LABORATORY REPORT

Application Development Lab
(CS33002)

B.Tech Program in ECSc

Submitted By

Name:- Kartik Sharma

Roll No: 2230176

Kalinga Institute of Industrial Technology
(Deemed to be University)
Bhubaneswar, India

Spring 2024-2025

Table of Content

Experiment Number	3
Experiment Title	Regression Analysis for Stock Prediction
Date of Experiment	21/01/2025
Date of Submission	27/01/2025

1. Objective:- To perform stock price prediction using Linear Regression and LSTM models.

2. Procedure:-

1. Collect historical stock price data.
2. Preprocess the data for analysis (missing data, scaling, splitting into train/test).
3. Implement Linear Regression to predict future stock prices.
4. Design and train an LSTM model for time-series prediction.
5. Compare the accuracy of both models.
6. Create a Flask backend for model predictions.
7. Build a frontend to visualize predictions using charts and graphs.

Code:-

Frontend code (html):

```
<!DOCTYPE html>
<html lang="en">
<head>
   <meta charset="UTF-8">
   <meta name="viewport" content="width=device-width, initial-scale=1.0">
   <title>Regression Analysis for Stock Prediction</title>
   <style>
      body {
         background-color: red;
         color: lightyellow;
         font-family: Arial, sans-serif;
         margin: 0;
         padding: 0;
         display: flex;
         flex-direction: column;
         align-items: flex-start;
         justify-content: center;
         min-height: 100vh;
         padding-left: 2rem;
         text-align: left;
      }

      h1 {
         font-size: 2.5rem;
         margin-bottom: 2rem;
         color: lightyellow;
      }

      a {
         display: inline-block;
         background-color: lightyellow;
         color: red;
```

```
        text-decoration: none;
        padding: 0.75rem 1.5rem;
        font-size: 1rem;
        border-radius: 5px;
        transition: background-color 0.3s, color 0.3s;
      }

      a:hover {
        background-color: darkred;
        color: lightyellow;
      }

      footer {
        margin-top: 2rem;
        font-size: 0.875rem;
        color: lightyellow;
      }
    </style>
</head>
<body>
    <h1>Regression Analysis for Stock Prediction</h1>
    <a href="https://38b641063eaf6a4715.gradio.live" target="_blank">Go to Stock Prediction Tool</a>

</body>
</html>
```

Backend Code (Python) :

```
pip install gradio yfinance pandas numpy scikit-learn tensorflow matplotlib

!python stock_prediction_gradio.py

import yfinance as yf
import pandas as pd
import numpy as np
from sklearn.preprocessing import MinMaxScaler
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error
```

```
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import LSTM, Dense
import matplotlib.pyplot as plt
import gradio as gr

def get_stock_data(ticker, start_date, end_date):
    data = yf.download(ticker, start=start_date, end=end_date)
    data.reset_index(inplace=True)
    return data
```

```
def preprocess_data(data):
    scaler = MinMaxScaler(feature_range=(0, 1))
    scaled_data = scaler.fit_transform(data[['Close']])
    return scaled_data, scaler
```

```
def create_features(data, lag=5):
    X, y = [], []
    for i in range(lag, len(data)):
        X.append(data[i-lag:i, 0])
        y.append(data[i, 0])
    return np.array(X), np.array(y)
```

```
def train_linear_regression(X_train, y_train):
    model = LinearRegression()
    model.fit(X_train, y_train)
    return model
```

```
def train_lstm_model(X_train, y_train):
    model = Sequential()
    model.add(LSTM(50, return_sequences=True, input_shape=(X_train.shape[1], 1)))
    model.add(LSTM(50, return_sequences=False))
    model.add(Dense(25))
    model.add(Dense(1))
    model.compile(optimizer='adam', loss='mean_squared_error')
    model.fit(X_train, y_train, batch_size=32, epochs=5, verbose=0)
    return model
```

```
def predict_and_visualize(ticker, start_date, end_date):

    data = get_stock_data(ticker, start_date, end_date)
    if data.empty:
        return "No data found for the given ticker and date range.", None
```

```
    scaled_data, scaler = preprocess_data(data)
    train_size = int(len(scaled_data) * 0.8)
    train_data = scaled_data[:train_size]
    test_data = scaled_data[train_size:]


    lag = 5
    X_train, y_train = create_features(train_data, lag)
    X_test, y_test = create_features(test_data, lag)
```

```
      lr_model = train_linear_regression(X_train, y_train)
    lr_predictions = lr_model.predict(X_test)
    lr_predictions_rescaled = scaler.inverse_transform(lr_predictions.reshape(-1, 1))
```

```

    X_train_lstm = X_train.reshape(X_train.shape[0], X_train.shape[1], 1)
    X_test_lstm = X_test.reshape(X_test.shape[0], X_test.shape[1], 1)
    lstm_model = train_lstm_model(X_train_lstm, y_train)
    lstm_predictions = lstm_model.predict(X_test_lstm)
    lstm_predictions_rescaled = scaler.inverse_transform(lstm_predictions)
```

```
    y_test_rescaled = scaler.inverse_transform(y_test.reshape(-1, 1))
```

```

    plt.figure(figsize=(12, 6))
    plt.plot(data.index[len(data) - len(y_test_rescaled):], y_test_rescaled, label="Actual Prices",
color="blue")
```

```
    plt.plot(data.index[len(data) - len(y_test_rescaled):], lr_predictions_rescaled, label="Linear
Regression Predictions", color="green")
    plt.plot(data.index[len(data) - len(y_test_rescaled):], lstm_predictions_rescaled, label="LSTM
Predictions", color="red")
    plt.title(f"Stock Price Prediction for {ticker}")
    plt.xlabel("Date")
    plt.ylabel("Price")
    plt.legend()
    plt.grid()
```

```
    plot_file =
    "prediction_plot.png"
   plt.savefig(plot_file)
   plt.close()
```

```
    lr_mse = mean_squared_error(y_test_rescaled, lr_predictions_rescaled)
    lstm_mse = mean_squared_error(y_test_rescaled, lstm_predictions_rescaled)
```

```
    return (f"Linear Regression MSE: {lr_mse:.4f}\nLSTM MSE: {lstm_mse:.4f}", plot_file)
```

