Multivariate Time Series Forecasting using RNN(LSTM)

Soubhik Khankary · Follow

5 min read · Jan 27, 2022

I was trying to forecast the future values of a variable where it not only depends on the previous values of itself but it also depends on the previous/current values of the other variables. In that case we consider that as multivariate time series forecasting problem.

EXAMPLE:

Let us consider a shop which is trying to sell two different Indian snacks i.e. Samosa and Kachori. He wants to forecast the number of samosas he must prepare next day to fulfill the demands of the customers. In this case let me give you a realistic example.

Samosa(available-yes),kachori(available-yes):

Consider a customer who has come with an intention to buy 10 samosas but as kachoris were also available in the shop. He ended up ordering 5 samosa

and 5 kachori . The sales of samosa dipped down because of the kachoris availability.

Samosa(available-yes),kachori(available-no)

Consider the same customer has come to the same shop with the intention of buying 5 samosas and 5 kachoris but because of the unavailability he ended up buying 10 samosas . The sales of samosas increased because of the unavailability of kachoris. The same could happen vice-versa.

In the case above the sales of samosa is not only dependent on its previous sales but also dependent on the current and past sales of kachori. Hence, it becomes multi-variate time series problem. Hope, it sounds clean and clear now.

PROBLEM STATEMENT:

Now we are going to solve the problem of forecasting the Open prices for stock of company 'XYZ' which is dependent on multiple other features like 'High', 'Low' & 'Close' price. In case if you are not clear with LSTM univariate please refer to the blog(https://medium.com/@soubhikkhankary28/univariate-time-series-forecasting-using-rnn-lstm-32702bd5cf4)

The Dataset is the same that I have used for forecasting future values. In the data above we will try to forecast the values for 'Open price' depending on other variables mentioned above. we have data from Jan 2012 to Dec 2016.

A quick look on the data set in excel first head():

	A	B	C	D	E	F
1	Date	Open	High	Low	Close	Volume
2	1/3/2012	325.25	332.83	324.97	663.59	7,380,500
3	1/4/2012	331.27	333.87	329.08	666.45	5,749,400
4	1/5/2012	329.83	330.75	326.89	657.21	6,590,300
5	1/6/2012	328.34	328.77	323.68	648.24	5,405,900
6	1/9/2012	322.04	322.29	309.46	620.76	11,688,800
7	1/10/2012	313.7	315.72	307.3	621.43	8,824,000
8	1/11/2012	310.59	313.52	309.4	624.25	4,817,800
9	1/12/2012	314.43	315.26	312.08	627.92	3,764,400
10	1/13/2012	311.96	312.3	309.37	623.28	4,631,800
11	1/17/2012	314.81	314.81	311.67	626.86	3,832,800
12	1/18/2012	312.14	315.82	309.9	631.18	5,544,000
13	1/19/2012	319.3	319.3	314.55	637.82	12,657,800
14	1/20/2012	294.16	294.4	289.76	584.39	21,231,800
15	1/23/2012	291.91	293.23	290.49	583.92	6,851,300
16	1/24/2012	292.07	292.74	287.92	579.34	6,134,400
17	1/25/2012	287.68	288.27	282.13	567.93	10,012,700
18	1/26/2012	284.92	286.17	281.22	566.54	6,476,500

Top rows in excel.

Please find the data set in excel for bottom rows(tail):

12/5/2016	757.71	763.9	752.9	762.52	1,394,200
12/6/2016	764.73	768.83	757.34	759.11	1,690,700
12/7/2016	761	771.36	755.8	771.19	1,761,000
12/8/2016	772.48	778.18	767.23	776.42	1,488,100
12/9/2016	780	789.43	779.02	789.29	1,821,900
12/12/2016	785.04	791.25	784.35	789.27	2,104,100
12/13/2016	793.9	804.38	793.34	796.1	2,145,200
12/14/2016	797.4	804	794.01	797.07	1,704,200
12/15/2016	797.34	803	792.92	797.85	1,626,500
12/16/2016	800.4	800.86	790.29	790.8	2,443,800
12/19/2016	790.22	797.66	786.27	794.2	1,232,100
12/20/2016	796.76	798.65	793.27	796.42	951,000
12/21/2016	795.84	796.68	787.1	794.56	1,211,300
12/22/2016	792.36	793.32	788.58	791.26	972,200
12/23/2016	790.9	792.74	787.28	789.91	623,400
12/27/2016	790.68	797.86	787.66	791.55	789,100
12/28/2016	793.7	794.23	783.2	785.05	1,153,800
12/29/2016	783.33	785.93	778.92	782.79	744,300
12/30/2016	782.75	782.78	770.41	771.82	1,770,000

Bottom rows of the data set

Let’s start the coding with python.

```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
%matplotlib inline
```

