Stock Price Prediction with LSTM

AMAN KHARWAL / JANUARY 3, 2022 / MACHINE LEARNING

LSTM stands for Long Short Term Memory Networks. It is a type of recurrent neural network that is commonly used for regression and time series forecasting in machine learning. It can memorize data for long periods, which differentiates LSTM neural networks from other **neural networks**. If you want to learn how to predict stock prices with LSTM, this article is for you. In this article, I will walk you through the task of stock price prediction with LSTM using Python.

Stock Price Prediction with LSTM

Using LSTM is one of the best machine learning approaches for time series forecasting. LSTMs are recurrent neural networks designed to remember data for a longer period. So, whenever you are working on a problem where your neural network fails to memorize data, you can use LSTM neural network. You can read more about LSTMs **here**.

Now in this section, I will take you through the task of Stock price prediction with LSTM using the Python programming language. I

will start this task by importing all the necessary Python libraries and collecting the latest stock price data of Apple:

```
import pandas as pd
import yfinance as yf
import datetime
from datetime import date, timedelta
today = date.today()

d1 = today.strftime("%Y-%m-%d")
end_date = d1
d2 = date.today() - timedelta(days=5000)
d2 = d2.strftime("%Y-%m-%d")
start_date = d2

data = yf.download('AAPL',
                   start=start_date,
                   end=end_date,
                   progress=False)
data["Date"] = data.index
data = data[["Date", "Open", "High", "Low", "Close"
             "Adj Close", "Volume"]]
data.reset_index(drop=True, inplace=True)
data.tail()
```

```
           Date        Open        High  ...       Close   Adj Close
Volume
3441 2021-12-27  177.089996  180.419998  ...  180.330002  180.330002
74919600
3442 2021-12-28  180.160004  181.330002  ...  179.289993  179.289993
79144300
3443 2021-12-29  179.330002  180.630005  ...  179.380005  179.380005
62348900
3444 2021-12-30  179.470001  180.570007  ...  178.199997  178.199997
59773000
3445 2021-12-31  178.089996  179.229996  ...  177.570007  177.570007
64025500

[5 rows x 7 columns]
```

A candlestick chart gives a clear picture of the increase and decrease in stock prices, so let's visualize a candlestick chart of the data before moving further:

```
import plotly.graph_objects as go
```

```
figure = go.Figure(data=[go.Candlestick(x=data["Dat
                                        open=data["
                                        high=data["
                                        low=data["L
                                        close=data[
figure.update_layout(title = "Apple Stock Price Ana
                     xaxis_rangeslider_visible=Fals
figure.show()
```

Now let's have a look at the correlation of all the columns with the Close column as it is the target column:

```
correlation = data.corr()
print(correlation["Close"].sort_values(ascending=Fa
```

```
Close        1.000000
Low          0.999890
High         0.999887
Adj Close    0.999845
Open         0.999783
Volume      -0.496325
Name: Close, dtype: float64
```

Training LSTM for Stock Price Prediction

Now I will start with training an LSTM model for predicting stock prices. I will first split the data into training and test sets:

```
x = data[["Open", "High", "Low", "Volume"]]
y = data["Close"]
x = x.to_numpy()
y = y.to_numpy()
y = y.reshape(-1, 1)

from sklearn.model_selection import train_test_spli
xtrain, xtest, ytrain, ytest = train_test_split(x, y
```

Now I will prepare a neural network architecture for LSTM:

```
from keras.models import Sequential
from keras.layers import Dense, LSTM
model = Sequential()
model.add(LSTM(128, return_sequences=True, input_sha
model.add(LSTM(64, return_sequences=False))
model.add(Dense(25))
model.add(Dense(1))
model.summary()
```

```
Model: "sequential_6"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 lstm_12 (LSTM)              (None, 4, 128)            66560

 lstm_13 (LSTM)              (None, 64)                49408

 dense_12 (Dense)            (None, 25)                1625

 dense_13 (Dense)            (None, 1)                 26

=================================================================
Total params: 117,619
Trainable params: 117,619
Non-trainable params: 0
_________________________________________________________________
```

