**STOCK PRICE PREDICTION (Phase 4)**

**Madras Institute of Technology**

2021506088 – Santhi Priya D N

2021506096 - Shiek Sajnathul Faizana

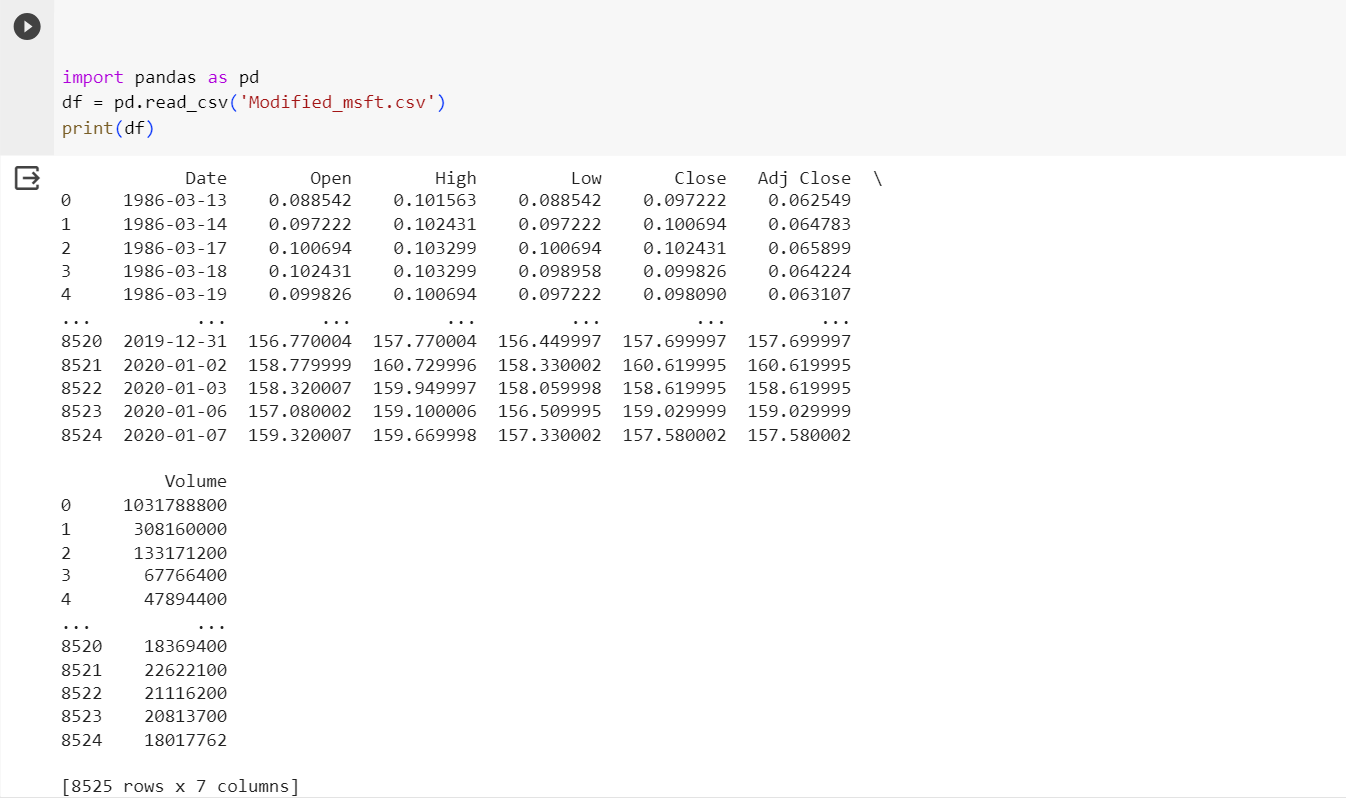
2021506103 – Sowmiya J

2021506323 - Revanth P

**INTRODUCTION:**

In Phase 4 of our stock price prediction project, we continue to advance our efforts to create a robust and accurate predictive model. Building upon the foundation laid in previous phases, this stage is dedicated to key activities that are instrumental in developing a model that can make well-informed predictions, optimize investment strategies, and adapt to dynamic market conditions.

