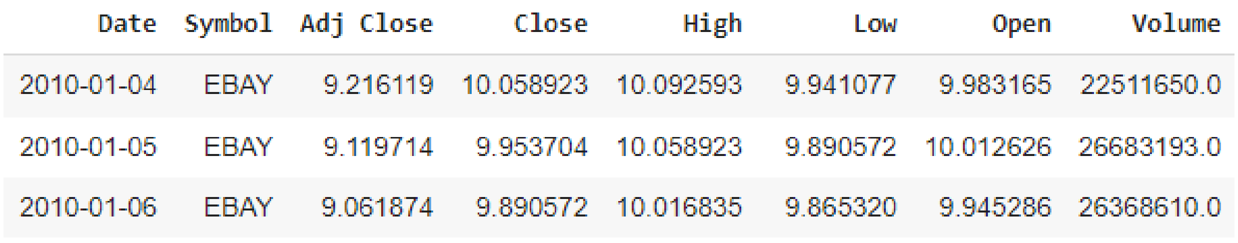
DATA PREPROCESSING:

- DATA:

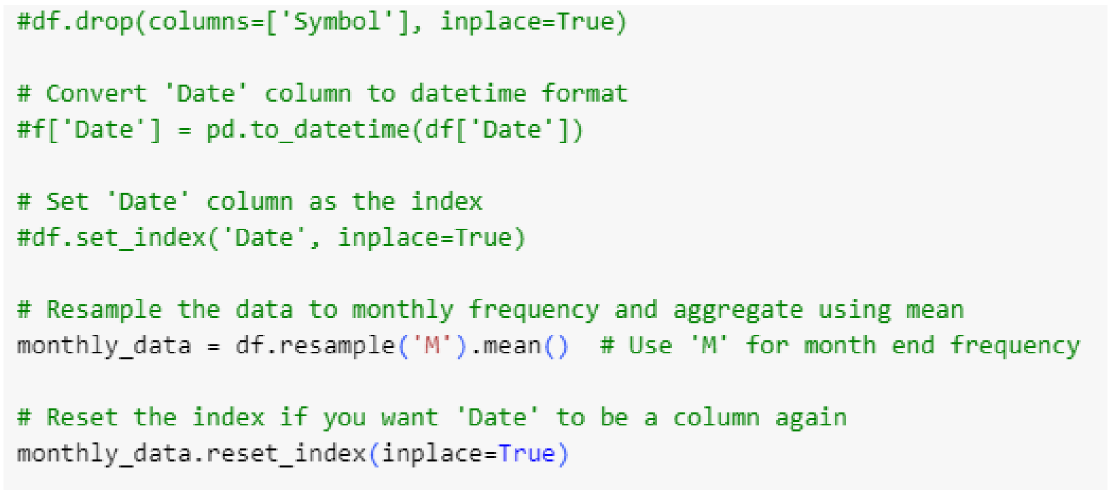
  Data contains information about Stocks of EBAY COMPANY. Data set contains all the information regarding stocks. Such as, Close, Adj close, Date, Volume, High, Low etc



Date	Symbol	Adj Close	Close	High	Low	Open	Volume
2010-01-04	EBAY	9.216119	10.058923	10.092593	9.941077	9.983165	22511650.0
2010-01-05	EBAY	9.119714	9.953704	10.058923	9.890572	10.012626	26683193.0
2010-01-06	EBAY	9.061874	9.890572	10.016835	9.865320	9.945286	26368610.0

- Preprocessing:

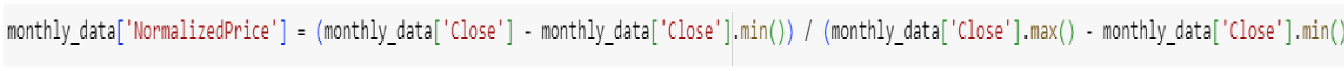
  Data set is based on daily stock data. As per requirement, Data is converted into monthly basis.



```
#df.drop(columns=['Symbol'], inplace=True)

# Convert 'Date' column to datetime format
#f['Date'] = pd.to_datetime(df['Date'])

# Set 'Date' column as the index
#df.set_index('Date', inplace=True)

# Resample the data to monthly frequency and aggregate using mean
monthly_data = df.resample('M').mean()  # Use 'M' for month end frequency

# Reset the index if you want 'Date' to be a column again
monthly_data.reset_index(inplace=True)
```

- Normalizing Data:

  As the P-value of the data is greater than 0.05 so I had to normalize that.



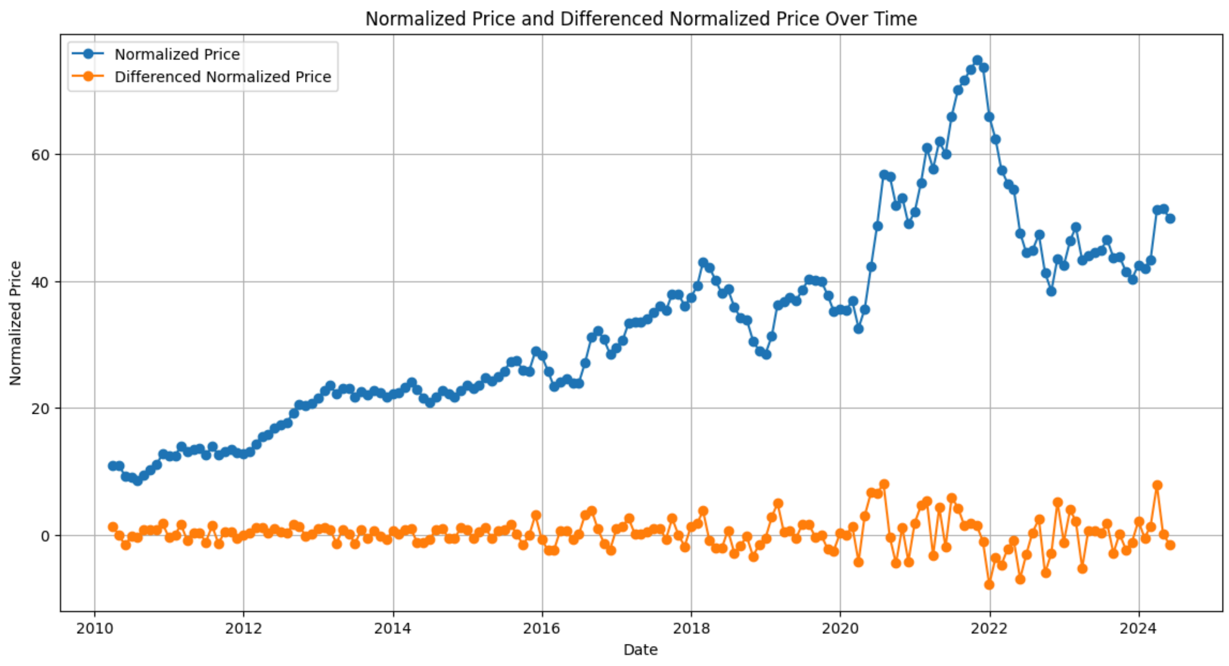
```
monthly_data['NormalizedPrice'] = (monthly_data['Close'] - monthly_data['Close'].min()) / (monthly_data['Close'].max() - monthly_data['Close'].min(
```