Now here is how we can train our neural network model for stock price prediction:

```
model.compile(optimizer='adam', loss='mean_squared_
model.fit(xtrain, ytrain, batch_size=1, epochs=30)
```

```
Epoch 1/30
2756/2756 [==============================] - 18s 5ms/step - loss:
6.7984
Epoch 2/30
2756/2756 [==============================] - 15s 5ms/step - loss:
4.4978
Epoch 3/30
2756/2756 [==============================] - 15s 5ms/step - loss:
5.6511
Epoch 4/30
2756/2756 [==============================] - 15s 5ms/step - loss:
6.8347
Epoch 5/30
2756/2756 [==============================] - 15s 5ms/step - loss:
9.5083
Epoch 6/30
2756/2756 [==============================] - 15s 5ms/step - loss:
7.4367
Epoch 7/30
2756/2756 [==============================] - 15s 5ms/step - loss:
4.3043
Epoch 8/30
2756/2756 [==============================] - 15s 6ms/step - loss:
4.2213
Epoch 9/30
2756/2756 [==============================] - 15s 5ms/step - loss:
5.7352
Epoch 10/30
2756/2756 [==============================] - 15s 6ms/step - loss:
5.2137
Epoch 11/30
2756/2756 [==============================] - 15s 5ms/step - loss:
6.0945
Epoch 12/30
2756/2756 [==============================] - 14s 5ms/step - loss:
4.1032
Epoch 13/30
2756/2756 [==============================] - 15s 5ms/step - loss:
4.3637
Epoch 14/30
2756/2756 [==============================] - 15s 5ms/step - loss:
6.2240
Epoch 15/30
2756/2756 [==============================] - 15s 5ms/step - loss:
1.9857
```

```
Epoch 16/30
2756/2756 [==============================] - 15s 5ms/step - loss:
6.3982
Epoch 17/30
2756/2756 [==============================] - 15s 5ms/step - loss:
3.3015
Epoch 18/30
2756/2756 [==============================] - 15s 5ms/step - loss:
3.9104
Epoch 19/30
2756/2756 [==============================] - 15s 5ms/step - loss:
4.6564
Epoch 20/30
2756/2756 [==============================] - 15s 5ms/step - loss:
3.3215
Epoch 21/30
2756/2756 [==============================] - 15s 6ms/step - loss:
4.3116
Epoch 22/30
2756/2756 [==============================] - 15s 5ms/step - loss:
2.8147
Epoch 23/30
2756/2756 [==============================] - 15s 5ms/step - loss:
5.7586
Epoch 24/30
2756/2756 [==============================] - 16s 6ms/step - loss:
4.1890
Epoch 25/30
2756/2756 [==============================] - 17s 6ms/step - loss:
3.6991
Epoch 26/30
2756/2756 [==============================] - 15s 6ms/step - loss:
4.0951
Epoch 27/30
2756/2756 [==============================] - 15s 5ms/step - loss:
3.5940
Epoch 28/30
2756/2756 [==============================] - 16s 6ms/step - loss:
3.7180
Epoch 29/30
2756/2756 [==============================] - 15s 6ms/step - loss:
3.5864
Epoch 30/30
2756/2756 [==============================] - 15s 5ms/step - loss:
3.7422
<keras.callbacks.History at 0x7f8c37686790>
```

Now let’s test this model by giving input values according to the features that we have used to train this model and predicting the final result:

```
import numpy as np
#features = [Open, High, Low, Adj Close, Volume]
features = np.array([[177.089996, 180.419998, 177.0
model.predict(features)
```

```
array([[179.95299]], dtype=float32)
```

So this is how we can use LSTM neural network architecture for the task of stock price prediction.