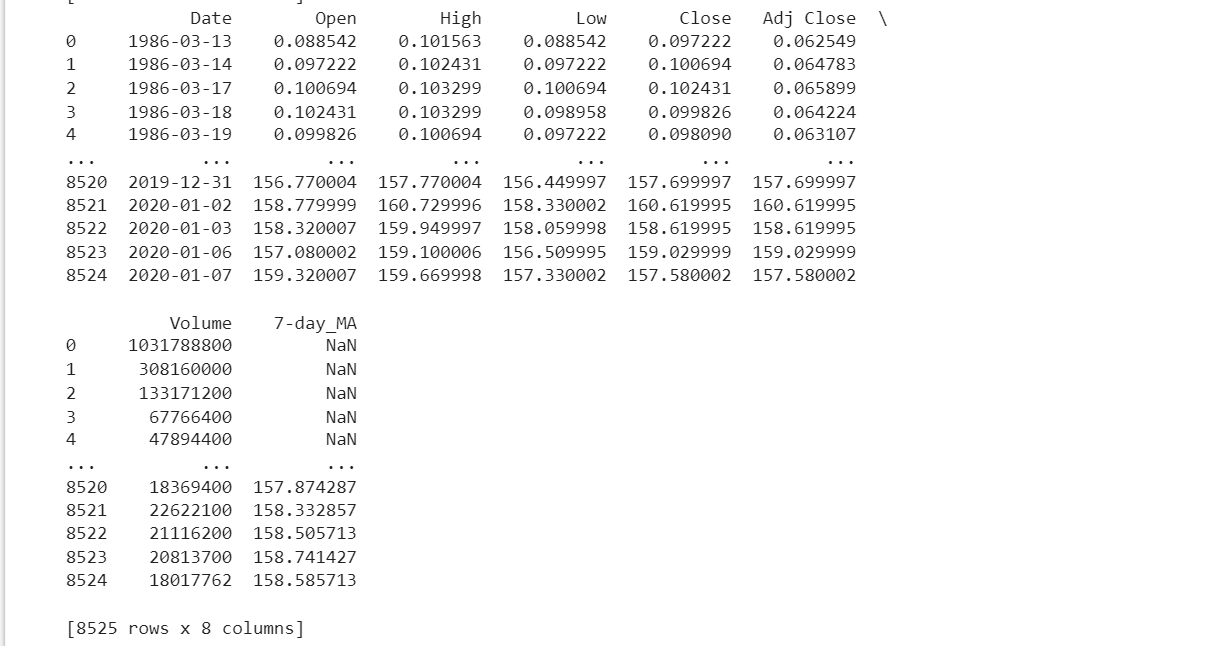
**Reading the dataset:**



**FEATURE ENGINEERING:**

Feature engineering is the process of creating new features or transforming existing ones from the raw data to improve the predictive power of the model. These features may capture patterns, trends, and relationships in the data that are not evident in the original features. In the context of our project, feature engineering involves calculating moving averages, technical indicators, and other relevant variables to enhance the model's ability to understand and forecast stock prices.

**7-day moving average:**



**LSTM MODEL:**

Long Short-Term Memory (LSTM) is a type of recurrent neural network (RNN) architecture that is designed to effectively capture and model long-term dependencies and sequential patterns in data. Here's a definition of an LSTM model:

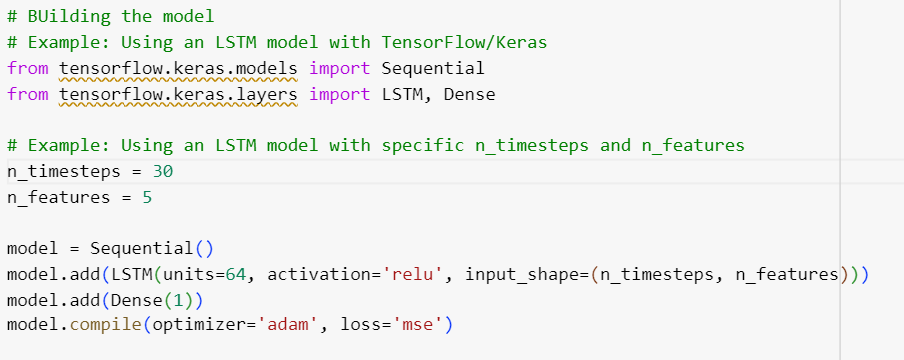
An LSTM model is a recurrent neural network architecture that is particularly well-suited for sequential data, such as time series, natural language text, and speech. It is designed to overcome the vanishing gradient problem in traditional RNNs and can capture long-range dependencies in data. An LSTM model consists of specialized units called "memory cells" that can store and retrieve information over extended sequences, making it highly effective for tasks like time series forecasting, natural language processing, and speech recognition.

Key features of LSTM models include the ability to remember information over long time horizons, the presence of gates to control the flow of information, and the capacity to handle sequences of varying lengths. These qualities make LSTM models a popular choice for a wide range of applications in machine learning and deep learning.

In the context of stock price prediction, LSTM models can be used to capture the temporal patterns and dependencies in historical market data, allowing for the creation of predictive models that consider the intricate nature of financial time series.

