 'AAPL', ..., 'YUM', 'ZBH', 'ZION', 'ZTS']
```

Normalization:

- `.isnull()`: This method is used to check for null values in a DataFrame. It returns a DataFrame of the same shape as `df`, where each element is True if the corresponding element in `df` is null, otherwise False.
- `.sum()`: This method is used to sum the values along a specified axis. When applied to the DataFrame returned by `.isnull()`, it calculates the total number of null values in each column.
- `.dropna()`: This method is used to remove rows or columns with null values from a DataFrame. The parameter `inplace=True` specifies that the changes should be made directly to the original DataFrame rather than returning a new DataFrame.

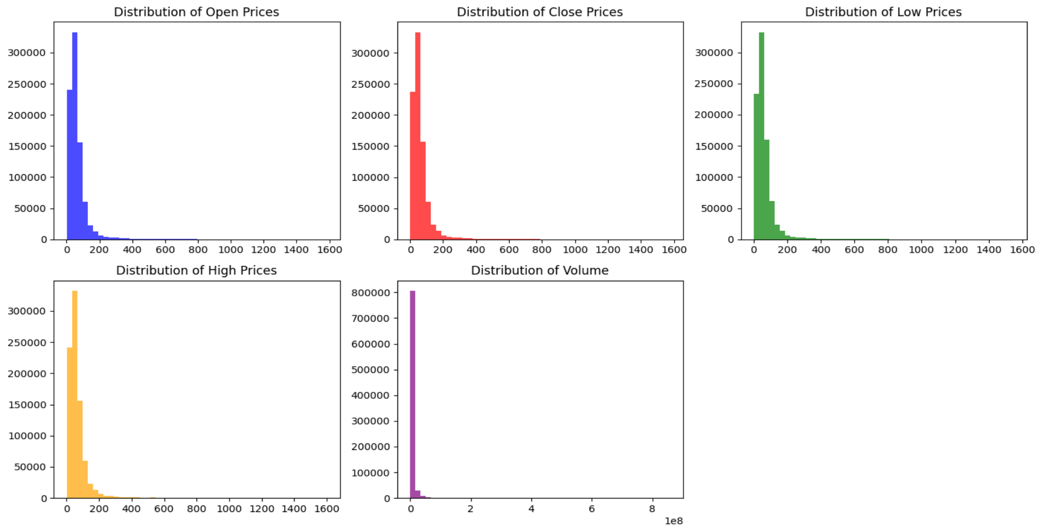


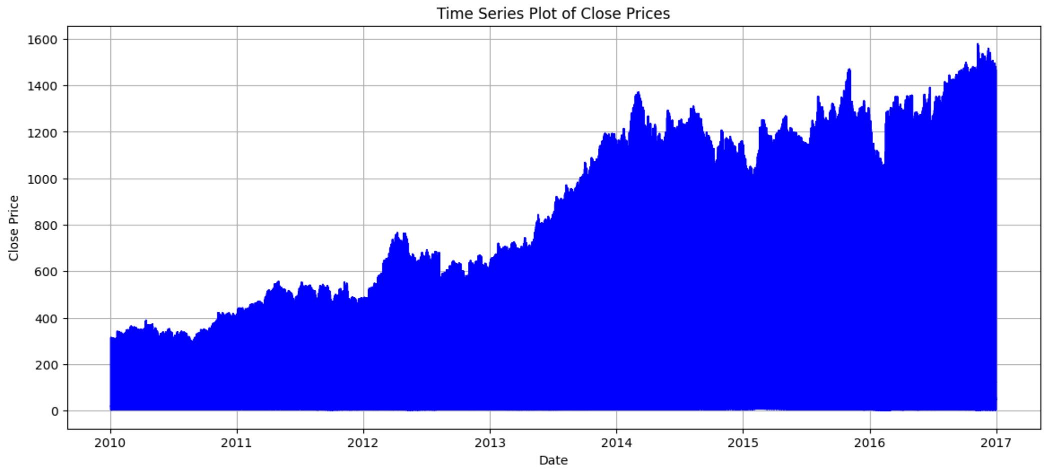
💡 Click here to ask Blackbox to help you code faster

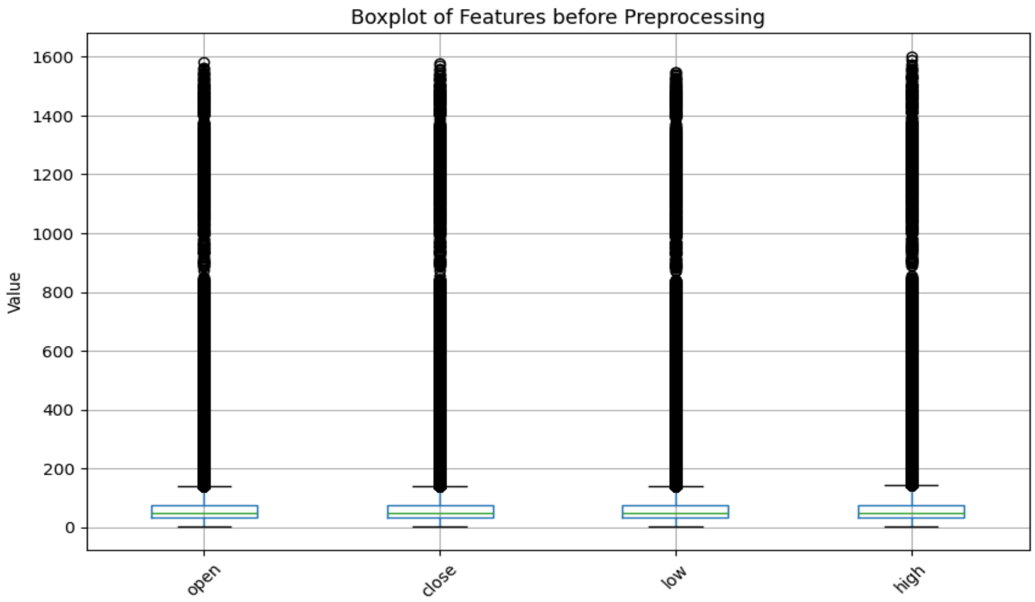
```
# Count and handle null values
print("Count of null values before dropping:", df.isnull().sum().sum())
df.dropna(inplace=True)
print("Count of null values after dropping:", df.isnull().sum().sum())
```

```
Count of null values before dropping: 0
Count of null values after dropping: 0
```

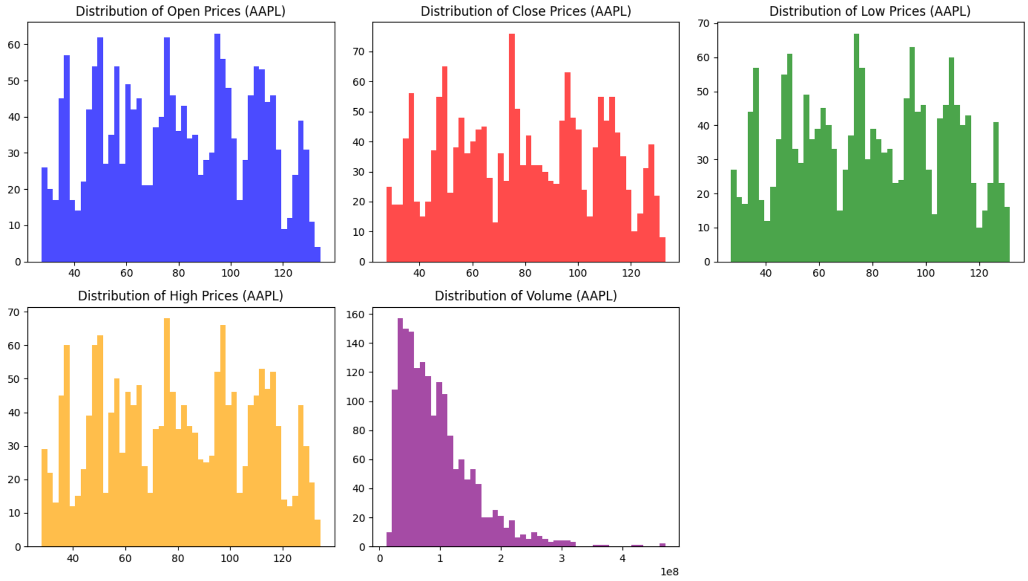
Parameter Distributions:(entire dataset)

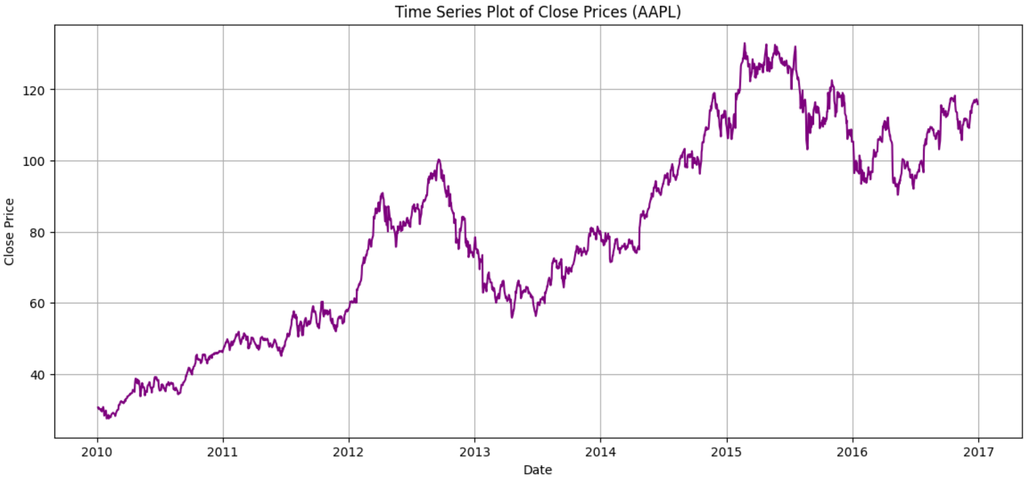