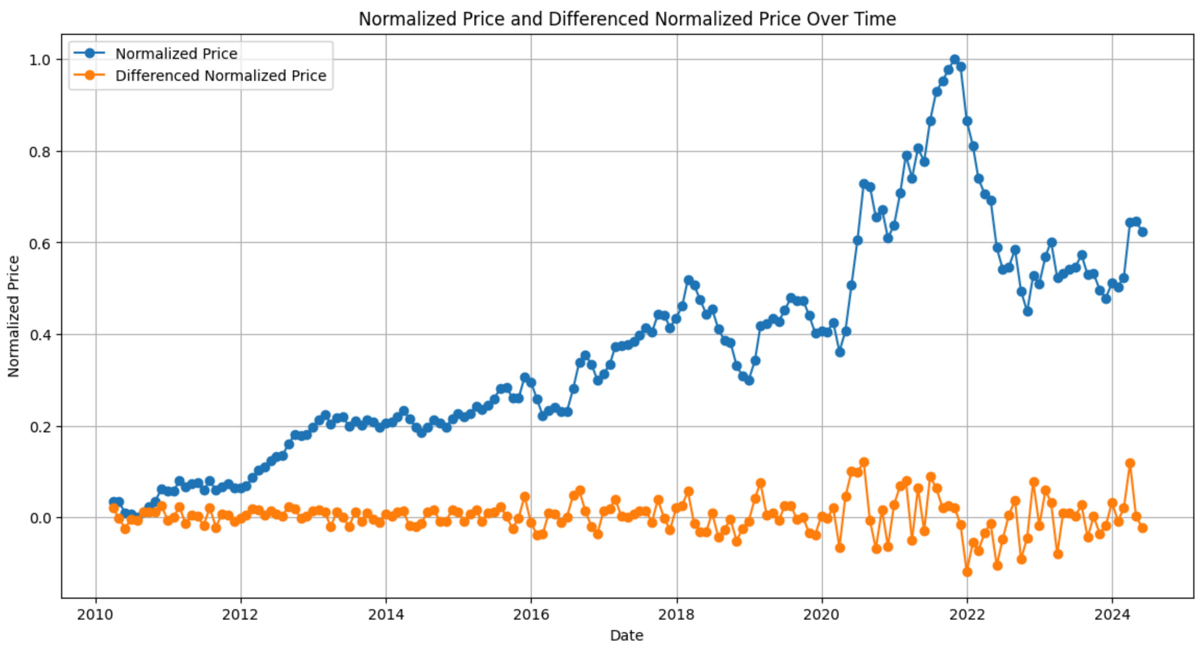
After this Step, take the differentiation “diff( )” of the data. After the First diff ( ), the P value of the data is 3.624430635934711e-18

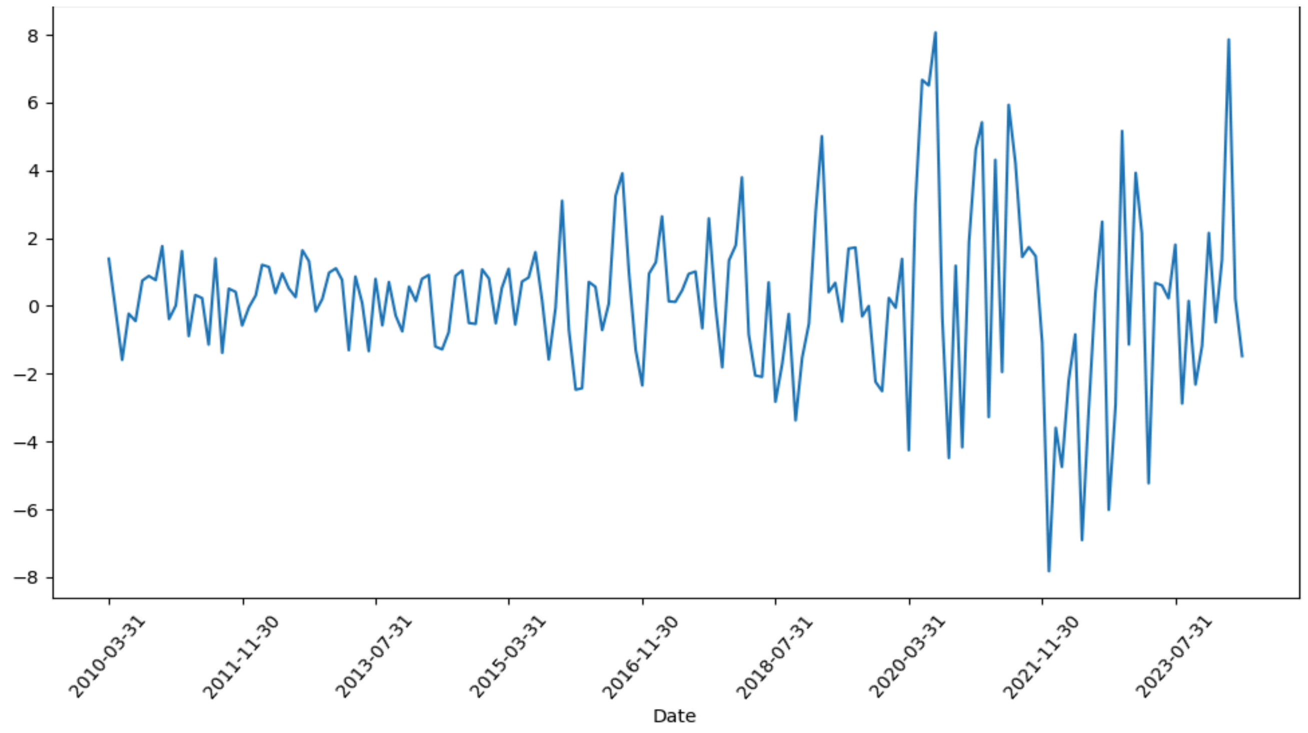
As P-value is not equal to zero and less than 0.05 so we can say data is normalized.

- Plotting:

Plotting a few graphs of the processed data to know about the trends of the data.







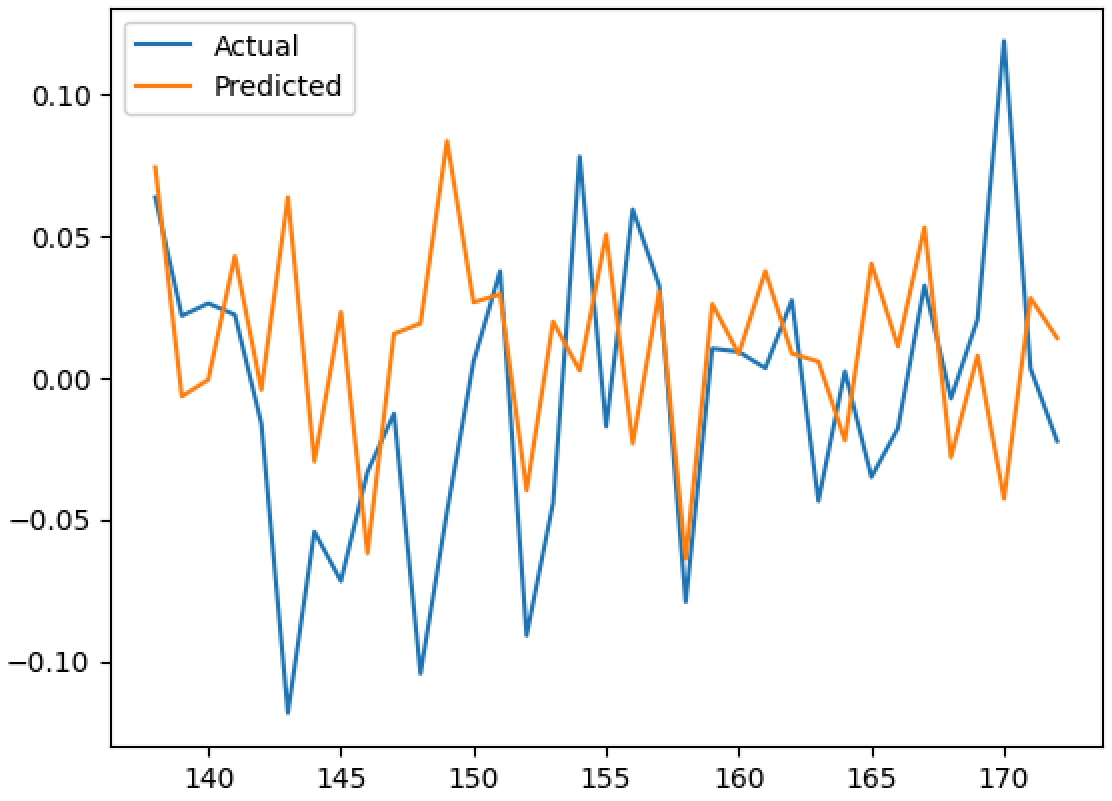
MODELS:

- **ARIMA:**

  Aim to forecast stock price movements using the ARIMA model.