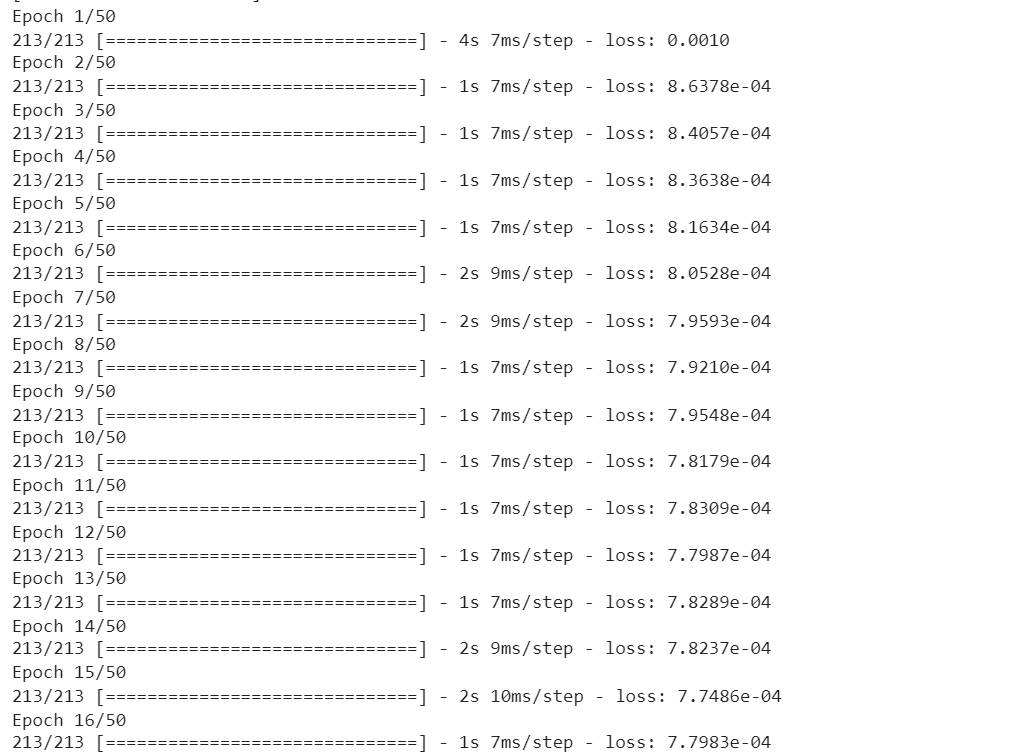
**MODEL BUILDING:**

Model training is the phase where we feed our prepared data into a chosen predictive model. The model learns to identify patterns and relationships in the training data, allowing it to make predictions on unseen data. In our case, we continue to explore and implement models, particularly focusing on LSTM, to capture the temporal dependencies in stock price movements.