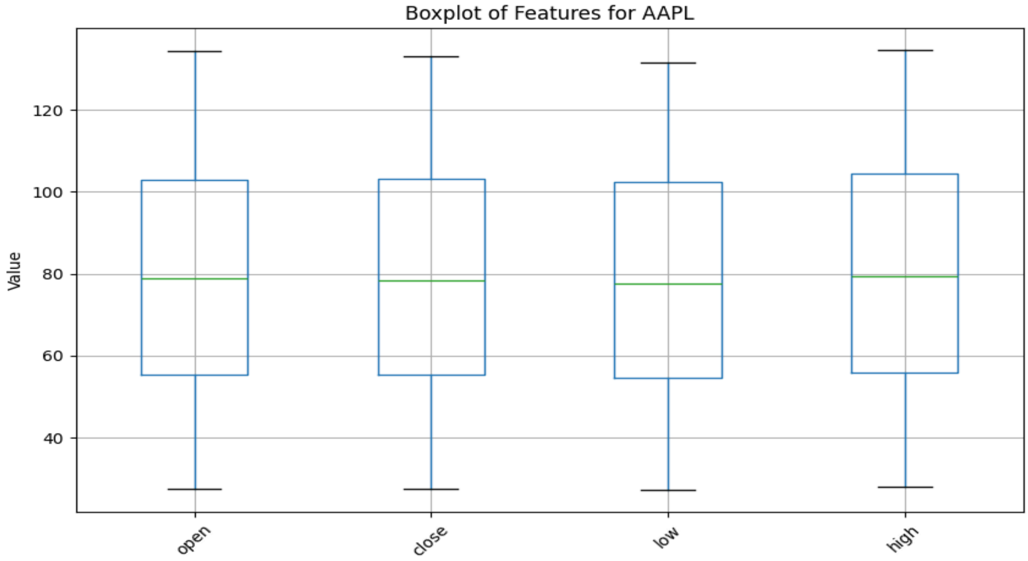




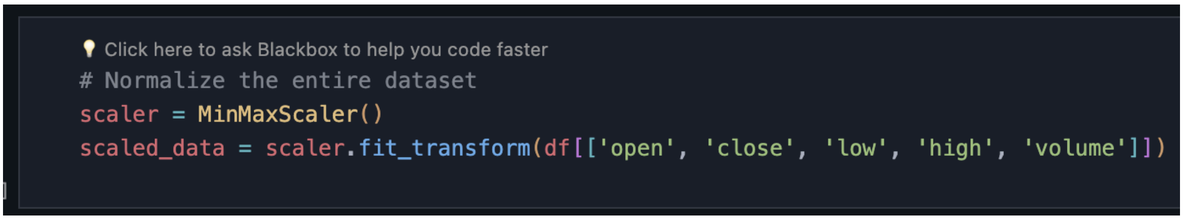
Parameter Distributions:(specific company: APPLE)







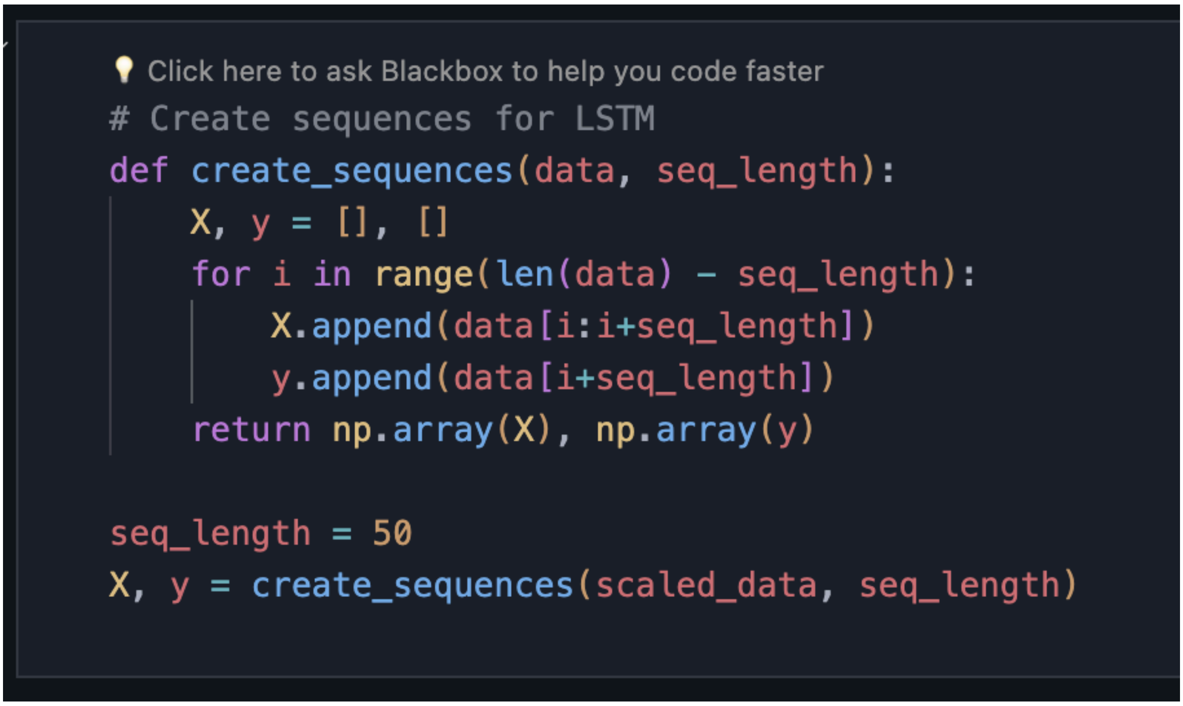
Normalizing the Dataset:



💡 Click here to ask Blackbox to help you code faster

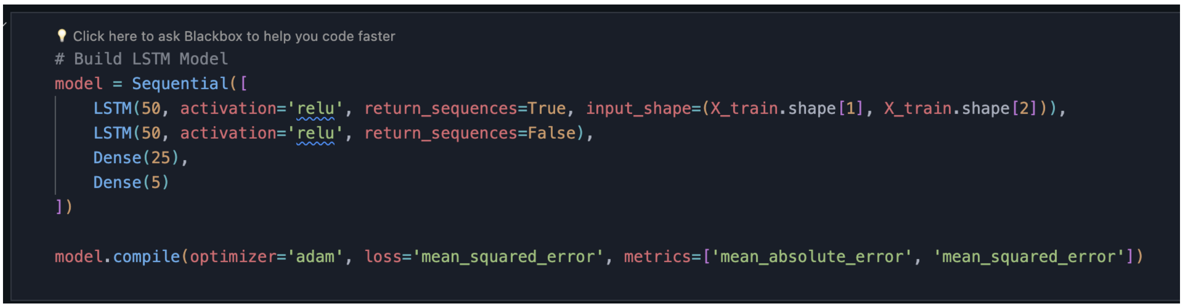
```
# Normalize the entire dataset
scaler = MinMaxScaler()
scaled_data = scaler.fit_transform(df[['open', 'close', 'low', 'high', 'volume']])
```

Creating LSTM sequence and architecture:



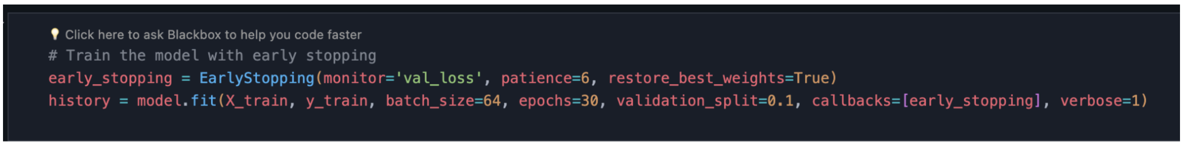
💡 Click here to ask Blackbox to help you code faster

```