Importing necessary basic modules

I have split the dataset for stocks into train and test dataset. Train dataset is the one where I would fit and embed the RNN layers and then test it on test data set.

```
############Splitting the dataset####################
df1=pd.read_csv("Stock_Price_Train.csv")
df2=df1[['Date','Open','High','Low','Close']]
#df2.index=pd.to_datetime(df2['Date'])
df2_date=df1[['Date']]
df2_train=df2.iloc[0:1000,1:]
df2_train.head()
```

	Open	High	Low	Close
0	325.25	332.83	324.97	663.59
1	331.27	333.87	329.08	666.45
2	329.83	330.75	326.89	657.21
3	328.34	328.77	323.68	648.24
4	322.04	322.29	309.46	620.76

```
df2_test=df2.iloc[1000:,1:]
print(df2_test.shape)
df2_test['Close']=df2_test['Close'].str.replace(",","")
df2_test['Close']=df2_test['Close'].astype('float64')
df2_test.reset_index(inplace=True)
df2_test.head()
```

```
(258, 4)
```

	index	Open	High	Low	Close
0	1000	753.47	754.21	744.00	750.31
1	1001	749.55	751.35	746.62	748.40
2	1002	752.92	762.99	749.52	762.51
3	1003	766.69	779.98	766.43	776.60
4	1004	776.60	777.60	766.90	771.00

Seems like the 'Close' column is of object type. We need to convert into float type. Here is the code for doing so:

```
df2_train['Close']=df2_train['Close'].str.replace(",","")
df2_train['Close']=df2_train['Close'].astype('float64')
df2_train.head()
```

	Open	High	Low	Close
0	325.25	332.83	324.97	663.59
1	331.27	333.87	329.08	666.45
2	329.83	330.75	326.89	657.21
3	328.34	328.77	323.68	648.24
4	322.04	322.29	309.46	620.76

In neural network it is very necessary to scale the values in data frame. I am using Standard Scaler in this case to scale values for my train data set. Please use two different scaler objects to scale the columns. One is for input and other one for output.

```
from sklearn.preprocessing import StandardScaler
sc=StandardScaler()
df2_train_scaled=sc.fit_transform(df2_train)
df2_train_scaled
print(df2_train_scaled.shape)
```

```
(1000, 4)
```

```
sc2=StandardScaler()
df2_train_scaled_y=sc2.fit_transform(df2_train[['Open']])
print(df2_train_scaled_y.shape)
df2_train_scaled_y
```

```
(1000, 1)
```

Now I have to create an array as I did in my previous post for the previous values for each column. As I have explained in detail about it in previous post. Please refer to the example to understand it easily.

```
hops=14
no_records=1000
no_cols=4
X_train=[]
y_train=[]
for i in range(14,1000):
    X_train.append(df2_train_scaled[i-14:i])
    y_train.append(df2_train_scaled_y[i][0])
X_train,y_train=np.array(X_train),np.array(y_train)
```

```
print(X_train.shape)
print(y_train.shape)
```

```
(986, 14, 4)
(986,)
```

Reshape the array into 3-d as I did below. It is necessary to fit in RNN model.

```
X_train_shape=np.reshape(X_train,(X_train.shape[0],X_train.shape[1],X_train.shape[2]))
X_train_shape.shape
```

```
]: (986, 14, 4)
```

Now it has become easy to apply the RNN model with two embedded layers of LSTM layers and stack it with one dense layer.

SMALL EXPLANATION:

I have used two LSTM layers extensively and dropped out few neurons to make sure that my model doesn't overfit . Please make sure to play around with different types of optimizers and loss if required . You should be able to write your own functions to hyper-tune the parameters.

```
from keras.models import Sequential
from keras.layers import Dense, LSTM, Dropout
```

```
model=Sequential()
model.add(LSTM(units=50,return_sequences=True,input_shape=(14,4)))
model.add(Dropout(0.2))
model.add(LSTM(units=50))
model.add(Dropout(0.2))
model.add(Dense(1))
model.compile(optimizer='adam',loss='mean_squared_error')
```

Let's fit the model. I have used 100 epochs and batch size to 32 for each epochs.

```
model.fit(X_train_shape,y_train,epochs=100,batch_size=32)
```

```
31/31 [==============================] - 1s 27ms/step - loss: 0.0120
Epoch 45/100
31/31 [==============================] - 1s 26ms/step - loss: 0.0132
Epoch 46/100
31/31 [==============================] - 1s 27ms/step - loss: 0.0129
Epoch 47/100
31/31 [==============================] - 1s 27ms/step - loss: 0.0128
Epoch 48/100
31/31 [==============================] - 1s 27ms/step - loss: 0.0120
```

Preparation of Test Dataset:

```
#####preparation of test data set####################
df1_train_last14=df2_train.iloc[-14:]
df1_test_full=df2_test
full_df=pd.concat((df1_train_last14,df1_test_full),axis=0)
print(full_df.shape)
full_df.head()
#####################################################
```

```
(272, 4)
```

