**Lab Assignment 10**

**Neural Network & Deep Learning**

**Stock Market Prediction using LSTM**

PART B

|  |  |
| --- | --- |
| Roll No: C0009 | Name: Samarth Borade |
| Class : B | Batch : EB1 |
| Date of Experiment: 01/03/24 | Date of Submission |
| Grade : |  |

**B.1 Software Code written by student:**

import numpy as np

import pandas as pd

from sklearn.preprocessing import MinMaxScaler

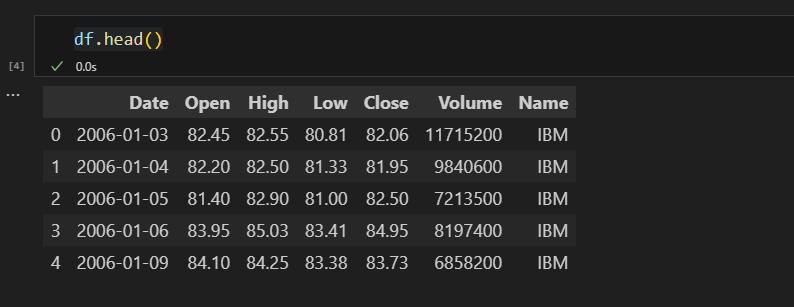
from keras.models import Sequential

from keras.layers import LSTM, Dense

import matplotlib.pyplot as plt

# Step 1: Load the dataset

df = pd.read\_csv('IBM.csv')



# Step 2: Select the appropriate feature for creating the model from the training data

data = df['Close'].values.reshape(-1, 1)

# Step 3: Normalize the features and convert it into timestamps of 60

scaler = MinMaxScaler(feature\_range=(0, 1))

data = scaler.fit\_transform(data)

def create\_dataset(data, timestamp):

    X, y = [], []

    for i in range(len(data) - timestamp):

        X.append(data[i:(i + timestamp), 0])

        y.append(data[i + timestamp, 0])

    return np.array(X), np.array(y)

# Step 4: Reshape the data for applying to the LSTM model

timestamp = 60

X, y = create\_dataset(data, timestamp)

# Reshape data into (samples, timestamp, features)

X = np.reshape(X, (X.shape[0], X.shape[1], 1))

# Step 5: Create a sequential LSTM model using Keras

model = Sequential()

model.add(LSTM(units=50, return\_sequences=True, input\_shape=(X.shape[1], 1)))

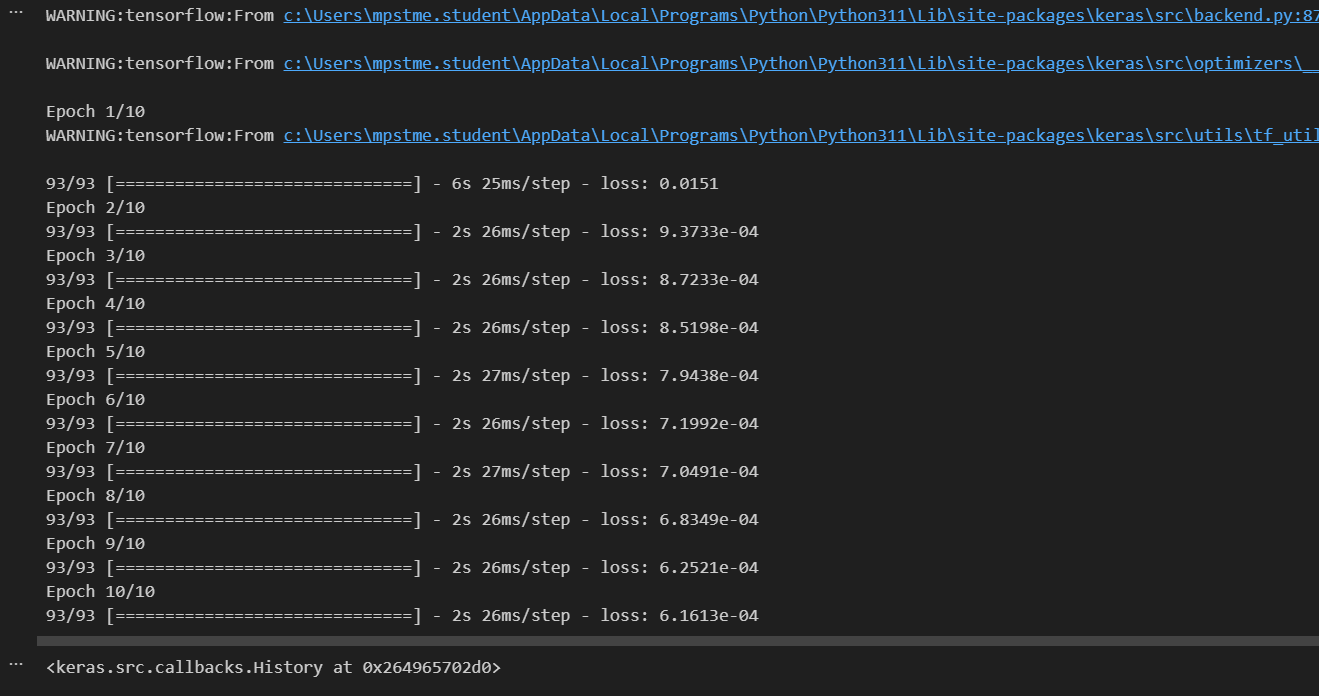
model.add(LSTM(units=50))

model.add(Dense(units=1))

