MACHINE LEARNING PROJECT

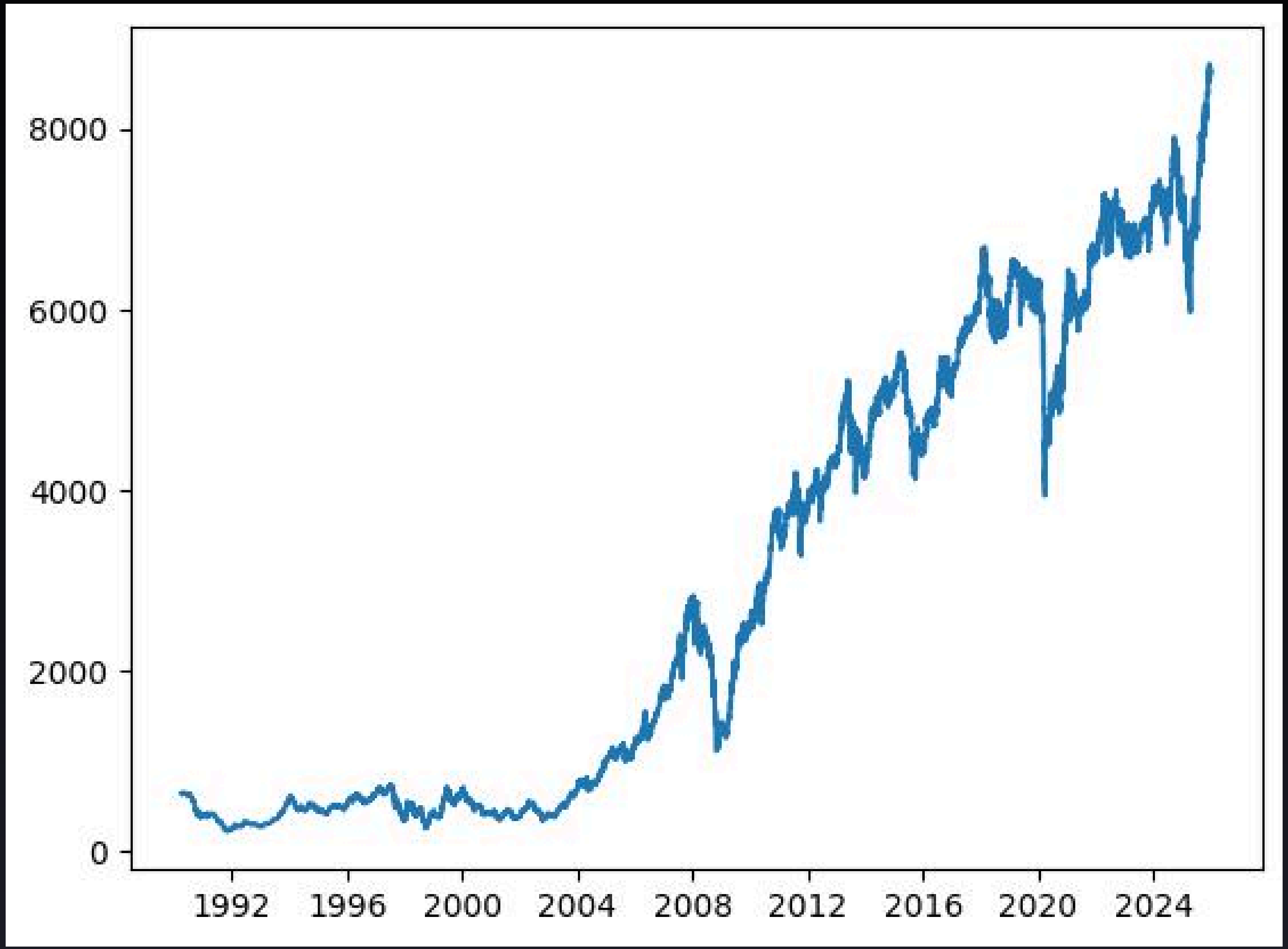
IDX Stock Prediction using LSTM

By,
Muhammad Zydan Priambada

Introduction

One of the emerging machine learning application in finance is stock prediction. Stock price data by default is a time series data, which is what LSTM (Long-Short Term Memory) model excels at. IDX composite is selected as an asset in order to measure how well does LSTM model perform on an emerging market.