  The dataset used in this analysis is the 'normalize_diff' column, which represents the normalized difference in stock prices. The dataset is split into training and test sets, with the training set comprising 80% of the data.:

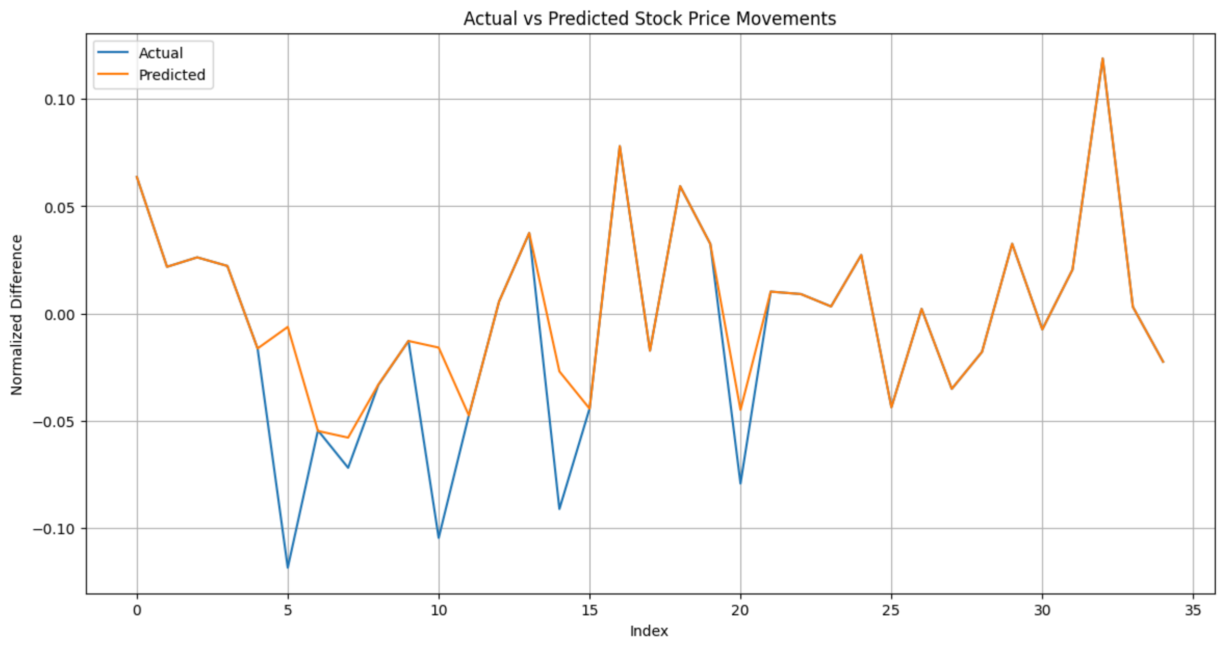
  We begin by fitting an ARIMA model to the training data. The ARIMA model is specified with an order of (22, 1, 22), indicating the number of lag observations, differences, and moving average terms to use in the model, respectively.



- **ANN:**

  Use of ANN to predict stock price movements based on historical data. The dataset used in this is the 'normalize_diff' column representing the normalized difference in stock prices. The dataset is split into training and test sets, with 80% of the data used for training and 20% for testing.

  The data is preprocessed by normalizing the features using MinMaxScaler. An ANN model is then constructed using the Keras framework with three dense layers. The model is compiled with the mean squared error loss function and the Adam optimizer

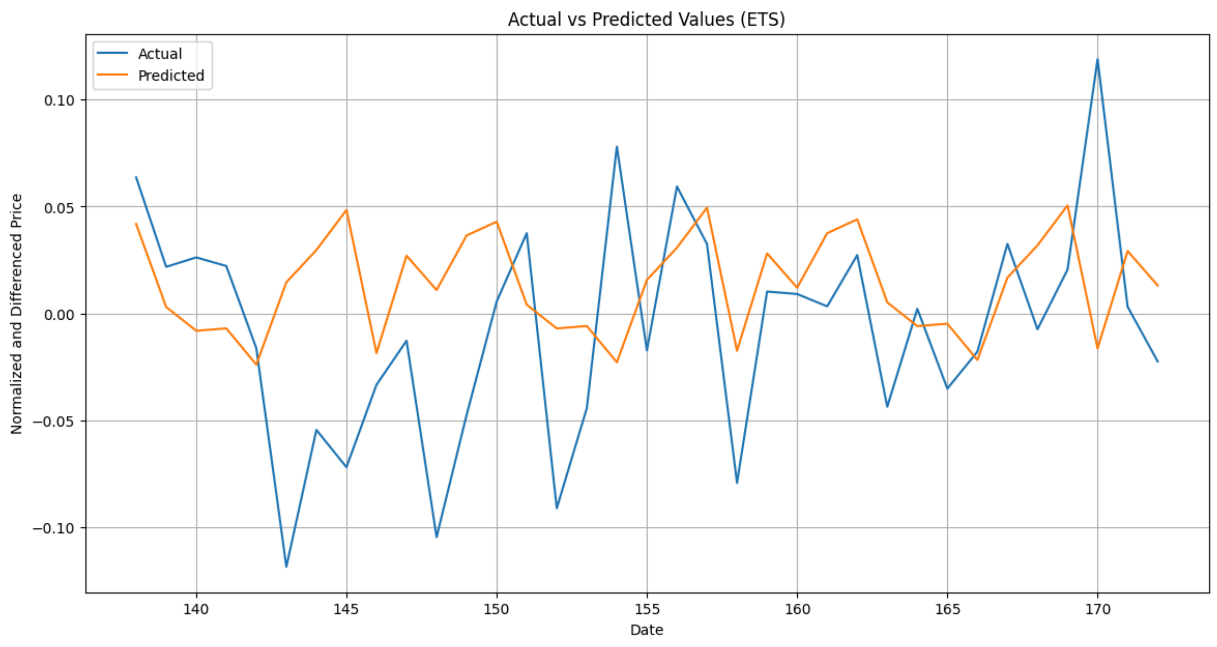


- **ETS:**

Exponential Smoothing is a popular technique for time series forecasting that assigns exponentially decreasing weights to past observations.

The dataset used is the 'normalize_diff' column representing the normalized difference in stock prices. The dataset is split into training and test sets, with 80% of the data used for training and 20% for testing.