Summary

LSTM stands for Long Short Term Memory Networks. It is a recurrent neural network designed to remember data for longer. Using LSTM is one of the best machine learning approaches for time series forecasting. I hope you liked this article on predicting stock prices with LSTM using Python. Feel free to ask valuable questions in the comments section below.

Aman Kharwal

I'm a writer and data scientist on a mission to educate others about the incredible power of data📈.

ARTICLES: 1376

PREVIOUS NEXT

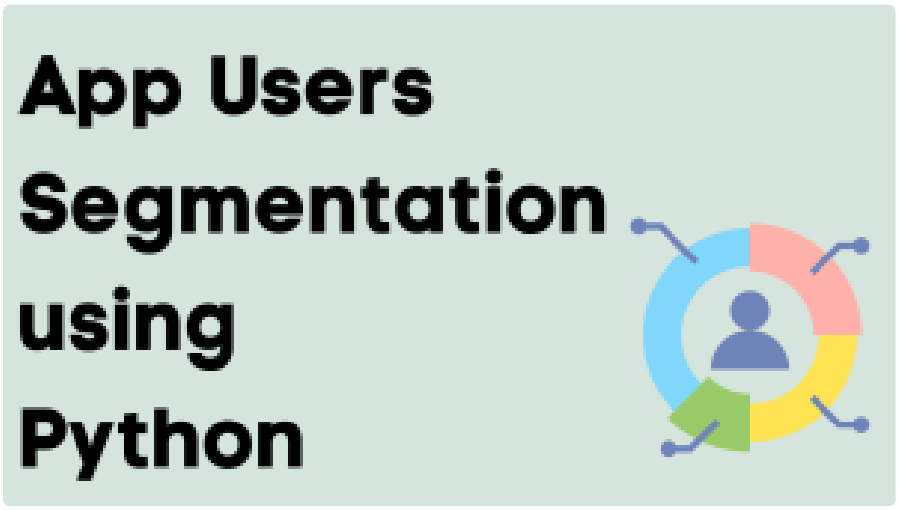
Recommended For You

Topic Modelling using Python

February 13, 2023

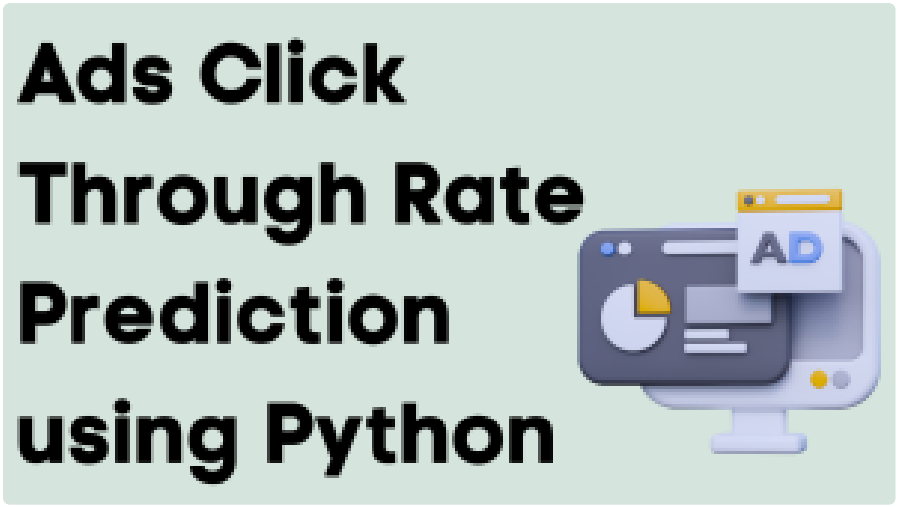
Text Emotions Classification using Python

February 6, 2023



App User Segmentation using Python

January 30, 2023



Ads Click Through Rate Prediction using Python

January 16, 2023 / 2 Comments

Leave a Reply

Enter your comment here...

FACEBOOK INSTAGRAM MEDIUM LINKEDIN