By,
Muhammad Zydan Priambada



CHART

IDX Closing Price throughout the year

The stock price data is retrieved from yahoo finance, spanning from 1990 to 2025

```
import yfinance as yf
import pandas as pd
import matplotlib.pyplot as plt

idx = yf.Ticker('^JKSE')
idx = pd.DataFrame(idx.history(period='max'))

plt.plot(idx['Close'])
plt.show()
```

Access Jupyter Notebook:
https://github.com/dan9111/DataSciencePortfolioProject/blob/main/IDX%20Stock%20Prediction%20LSTM/IDX_Stock_Prediction_using_LSTM.ipynb

DATA SPLITTING

80% of the data will be used as a training, from 1990 to 2018

In a time series dataset, splitting the training data and testing data can only be done by slicing the data array into two parts in order to maintain the continuity.

DATA NORMALIZATION

Stock Prices are normalized before trained into the model, The LSTM will have a window size of 60, thus the model will take an input of 60 latest data to predict the next data.

Access Jupyter Notebook:
https://github.com/dan9111/DataSciencePortfolioProject/blob/main/IDX%20Stock%20Prediction%20LSTM/IDX_Stock_Prediction_using_LSTM.ipynb

DATA SPLITTING SNIPPET

```
import numpy as np

X = idx.values
window_size = 60
size = int(np.ceil(len(X)*0.8))
train, test = X[:size], X[size-window_size:]
```

DATA NORMALIZATION SNIPPET

```
from sklearn.preprocessing import
MinMaxScaler

scaler = MinMaxScaler()
scaler.fit(train)
train = scaler.transform(train)

X_train, y_train = [], []

for i in range(window_size, len(train)):
  X_train.append(train[i-window_size:i, 0])
  y_train.append(train[i, 0])

X_train, y_train = np.array(X_train),
np.array(y_train)
X_train = np.reshape(X_train,
(X_train.shape[0], X_train.shape[1], 1))
```

TENSORFLOW MODEL

LSTM Model Architecture

Layer (type)	Output Shape	Param #
lstm (LSTM)	(None, 60, 64)	16,896
lstm_1 (LSTM)	(None, 64)	33,024
dense (Dense)	(None, 128)	8,320
dropout (Dropout)	(None, 128)	0
dense_1 (Dense)	(None, 1)	129

The model consists of:

- Input LSTM layer from the 60 latest data, outputting 60×64 2D layer
- Hidden LSTM layer taking 60×64 data and outputting 1D 64 layer
- Hidden 128 dense layer
- Hidden 128 dropout layer to randomly deactivate neurons, prevent overfitting
- Output dense layer, resulting in 1 prediction data

The model is compiled with adam optimizer and mean absolute error for loss measurement

Access Jupyter Notebook:
https://github.com/dan9111/DataSciencePortfolioProject/blob/main/IDX%20Stock%20Prediction%20LSTM/IDX_Stock_Prediction_using_LSTM.ipynb