The Exponential Smoothing model is implemented using the Holt-Winters method with additive trend, additive seasonal components, and a seasonal period of 12 months. The model is trained on the training data and then used to make predictions for the next two months

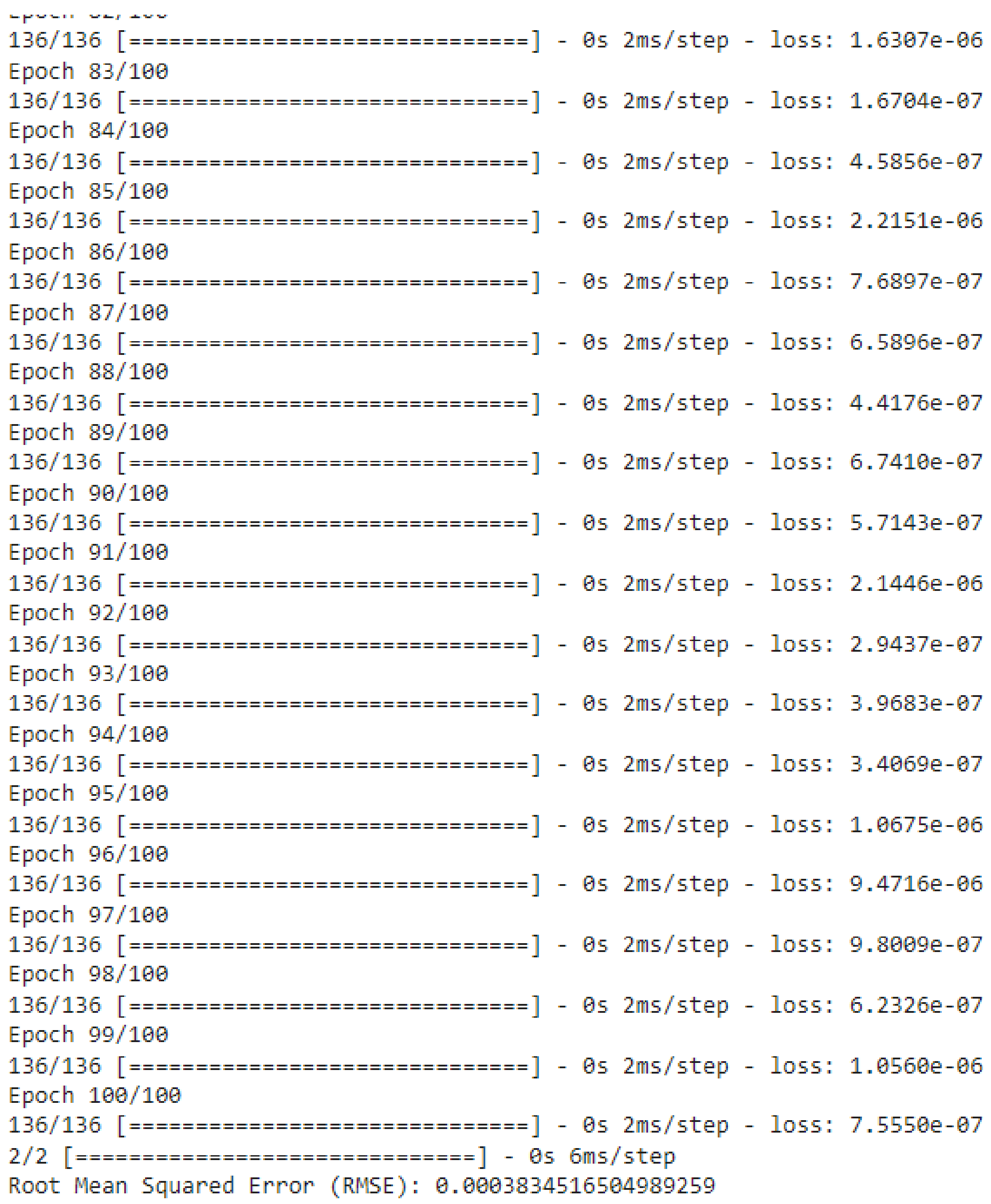


- **LSTM:**

Explores the LSTM neural networks for forecasting stock price movements based on historical data.

The 'normalize_diff' column represents the normalized difference in stock prices. The data is preprocessed and scaled using a MinMaxScaler to ensure compatibility with the LSTM model.

The LSTM model is implemented using the TensorFlow Keras API. The model architecture consists of a single LSTM layer with 50 units followed by a dense layer. The model is trained on 80% of the data and tested on the remaining 20%. The mean squared error is used as the loss function during training



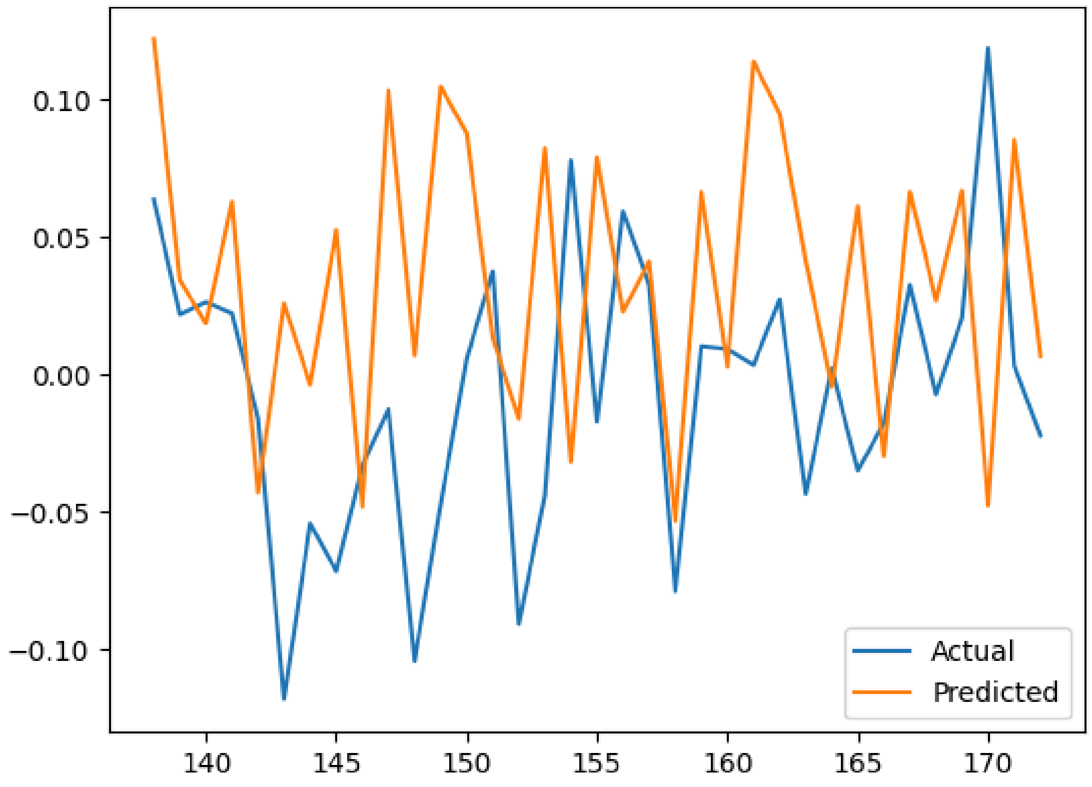
```
136/136 [==============================] - 0s 2ms/step - loss: 1.6307e-06
Epoch 83/100
136/136 [==============================] - 0s 2ms/step - loss: 1.6704e-07
Epoch 84/100
136/136 [==============================] - 0s 2ms/step - loss: 4.5856e-07
Epoch 85/100
136/136 [==============================] - 0s 2ms/step - loss: 2.2151e-06
Epoch 86/100
136/136 [==============================] - 0s 2ms/step - loss: 7.6897e-07
Epoch 87/100
136/136 [==============================] - 0s 2ms/step - loss: 6.5896e-07
Epoch 88/100
136/136 [==============================] - 0s 2ms/step - loss: 4.4176e-07
Epoch 89/100
136/136 [==============================] - 0s 2ms/step - loss: 6.7410e-07
Epoch 90/100
136/136 [==============================] - 0s 2ms/step - loss: 5.7143e-07
Epoch 91/100
136/136 [==============================] - 0s 2ms/step - loss: 2.1446e-06
Epoch 92/100
136/136 [==============================] - 0s 2ms/step - loss: 2.9437e-07
Epoch 93/100
136/136 [==============================] - 0s 2ms/step - loss: 3.9683e-07
Epoch 94/100
136/136 [==============================] - 0s 2ms/step - loss: 3.4069e-07
Epoch 95/100
136/136 [==============================] - 0s 2ms/step - loss: 1.0675e-06
Epoch 96/100
136/136 [==============================] - 0s 2ms/step - loss: 9.4716e-06
Epoch 97/100
136/136 [==============================] - 0s 2ms/step - loss: 9.8009e-07
Epoch 98/100
136/136 [==============================] - 0s 2ms/step - loss: 6.2326e-07
Epoch 99/100
136/136 [==============================] - 0s 2ms/step - loss: 1.0560e-06
Epoch 100/100
136/136 [==============================] - 0s 2ms/step - loss: 7.5550e-07
2/2 [==============================] - 0s 6ms/step
Root Mean Squared Error (RMSE): 0.0003834516504989259
```