# Create sequences for LSTM
def create_sequences(data, seq_length):
    X, y = [], []
    for i in range(len(data) - seq_length):
        X.append(data[i:i+seq_length])
        y.append(data[i+seq_length])
    return np.array(X), np.array(y)

seq_length = 50
X, y = create_sequences(scaled_data, seq_length)
```



💡 Click here to ask Blackbox to help you code faster

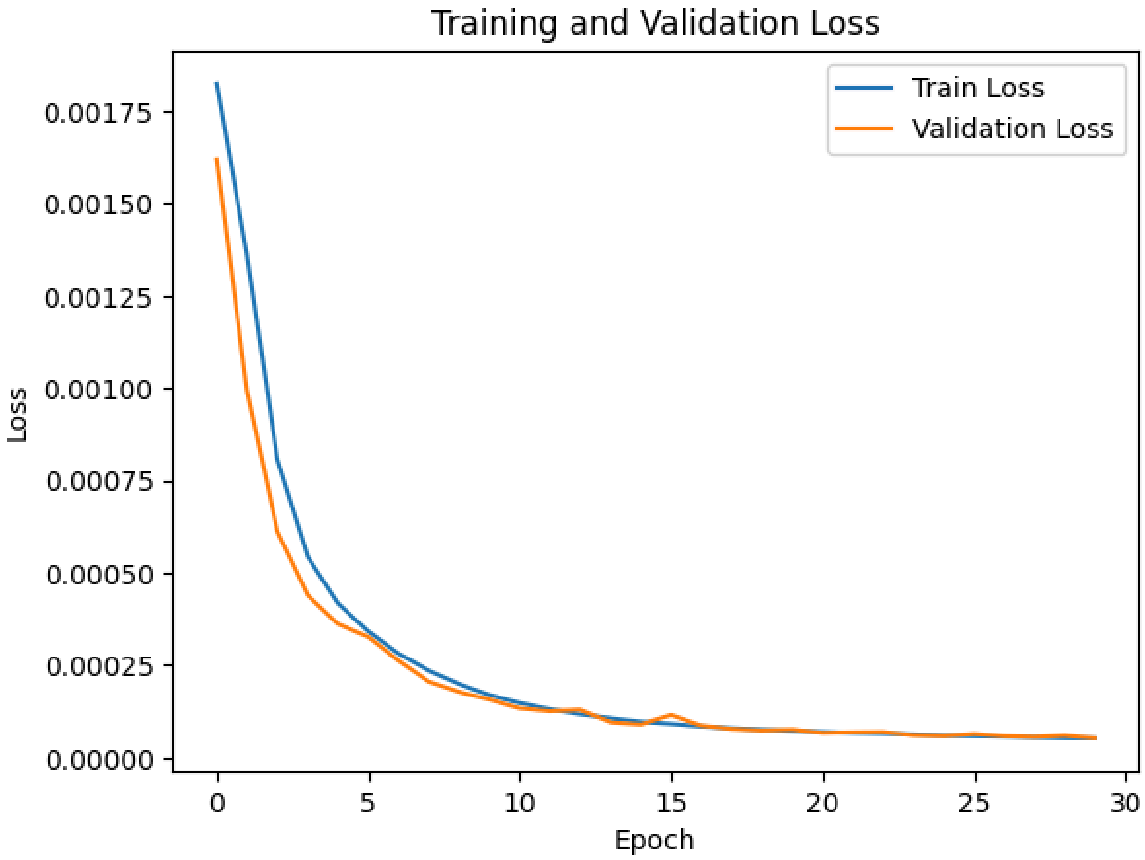
```
# Build LSTM Model
model = Sequential([
    LSTM(50, activation='relu', return_sequences=True, input_shape=(X_train.shape[1], X_train.shape[2])),
    LSTM(50, activation='relu', return_sequences=False),
    Dense(25),
    Dense(5)
])