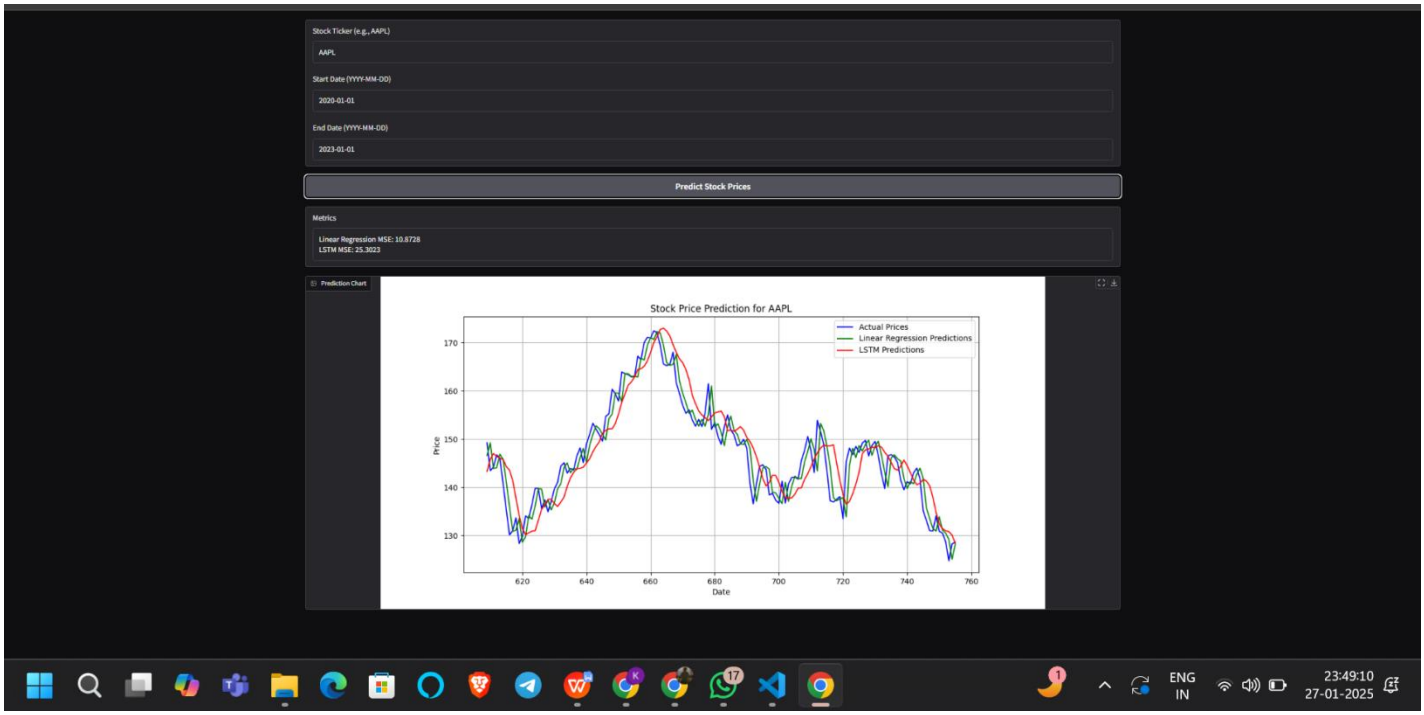
```
 def gradio_interface():
   with gr.Blocks() as demo:

       ticker_input = gr.Textbox(label="Stock Ticker (e.g., AAPL)", value="AAPL")
       start_date_input = gr.Textbox(label="Start Date (YYYY-MM-DD)", value="2020-01-01")
       end_date_input = gr.Textbox(label="End Date (YYYY-MM-DD)", value="2023-01-01")


       predict_button = gr.Button("Predict Stock Prices")

              output_text = gr.Textbox(label="Metrics", interactive=False)
       output_image = gr.Image(label="Prediction Chart", interactive=False)


       predict_button.click(
           fn=predict_and_visualize,
           inputs=[ticker_input, start_date_input, end_date_input],
           outputs=[output_text, output_image]
       )

    demo.launch()
```

```
if __name__ == "__main__":
    gradio_interface()
```

3. Results/Output:- Entire Screen Shot including Date & Time

Regression Analysis for Stock Prediction

Go to Stock Prediction Tool



4. Remarks:-

Kartik Sharma

(Name of the Student)

Signature of the Lab Coordinator

(Name of the Coordinator)