# Step 6: Compile the model and train it using the training data

model.compile(optimizer='adam', loss='mean\_squared\_error')

model.fit(X, y, epochs=10, batch\_size=32)



Output:

# Step 7: Predict using the test data

test\_data = df['Close'].values.reshape(-1, 1)

scaled\_test\_data = scaler.transform(test\_data)

X\_test, y\_test = create\_dataset(scaled\_test\_data, timestamp)

X\_test = np.reshape(X\_test, (X\_test.shape[0], X\_test.shape[1], 1))

predicted\_stock\_price = model.predict(X\_test)

predicted\_stock\_price = scaler.inverse\_transform(predicted\_stock\_price)

# Plotting the results

plt.plot(test\_data[timestamp:], color='blue', label='Actual Stock Price')

plt.plot(predicted\_stock\_price, color='red', label='Predicted Stock Price')

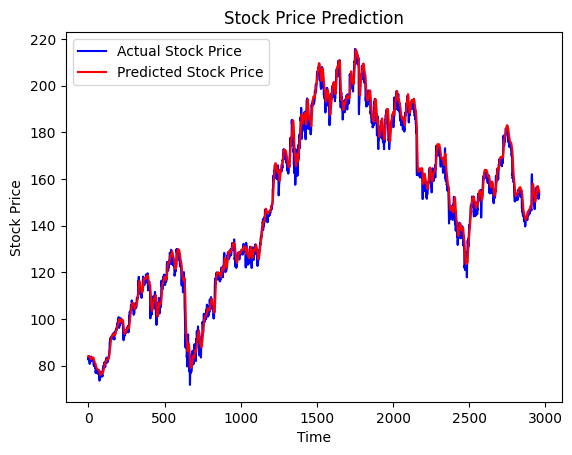
plt.title('Stock Price Prediction')

plt.xlabel('Time')

plt.ylabel('Stock Price')

plt.legend()

plt.show()



**B.3 Observations and learning:**

**Prediction for next year:**

# Step 7: Predict for the next year

future\_timestamps = 252  # Assuming 252 trading days in a year

future\_data = data[-timestamp:]  # Taking the last 'timestamp' data points

predicted\_prices = []

for i in range(future\_timestamps):

    X\_future = np.reshape(future\_data, (1, timestamp, 1))

    predicted\_price = model.predict(X\_future)

    predicted\_prices.append(predicted\_price)

    future\_data = np.append(future\_data[1:], predicted\_price)  # Shift the data window by 1

# Inverse transform the predicted prices to get the actual stock prices

predicted\_prices = scaler.inverse\_transform(np.array(predicted\_prices).reshape(-1, 1))

# Plotting the results

plt.plot(predicted\_prices, color='green', label='Predicted Stock Price for Next Year')

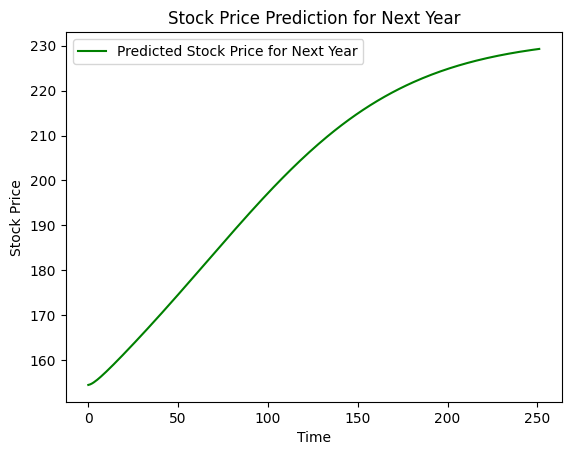
plt.title('Stock Price Prediction for Next Year')

plt.xlabel('Time')

plt.ylabel('Stock Price')

plt.legend()

plt.show()

****

After training an LSTM model on historical IBM stock prices, I applied it to predict the stock prices for the next year. Despite the model's capability to capture temporal patterns in the data, the predicted stock prices exhibit a smooth upward trend, lacking the volatility and fluctuations characteristic of real stock market behavior.

**B.4 Conclusion:**

* While the LSTM model demonstrates an understanding of the underlying patterns in the historical data, its predictions for future stock prices appear overly simplistic, failing to account for the complexities and uncertainties inherent in financial markets.
* The model's reliance on past data might lead to a bias towards the prevailing trends, overlooking sudden market shifts or external influences that can significantly impact stock prices.
* Therefore, while LSTM models can offer valuable insights into potential future trends, they should be used cautiously in real-world investment decision-making, complemented by thorough analysis and consideration of various factors affecting stock market dynamics.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*