PREDICTION TEST

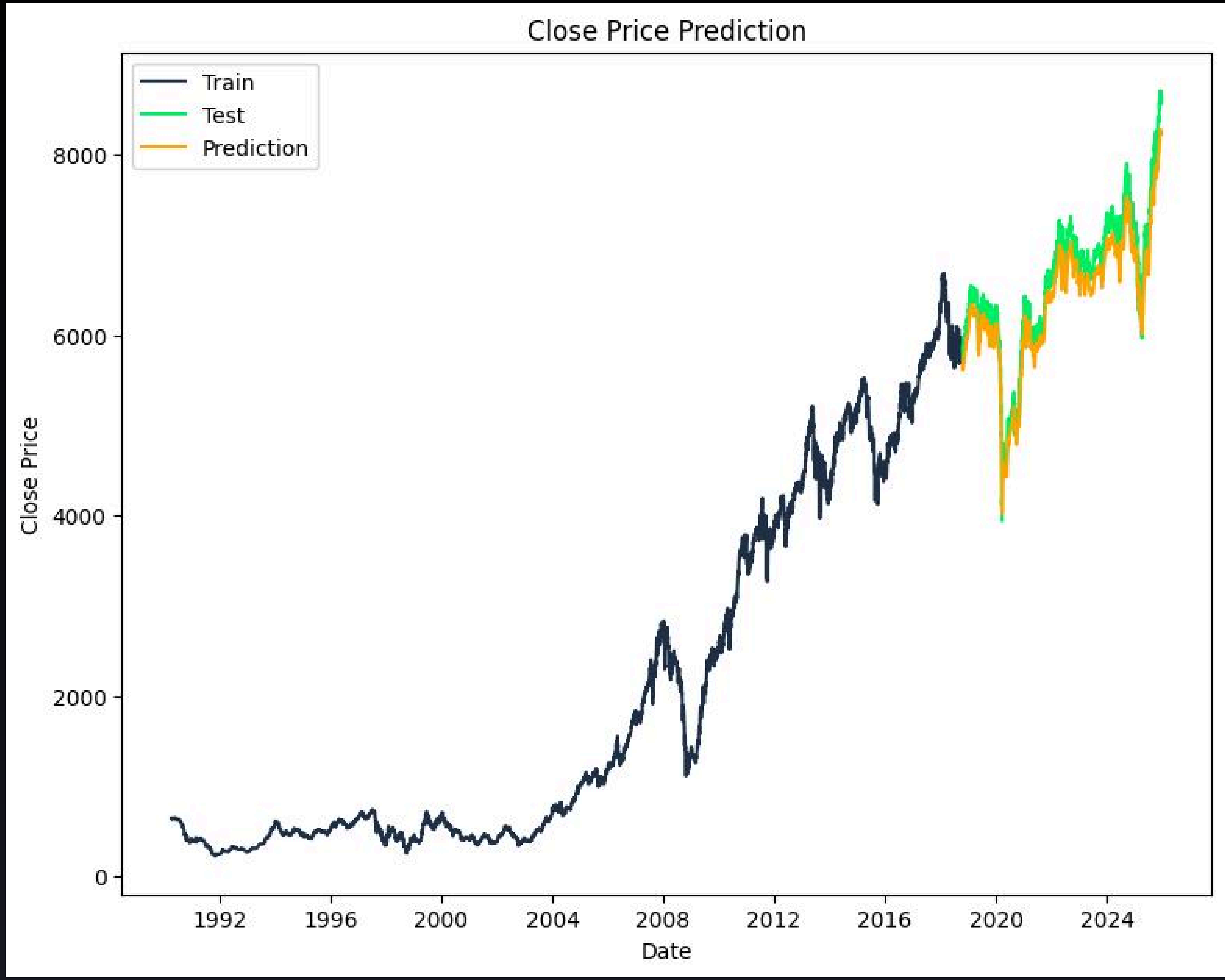
Comparing test dataset to model prediction

The shape of the prediction graph is closely resembling the actual graph

Prediction graph is slightly shifted down from the actual data. This might cause some concerns regarding the reliability of this model.

Both RMSE and Naive RMSE needs to be measured, if RMSE is higher than Naive RMSE, then the model is just predicting yesterday's price as today's price. The RMSE is lower than Naive RMSE so it has prediction skill. (RMSE = 227.61, Naive RMSE = 236.94)

In stock prediction what matters is the direction whether the price will go up or down, the directional accuracy is measured to be 49.86% thus the model is as good as a coin flip.