model.compile(optimizer='adam', loss='mean_squared_error', metrics=['mean_absolute_error', 'mean_squared_error'])
```



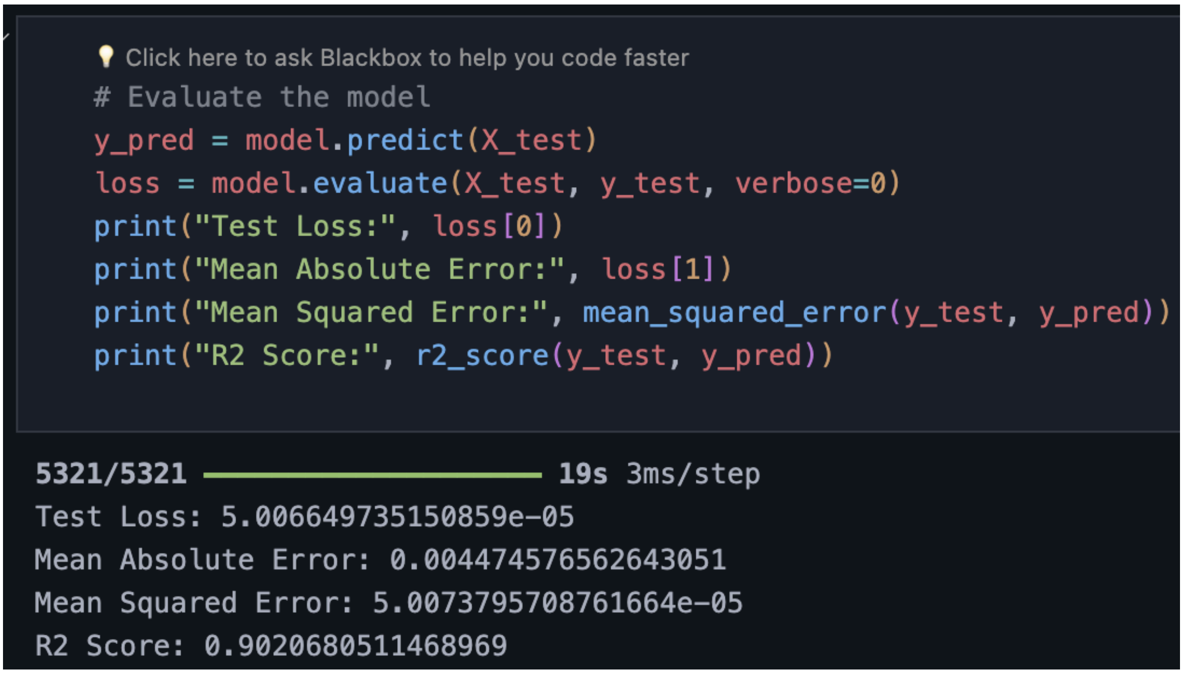
💡 Click here to ask Blackbox to help you code faster

```
# Train the model with early stopping
early_stopping = EarlyStopping(monitor='val_loss', patience=6, restore_best_weights=True)
history = model.fit(X_train, y_train, batch_size=64, epochs=30, validation_split=0.1, callbacks=[early_stopping], verbose=1)
```

Plots after model training:

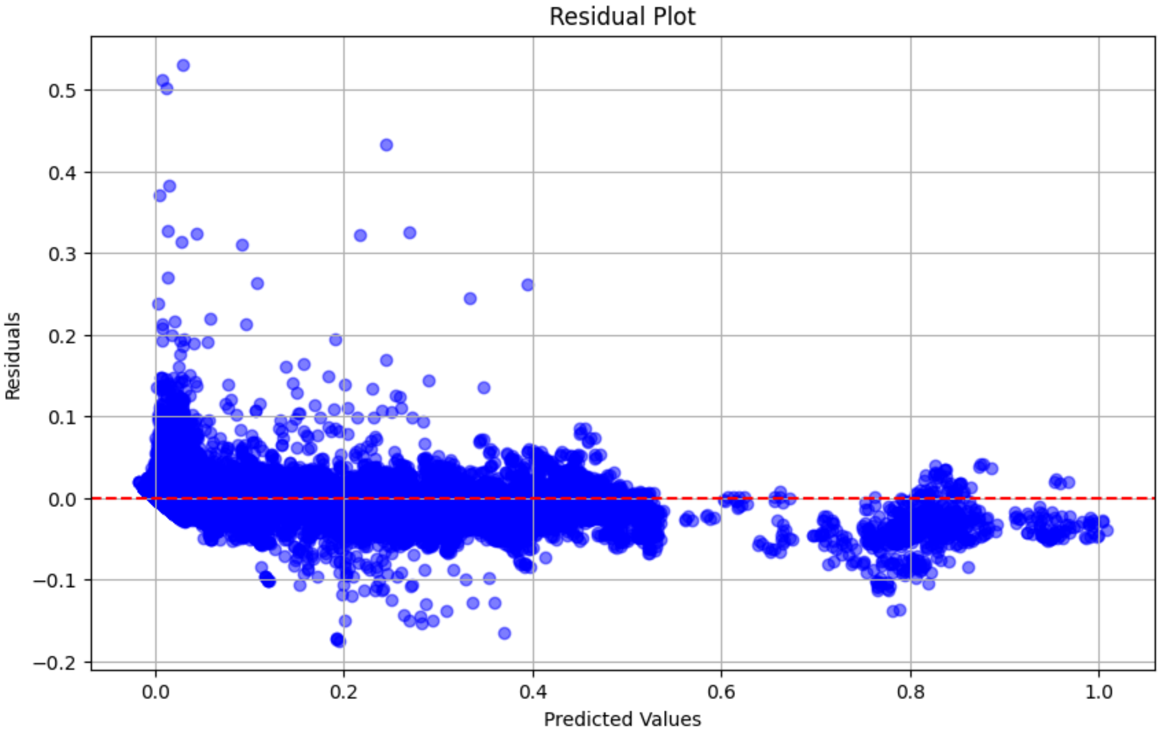


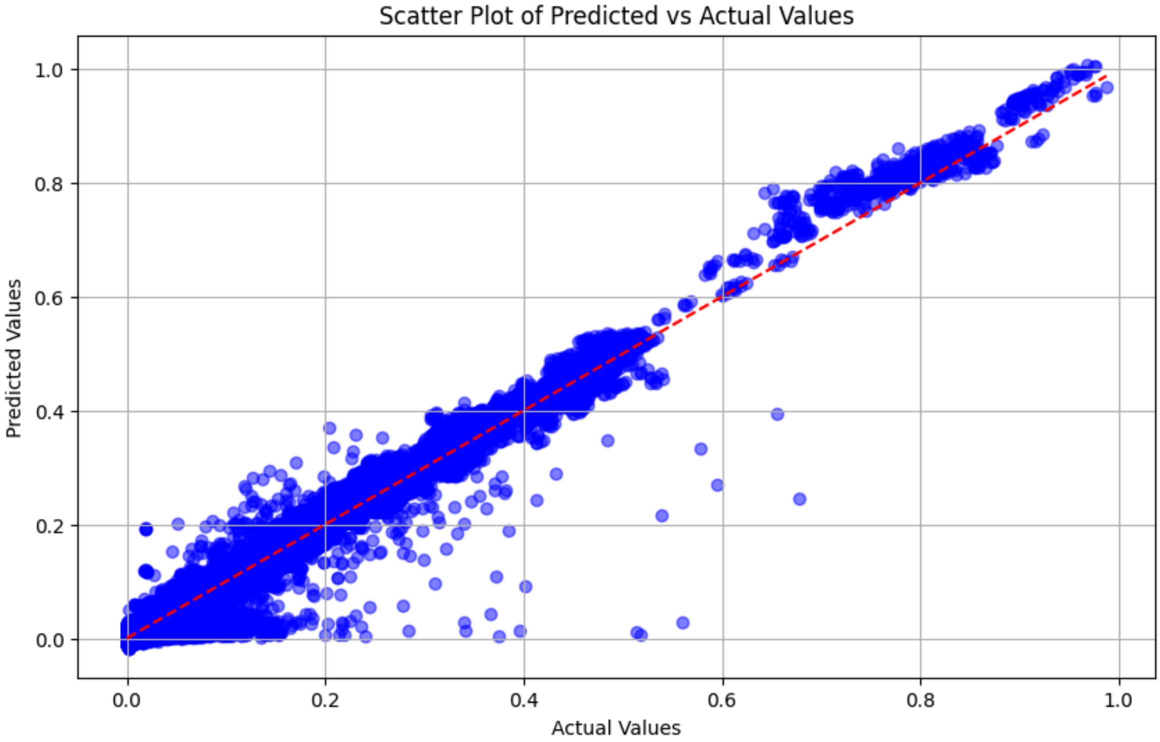
Evaluation of model performance:

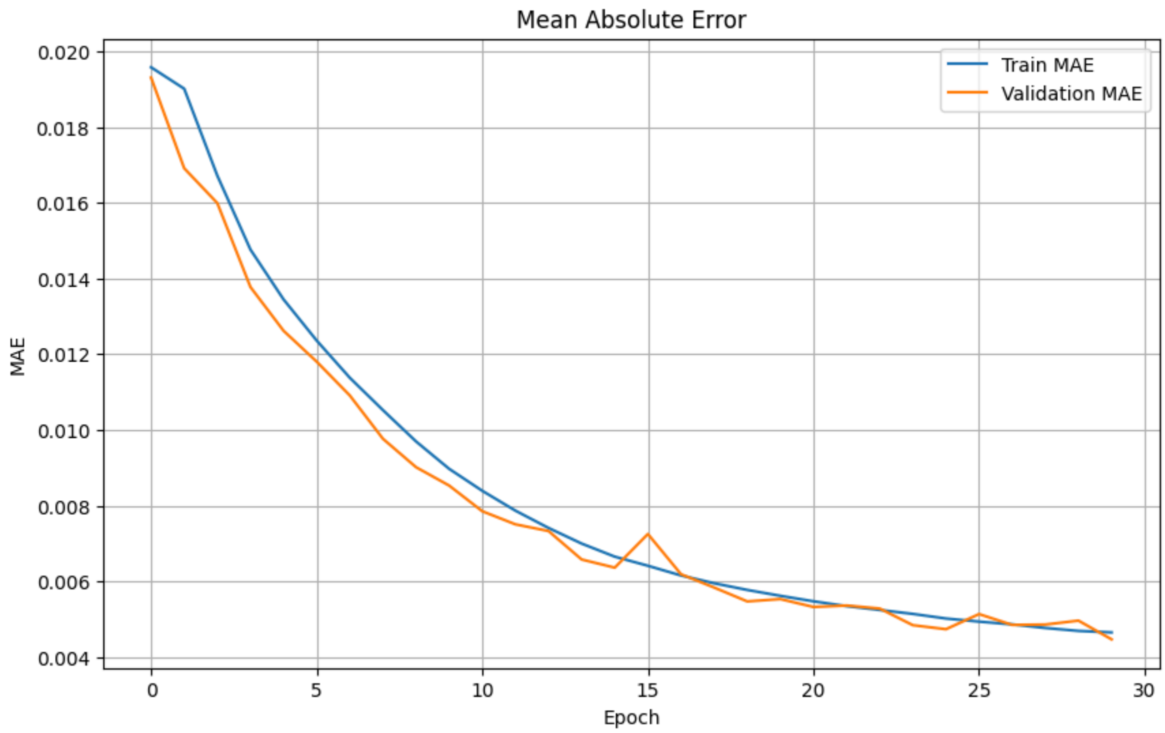