- **SARIMA**:

SARIMA model for forecasting stock price movements based on historical data.

'normalize_diff' column representing the normalized difference in stock prices. The data is split into training and test sets, with 80% of the data used for training and 20% for testing.

The SARIMA model is implemented using the SARIMAX class from the statsmodels library. The model is trained on the training set with an order of (22, 1, 22) and a seasonal order of (0, 1, 0, 12), assuming a seasonal period of 12 months. The model is then used to make predictions on the test set.



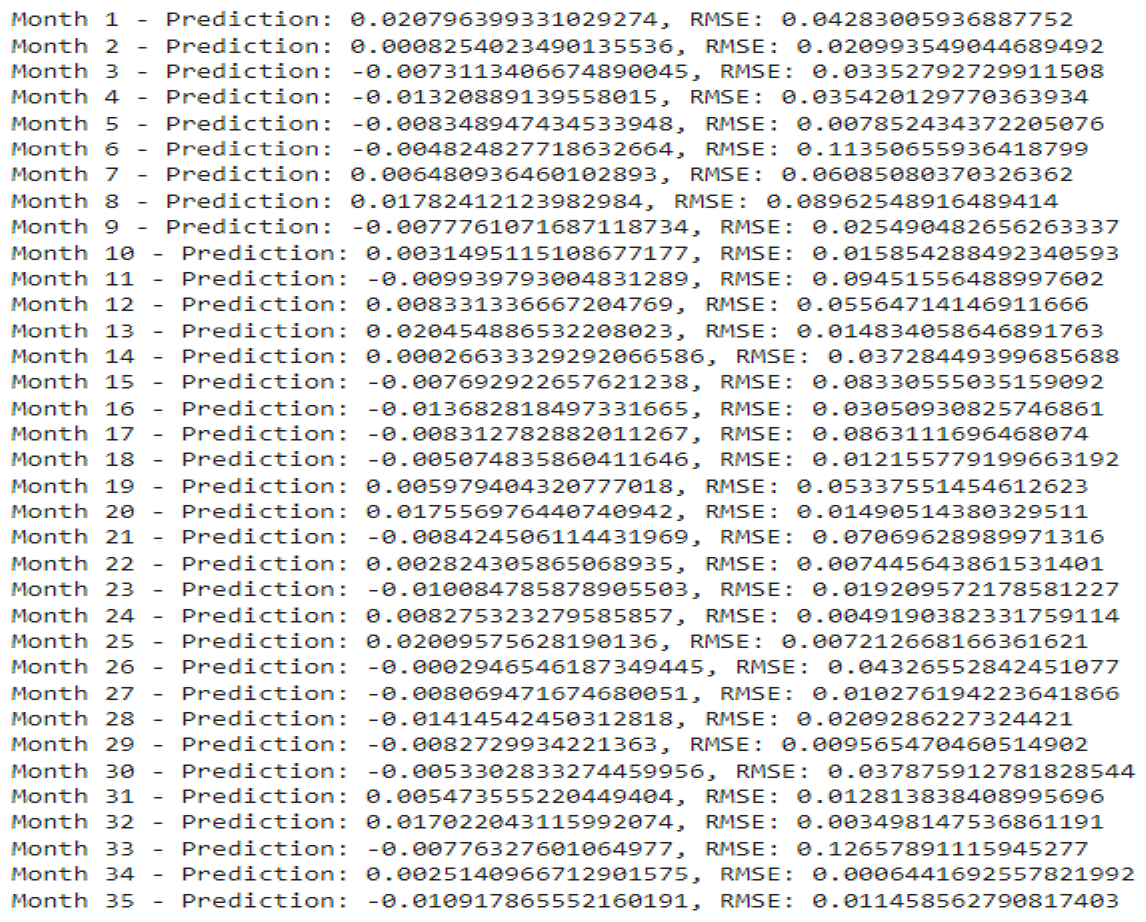
```
Root Mean Squared Error (RMSE): 0.07980269350430785
```

- **PROPHET:**

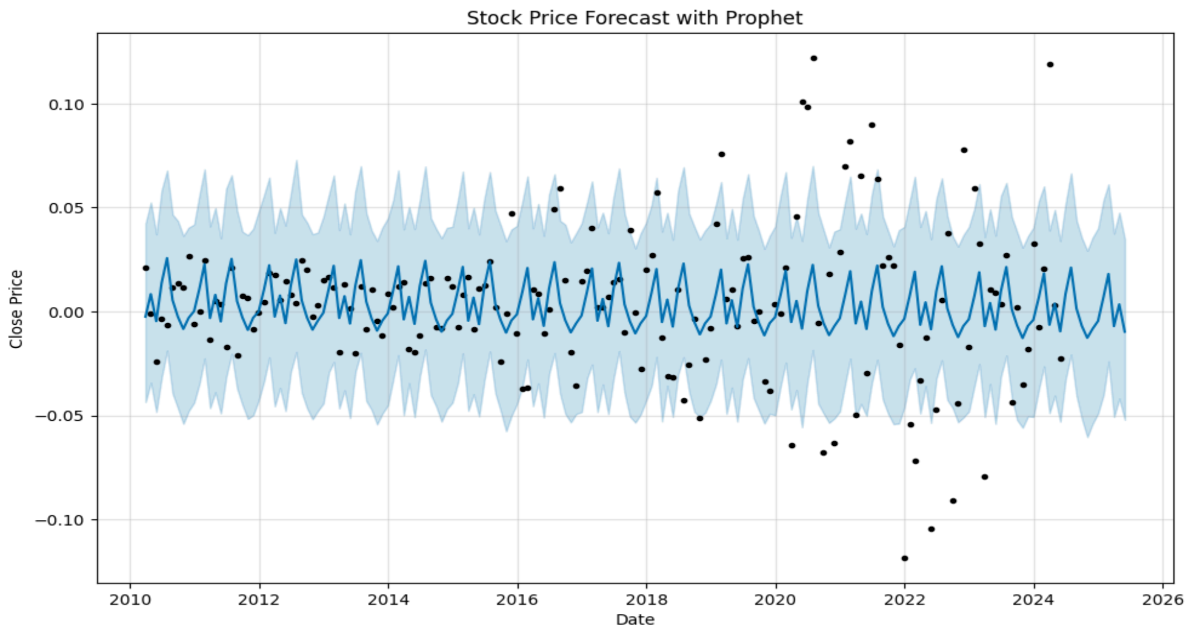
Prophet forecasting model for predicting stock price movements based on historical data. Monthly stock price data, with the 'normalize_diff' column representing the normalized difference in stock prices.