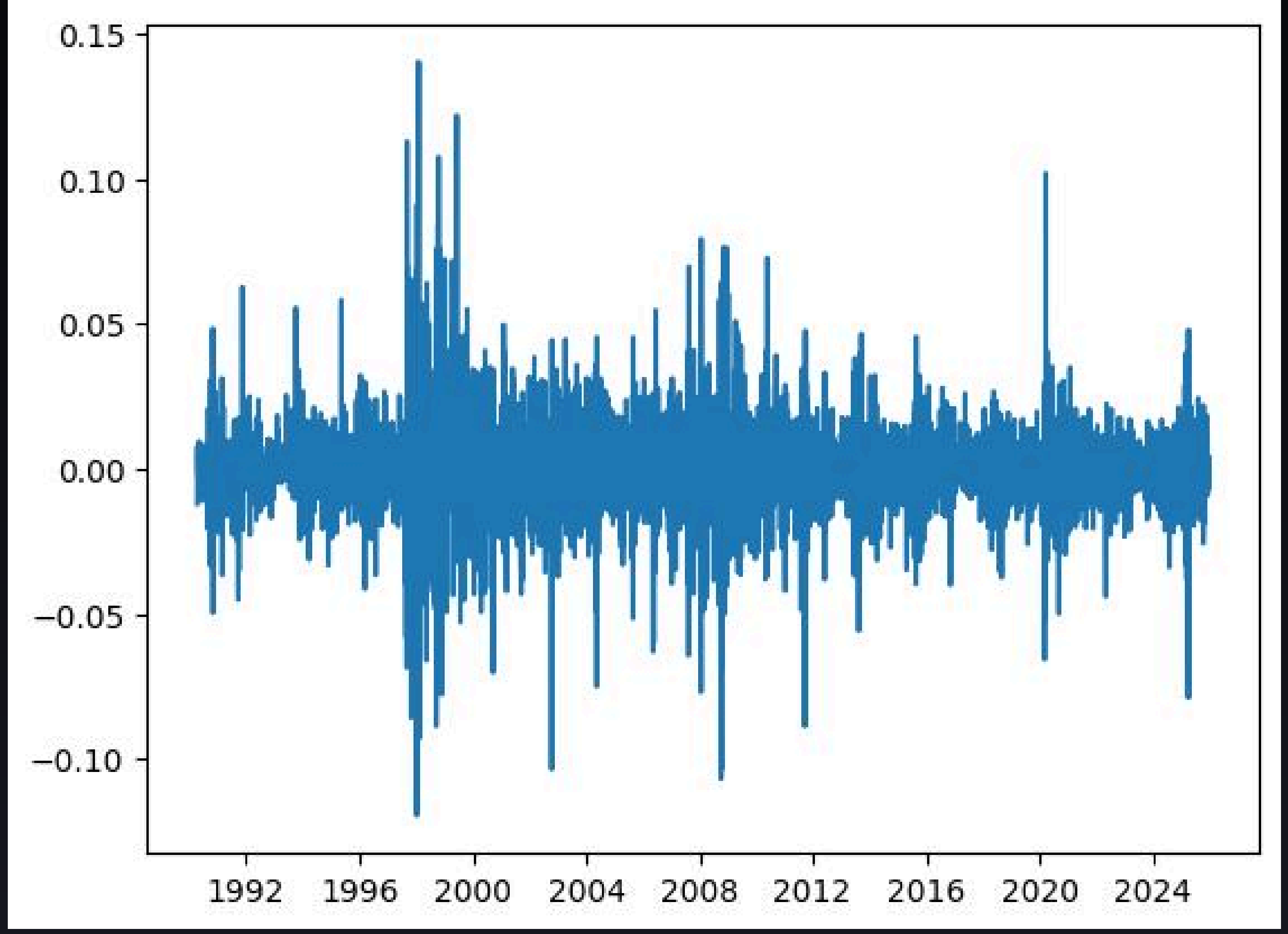

```
actual_change = np.diff(y_test.flatten())
predicted_change = np.diff(y_pred.flatten())

direction = np.sign(actual_change) == np.sign(predicted_change)
accuracy = np.mean(direction)*100

print(f'Accuracy: {accuracy:.2f}%')
```

```
Accuracy: 49.86%
```

Access Jupyter Notebook:
https://github.com/dan9111/DataSciencePortfolioProject/blob/main/IDX%20Stock%20Prediction%20LSTM/IDX_Stock_Prediction_using_LSTM.ipynb



Date	Return
1990-04-09 00:00:00+07:00	-0.012143
1990-04-10 00:00:00+07:00	-0.002204
1990-04-11 00:00:00+07:00	0.004124
1990-04-12 00:00:00+07:00	0.007753
1990-04-16 00:00:00+07:00	0.000625
...	...
2025-12-18 00:00:00+07:00	-0.006817
2025-12-19 00:00:00+07:00	-0.001003
2025-12-22 00:00:00+07:00	0.004215
2025-12-23 00:00:00+07:00	-0.007063
2025-12-24 00:00:00+07:00	-0.001436

8701 rows × 1 columns

SWITCH PARAMETER

Using daily return as a new parameter

In real world application, what matters in stock prediction is whether you can get a positive return or a negative return. Instead of using closing price, daily return of a stock will be used.

Access Jupyter Notebook:
https://github.com/dan9111/DataSciencePortfolioProject/blob/main/IDX%20Stock%20Prediction%20LSTM/IDX_Stock_Prediction_using_LSTM.ipynb