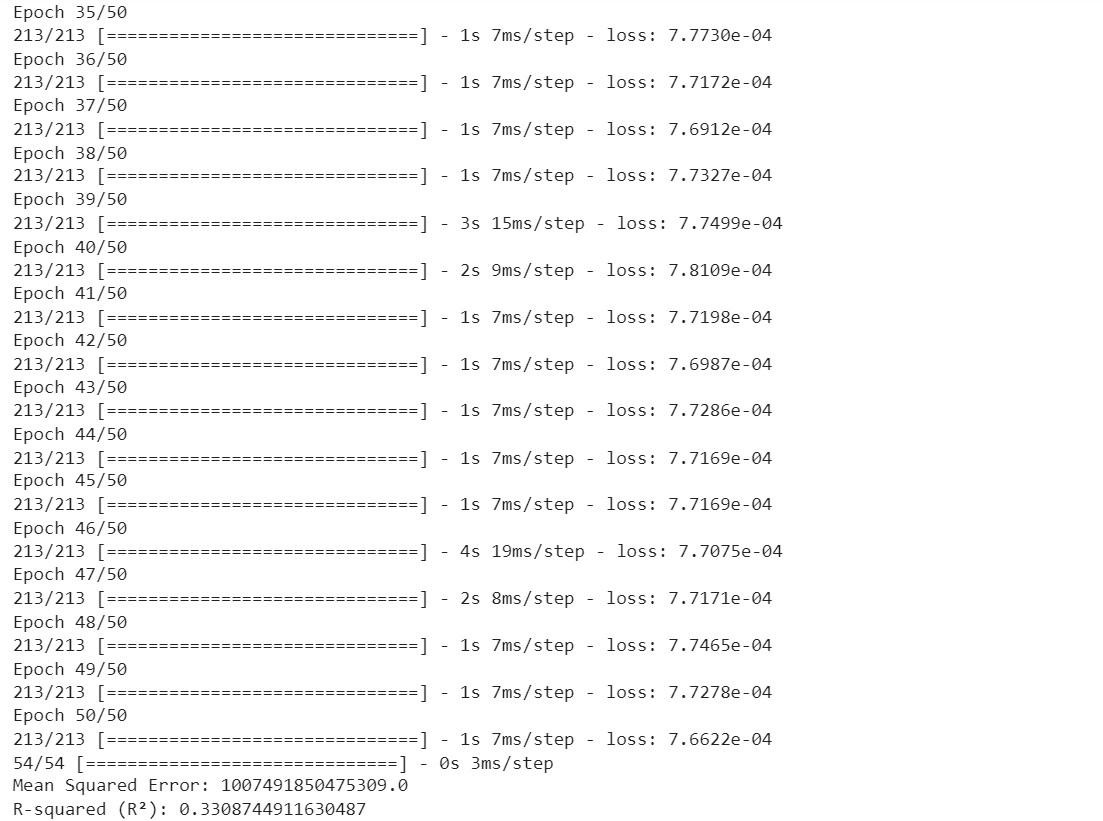


**MODEL TRAINING AND EVALUATION:**

Evaluation is the critical process of assessing the performance of our model. To measure its accuracy, we employ various evaluation metrics such as Mean Absolute Error (MAE), Root Mean Squared Error (RMSE), and others specifically designed for time series forecasting. By evaluating the model's performance on historical data, we gauge its predictive capabilities and its potential to assist investors in making informed decisions.



****

****

**SOURCE CODE:**

*import pandas as pd*

*df = pd.read\_csv('Modified\_msft.csv')*

*print(df)*

*#feature engineering*

*#--adding lag values*

*df['Volume\_lag1']=df['Volume'].shift(1)*

*df['Volume\_lag2']=df['Volume'].shift(2)*

*#-- 7 day moving average*

*df['7-day\_MA'] = df['Volume'].rolling(window=7).mean()*

*print(df)*

*# Import necessary libraries*

*import numpy as np*

*import pandas as pd*

*import tensorflow as tf*

*from tensorflow import keras*

*from sklearn.model\_selection import train\_test\_split*

*from sklearn.preprocessing import MinMaxScaler*

*from sklearn.metrics import mean\_squared\_error, r2\_score*

*# Load your time series data or sequence data*

*# Replace 'data' with your time series data*

*# Example: data = load\_your\_data()*

*# Data preprocessing*

*scaler = MinMaxScaler()*

*scaled\_data = scaler.fit\_transform(df[['Volume']])*

*# Define the sequence length and split the data into sequences*

*sequence\_length = 10 # Adjust this based on your problem*

*X, y = [], []*

*for i in range(len(scaled\_data) - sequence\_length):*

*X.append(scaled\_data[i:i+sequence\_length])*

*y.append(scaled\_data[i+sequence\_length])*

*X, y = np.array(X), np.array(y)*

*# Split the data into training and testing sets*

*X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2, random\_state=42)*

*# Reshape the input data to match the expected input shape*

*num\_features = X\_train.shape[2] # Extract the number of features from the reshaped data*

*# Define and compile the LSTM model*

*model = keras.Sequential()*

*model.add(keras.layers.LSTM(units=50, activation='relu', input\_shape=(sequence\_length, num\_features)))*

*model.add(keras.layers.Dense(1)) # Adjust the output layer for your problem*

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

*# Train the model*

*model.fit(X\_train, y\_train, epochs=50, batch\_size=32)*

*# Make predictions on the test set*

*y\_pred = model.predict(X\_test)*

*# Inverse transform predictions to the original scale if needed*

*y\_pred = scaler.inverse\_transform(y\_pred)*

*y\_test = scaler.inverse\_transform(y\_test)*

*# Evaluate the model using appropriate metrics*

*mse = mean\_squared\_error(y\_test, y\_pred)*

*r2 = r2\_score(y\_test, y\_pred)*

*# Print or visualize the model's performance metrics*

*print(f"Mean Squared Error: {mse}")*

*print(f"R-squared (R²): {r2}")*

**CONCLUSION:**

With feature engineering, we enrich our dataset with additional informative variables, enhancing the model's capacity to understand the complexities of stock market behaviour. Model training is a pivotal stage where our selected LSTM model learns from the historical data, laying the foundation for predictions. Lastly, the evaluation process provides valuable insights into our model's performance, enabling us to refine its architecture and optimize its predictive accuracy.