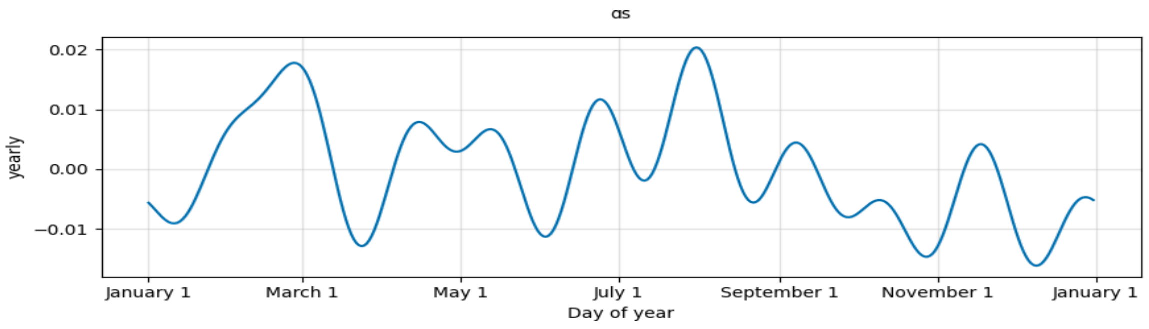
The data is prepared for Prophet by renaming the columns to 'ds' and 'y', where 'ds' represents the date and 'y' represents the target variable (normalized difference in stock prices).

The Prophet model is implemented using the Prophet class from the prophet library. The model is trained on the prepared data and used to make future predictions for the next 12 months. The forecasted values are then visualized using the plot and plot_components functions provided by Prophet.



```
Month 1 - Prediction: 0.020796399331029274, RMSE: 0.04283005936887752
Month 2 - Prediction: 0.0008254023490135536, RMSE: 0.020993549044689492
Month 3 - Prediction: -0.007311340667489045, RMSE: 0.03352792729911508
Month 4 - Prediction: -0.01320889139558015, RMSE: 0.035420129770363934
Month 5 - Prediction: -0.008348947434533948, RMSE: 0.007852434372205076
Month 6 - Prediction: -0.004824827718632664, RMSE: 0.11350655936418799
Month 7 - Prediction: 0.006480936460102893, RMSE: 0.06085080370326362
Month 8 - Prediction: 0.01782412123982984, RMSE: 0.08962548916489414
Month 9 - Prediction: -0.0077761071687118734, RMSE: 0.025490482656263337
Month 10 - Prediction: 0.0031495115108677177, RMSE: 0.015854288492340593
Month 11 - Prediction: -0.009939793004831289, RMSE: 0.09451556488997602
Month 12 - Prediction: 0.008331336667204769, RMSE: 0.05564714146911666
Month 13 - Prediction: 0.020454886532208023, RMSE: 0.014834058646891763
Month 14 - Prediction: 0.00026633329292066586, RMSE: 0.03728449399685688
Month 15 - Prediction: -0.007692922657621238, RMSE: 0.08330555035159092
Month 16 - Prediction: -0.013682818497331665, RMSE: 0.03050930825746861
Month 17 - Prediction: -0.008312782882011267, RMSE: 0.0863111696468074
Month 18 - Prediction: -0.005074835860411646, RMSE: 0.012155779199663192
Month 19 - Prediction: 0.005979404320777018, RMSE: 0.05337551454612623
Month 20 - Prediction: 0.017556976440740942, RMSE: 0.01490514380329511
Month 21 - Prediction: -0.008424506114431969, RMSE: 0.07069628989971316
Month 22 - Prediction: 0.002824305865068935, RMSE: 0.007445643861531401
Month 23 - Prediction: -0.010084785878905503, RMSE: 0.019209572178581227
Month 24 - Prediction: 0.008275323279585857, RMSE: 0.0049190382331759114
Month 25 - Prediction: 0.02009575628190136, RMSE: 0.007212668166361621
Month 26 - Prediction: -0.0002946546187349445, RMSE: 0.04326552842451077
Month 27 - Prediction: -0.008069471674680051, RMSE: 0.010276194223641866
Month 28 - Prediction: -0.01414542450312818, RMSE: 0.0209286227324421
Month 29 - Prediction: -0.0082729934221363, RMSE: 0.009565470460514902
Month 30 - Prediction: -0.0053302833274459956, RMSE: 0.037875912781828544
Month 31 - Prediction: 0.005473555220449404, RMSE: 0.012813838408995696
Month 32 - Prediction: 0.017022043115992074, RMSE: 0.003498147536861191
Month 33 - Prediction: -0.00776327601064977, RMSE: 0.12657891115945277
Month 34 - Prediction: 0.0025140966712901575, RMSE: 0.0006441692557821992
Month 35 - Prediction: -0.010917865552160191, RMSE: 0.011458562790817403
```



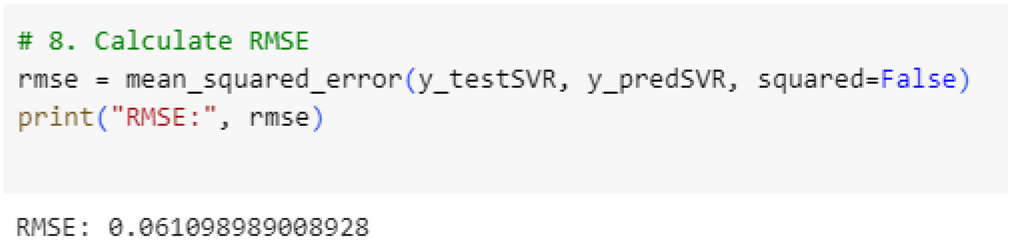


- **SVR:**
  SVR model is a type of supervised learning algorithm that learns a mapping from the input features to the target variable based on the training data. If the model has learned that the same input features lead to the same predicted output, it will produce the same prediction for both instances

  Feature Selection: Features such as 'Open', 'High', 'Low', and 'Adj Close' are selected for training the SVR model. The target variable is 'normalize_diff'.

  Train-Test Split: The dataset is split into training and test sets, with 80% of the data used for training and 20% for testing.

  Model Selection: The SVR model is selected with the best parameters ('C': 1, 'gamma': 'scale', 'kernel': 'linear').



```
# 8. Calculate RMSE
rmse = mean_squared_error(y_testSVR, y_predSVR, squared=False)
print("RMSE:", rmse)
```

```
RMSE: 0.061098989008928
```

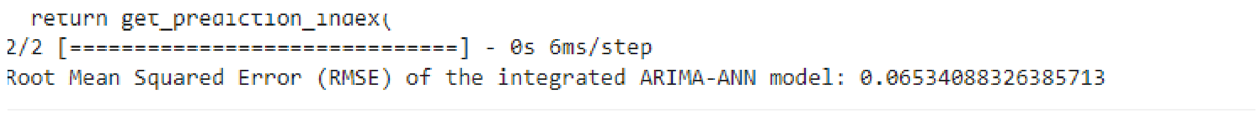
- **HYBRID (ARIMA - ANN) :**

ARIMA and ANN for predicting stock price movements. The ARIMA model is used to capture the time series patterns in the data, while the ANN is used to improve the prediction accuracy by learning from the residuals of the ARIMA model.