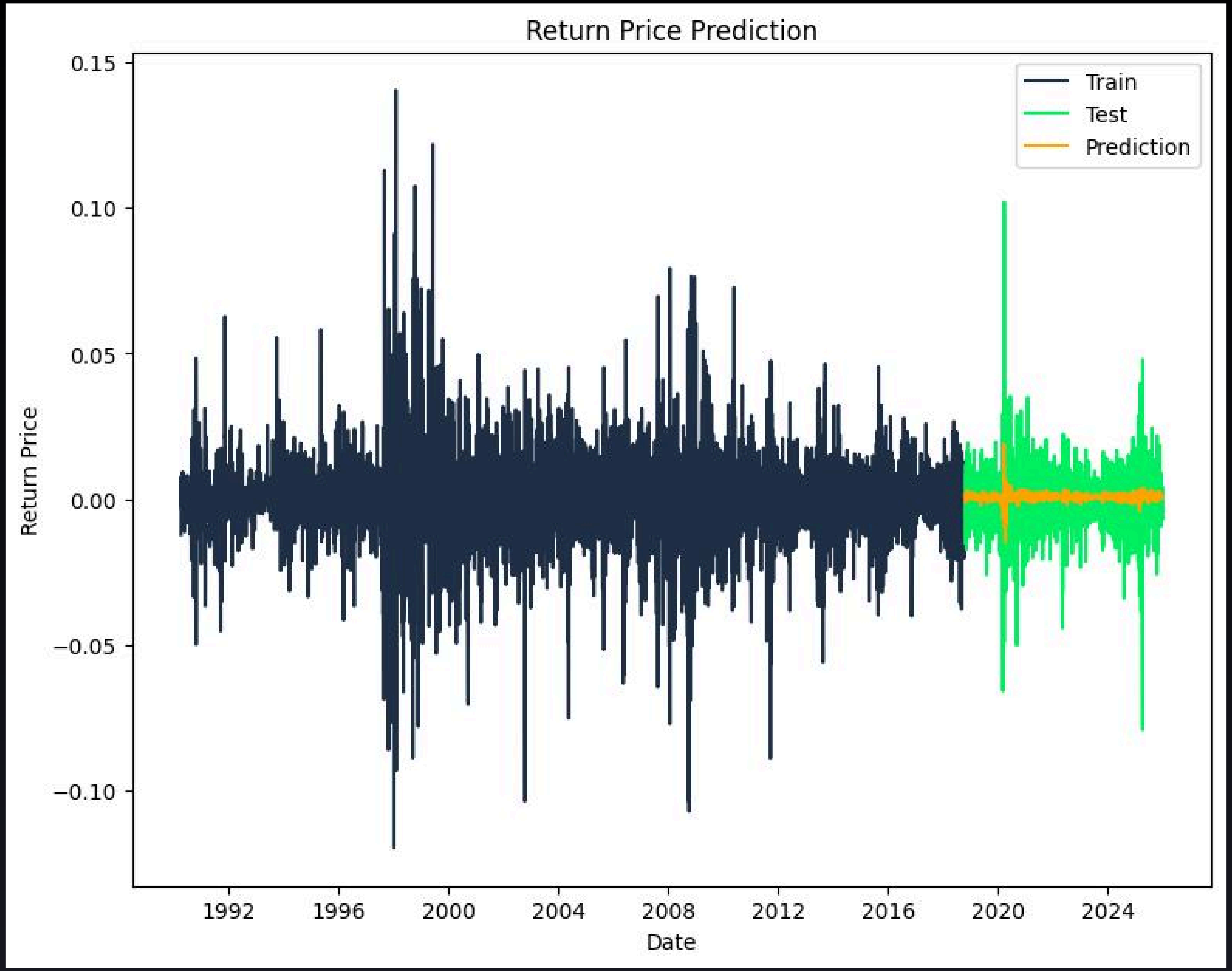
PREDICTION TEST

Comparing test dataset to model prediction (daily return)

Using the same model, the prediction has a smaller magnitude of return than the actual data

For the RMSE and Naive RMSE comparison, the RMSE is higher than Naive RMSE so this model can't use daily return as a parameter. (RMSE = 0.01026, Naive RMSE = 0.01025)

The directional accuracy is measured to be 31.99%, having way worse accuracy than using the closing price as a parameter.




```
actual_change = np.diff(y_test.flatten())
predicted_change = np.diff(y_pred.flatten())

direction = np.sign(actual_change) == np.sign(predicted_change)
accuracy = np.mean(direction)*100

print(f'Accuracy: {accuracy:.2f}%')
```

```
Accuracy: 31.99%
```

Access Jupyter Notebook:
https://github.com/dan9111/DataSciencePortfolioProject/blob/main/IDX%20Stock%20Prediction%20LSTM/IDX_Stock_Prediction_using_LSTM.ipynb

CONCLUSION

What can be gained from this experiment?

The model shows a prediction graph that resembles the actual graph.

However in real world applications this model still has a lot of flaws, the model itself cannot predict exactly when is the price going up or down. With the current accuracy of 49.86% the model is just guessing.

Using the daily return as a parameter results in worse accuracy. This means that the current model is unable to predict the IDX stock price

Since IDX is an emerging market, a simple LSTM model cannot represent most of the market movement. There are additional external factors that directly influences the market, such as economic policies both national and international.

Thank You

CONTACT ME FOR MORE INFORMATION:
EMAIL: DAN80308@GMAIL.COM
LINKEDIN: HTTPS://WWW.LINKEDIN.COM/IN/MUHAMMADZYDAN