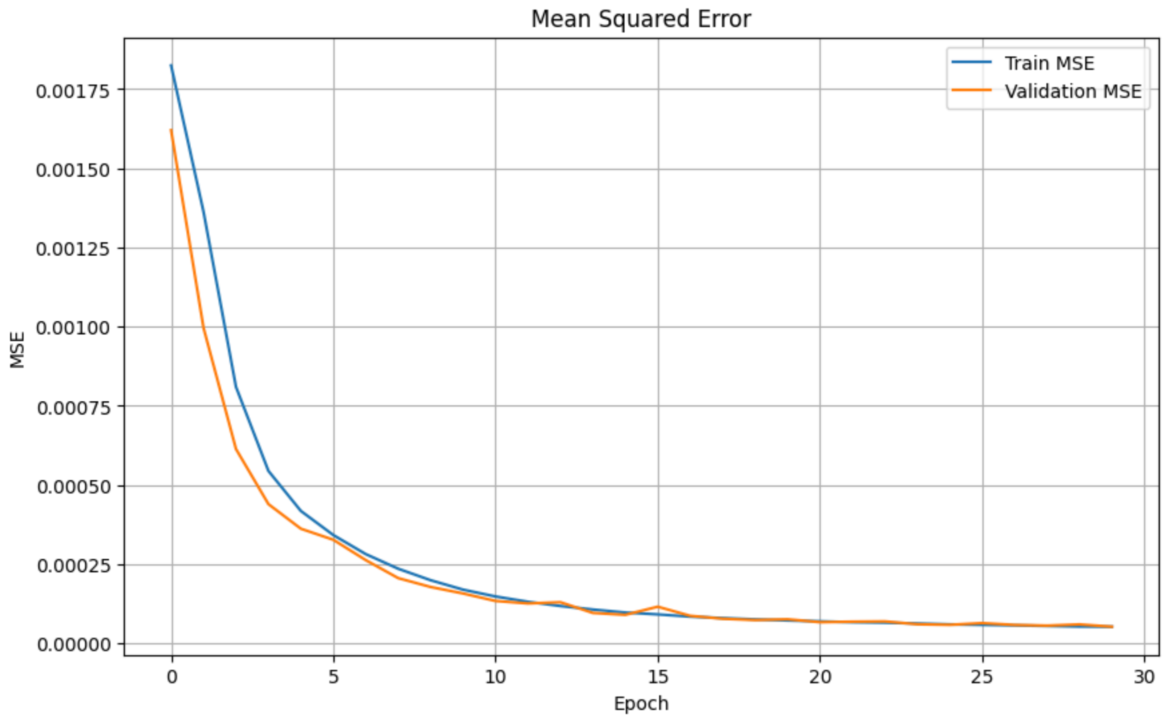
```
# Evaluate the model
y_pred = model.predict(X_test)
loss = model.evaluate(X_test, y_test, verbose=0)
print("Test Loss:", loss[0])
print("Mean Absolute Error:", loss[1])
print("Mean Squared Error:", mean_squared_error(y_test, y_pred))
print("R2 Score:", r2_score(y_test, y_pred))
```

```
5321/5321 ━━━━━━━━━━━━━━━━━━━━ 19s 3ms/step
Test Loss: 5.006649735150859e-05
Mean Absolute Error: 0.004474576562643051
Mean Squared Error: 5.0073795708761664e-05
R2 Score: 0.9020680511468969
```


Residual Plot
Residuals
0.5
0.4
0.3
0.2
0.1
0.0
−0.1
−0.2
0.0
0.2
0.4
0.6
0.8
1.0
Predicted Values


Scatter Plot of Predicted vs Actual Values