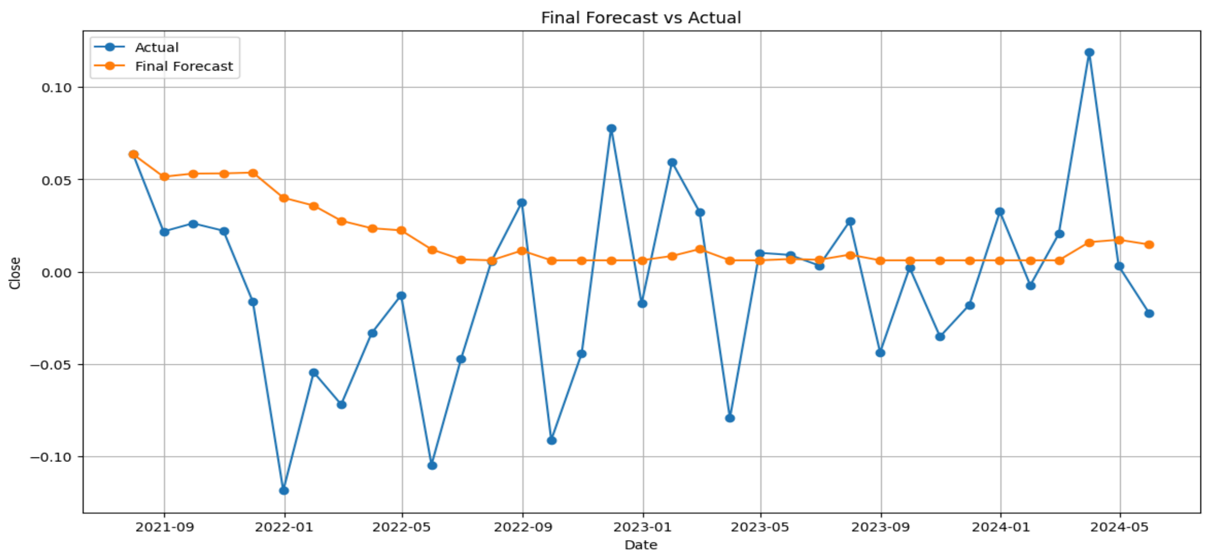
The dataset used in this analysis contains monthly stock price data, including features such as 'High', 'Open', and 'Low'. The target variable is 'normalize_diff', representing the normalized difference in stock prices. The dataset is split into training and test sets, with 80% of the data used for training and 20% for testing.

The ARIMA model is trained on the training data to capture the underlying time series patterns. The model is used to forecast future stock price movements on the test set, generating forecasted values.

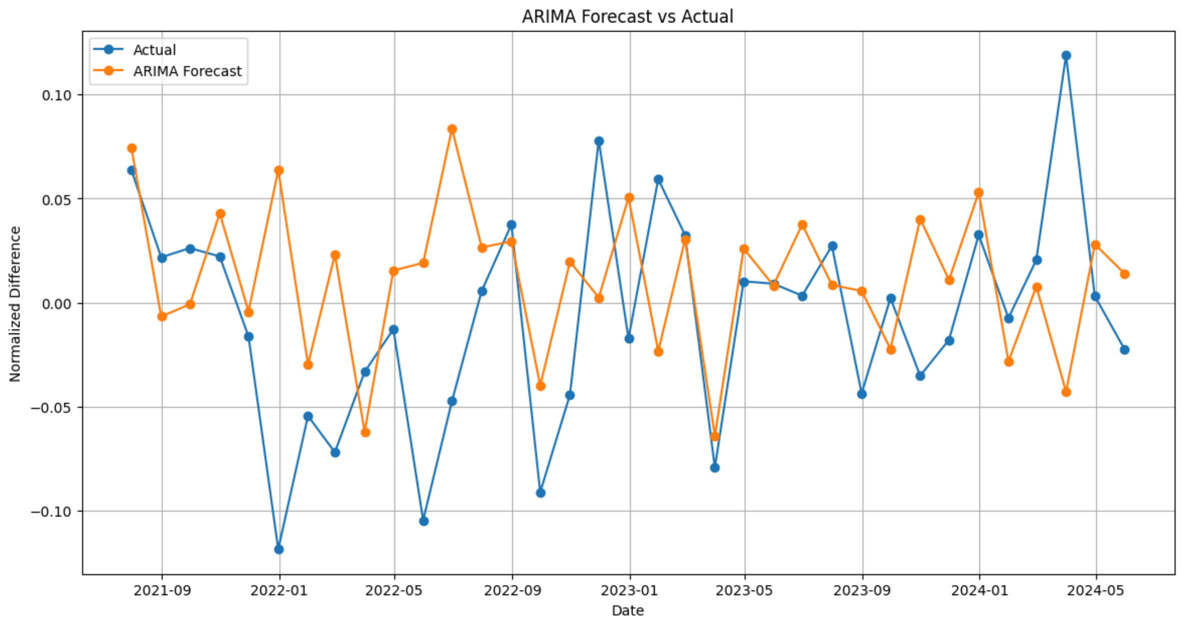
ANN model is constructed with input features 'High', 'Open', and 'Low', aiming to predict the residuals from the ARIMA model.



```
  return get_prediction_index(
2/2 [==============================] - 0s 6ms/step
Root Mean Squared Error (RMSE) of the integrated ARIMA-ANN model: 0.06534088326385713
```

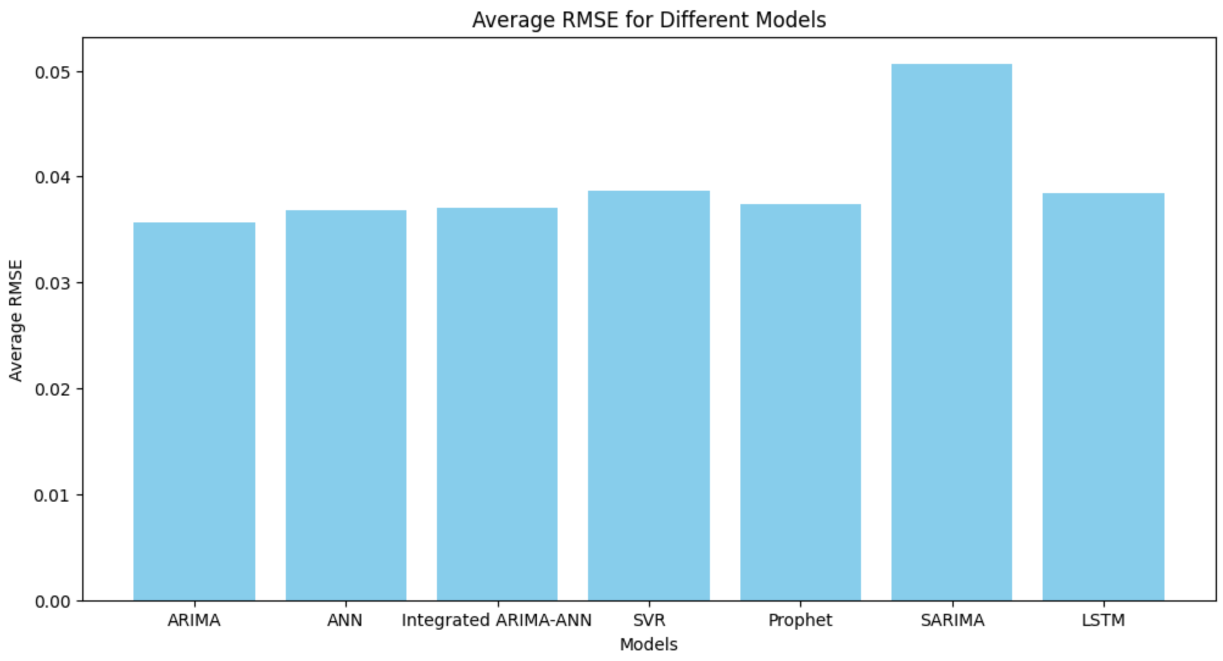


```
Root Mean Squared Error (RMSE) of the final forecast: 0.05819316056680427
```



CROSS VALIDATION:

Comparative Analysis of ARIMA, ANN, ETS, LSTM, SVR and Integrated ARIMA-ANN Models for Stock Price



FRONTEND:

Features:

1. Model Selection: Users can select from a list of machine learning models including ARIMA, ANN, SARIMA, ETS, Prophet, LSTM, and HYBRID ARIMA.
2. Predictions Display: Upon selecting a model, the app retrieves predictions made by that model from the backend and displays them on the web page.
3. Graphical Representation: Predictions are presented graphically using D3.js, a JavaScript library for data visualization. Line charts and bar charts are used to display predicted values and root mean square error (RMSE) values for different models.
4. Interactive Interface: The web interface is interactive, allowing users to select different models and view corresponding predictions in real-time.