Predicted Values
1.0
0.8
0.6
0.4
0.2
0.0
0.0
0.2
0.4
0.6
0.8
1.0
Actual Values


Mean Absolute Error
Train MAE
Validation MAE
MAE
0.020
0.018
0.016
0.014
0.012
0.010
0.008
0.006
0.004
0
5
10
15
20
25
30
Epoch


Mean Squared Error
Train MSE
Validation MSE
MSE
0.00175
0.00150
0.00125
0.00100
0.00075
0.00050
0.00025
0.00000
0
5
10
15
20
25
30
Epoch

10. References

[1] Nandakumar, R., Uttamraj, K. R., Vishal, R., & Lokeswari, Y. V. (2018). Stock price prediction using long short term memory. International Research Journal of Engineering and Technology, 5(03).

[2] Soulas, Eleftherios, and Dennis Shasha. “Online machine learning algorithms for currency exchange prediction.” Computer Science Department in New York University, Tech. Rep 31 (2013).

[3] Murtaza Roondiwala, Harshal Patel, Shraddha Varma, “Predicting Stock Prices Using LSTM”, April 2017.

[4] Ryo Akita, Akira Yoshihara, Takashi Matsubara, Kuniaki Uehara, “Deep learning for stock prediction using numerical and textual information”, 2016.

[5] Graves, A., Mohamed, A.r., Hinton, G.: Speech recognition with deep recurrent neural networks.

[6] Pang, X., Zhou, Y., Wang, P., Lin, W., Chang, V.: An innovative neural network approach for stock market prediction. J. Supercomput. **76**(3), 2098–2118 (2020)