Backend Implementation:

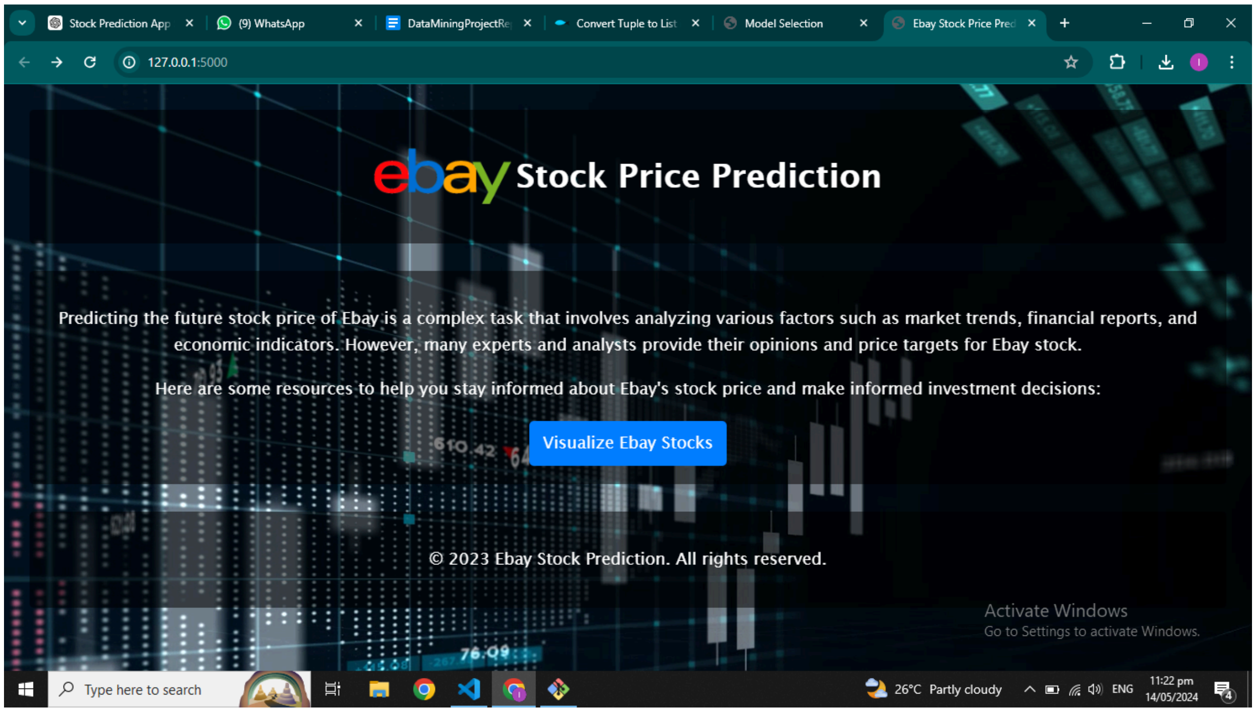
1. Flask Routes: The app defines Flask routes to handle different HTTP requests. For example, the / route serves the main HTML page, /get_predictions route fetches predictions from the backend, and so on.
2. Data Processing: Predictions data is processed in the backend to ensure it is in the correct format before being sent to the frontend. This includes filtering, formatting, and transforming data as necessary.
3. Error Handling: Error handling mechanisms are implemented to handle situations such as model not found, invalid requests, etc., ensuring a smooth user experience.

Frontend Implementation:

1. HTML Templates: HTML templates are used to structure the frontend of the web application. These templates are rendered dynamically using Flask's templating engine.
2. JavaScript Interactivity: JavaScript is used to make the web interface interactive. D3.js library is used for data visualization, allowing users to

interact with the charts and view detailed information.

3. Styling: CSS stylesheets are applied to enhance the visual appeal of the web pages and ensure a consistent user interface.

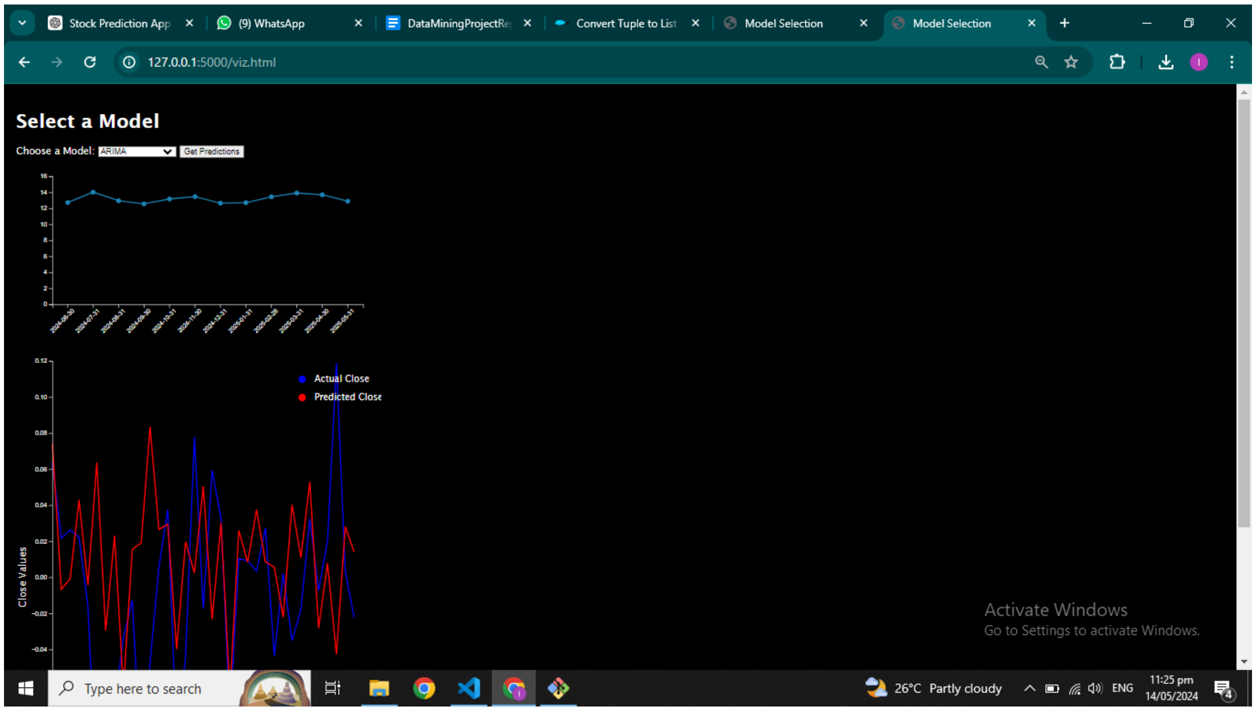


Stock Price Prediction

Predicting the future stock price of Ebay is a complex task that involves analyzing various factors such as market trends, financial reports, and economic indicators. However, many experts and analysts provide their opinions and price targets for Ebay stock.

Here are some resources to help you stay informed about Ebay's stock price and make informed investment decisions:

© 2023 Ebay Stock Prediction. All rights reserved.



Select a Model

Choose a Model: ARIMA