	Open	High	Low	Close
986	766.01	768.99	745.63	752.54
987	753.10	768.49	750.00	766.81
988	767.77	768.73	755.09	763.25
989	757.89	764.80	754.20	762.37
990	759.17	764.23	737.00	751.61

Rescaling the pandas dataframe>

```
###########Rescaling the test pandas dataframe#############
full_df=sc.transform(full_df)
full_df.shape
```

```
(272, 4)
```

Reshaping the test data frame in similar way as we did for training dataset.

```
###########Reshaping the test pandas dataframe#############
hops=14
no_records=272
no_cols=4
X_train_shape_pred=[]
for i in range(14,272):
    X_train_shape_pred.append(full_df[i-14:i])
X_train_shape_pred=np.array(X_train_shape_pred)
print(X_train_shape_pred.shape)
```

```
(258, 14, 4)
```

Try predicting the output for the Open Price and rescale it back to original value using inverse transform method.

```
len(X_train_shape_pred)
```

```
]: 258
```

```
ytest=model.predict(X_train_shape_pred)
```

```
y_final_pred=sc2.inverse_transform(ytest)
y_final_pred
```

Converted the predicted values Series into Dataframe as i did below:

```
final_open_pred=pd.DataFrame(final_open_pred)
final_open_pred.columns=['final_open_pred']
final_open_pred
```

]:

	final_open_pred
0	749.620911
1	748.185669
2	747.589661
3	753.019104
4	762.349976

Finally, Concatenating the actual and predicted values in same dataframe and trying to visualize the differences.

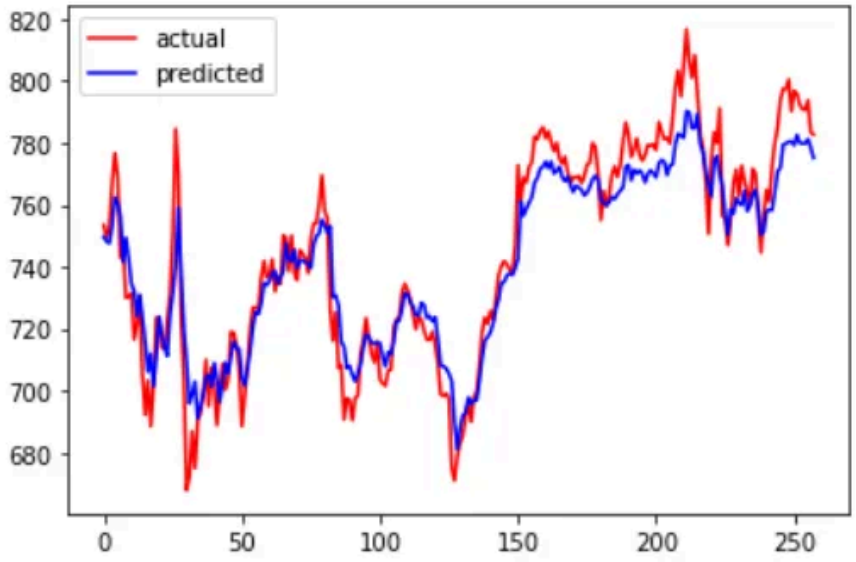
```
fully_final=pd.concat((final_open_pred,df2_test),axis=1)
fully_final=fully_final[['Open','final_open_pred','High','Low','Close']]
fully_final
```

	Open	final_open_pred	High	Low	Close
0	753.47	749.620911	754.21	744.00	750.31
1	749.55	748.185669	751.35	746.62	748.40
2	752.92	747.589661	762.99	749.52	762.51
3	766.69	753.019104	779.98	766.43	776.60
4	776.60	762.349976	777.60	766.90	771.00
...	...	...	...	...	...
253	790.90	779.770813	792.74	787.28	789.91
254	790.68	779.821777	797.86	787.66	791.55
255	793.70	781.177979	794.23	783.20	785.05
256	783.33	778.488098	785.93	778.92	782.79
257	782.75	775.146973	782.78	770.41	771.82

Plotting the actual and forecasted values together on the graph to visualize the differences.

```
plt.plot(fully_final['Open'],label='actual',color='red')
plt.plot(fully_final['final_open_pred'],label='predicted',color='blue')
plt.legend()
```

```
<matplotlib.legend.Legend at 0x2a236724fd0>
```



I see the forecast was quite close to actual values . I feel we all did a great job at the end. Hope I was able to explain you all very well how this forecasting worked. I will be coming up with many other methods which would be easier to implement and use. Thanks again!!!

In case of questions, please comment and I will be happy to help you out.

Mlearning.ai Submission Suggestions

How to become a writer on Mlearning.ai

medium.com

Machine Learning | Deep Learning | Time Series Analysis | Time Series Forecasting

Written by Soubhik Khankary

184 Followers · 13 Following

Follow

Data Engineer by job , Teaching computers by stats and love to learn never endless math.

Responses (7)

Write a response

What are your thoughts?

Ekruchowy
Mar 10, 2023

There is a problem with prediction part. You are making a prediction on test data as the input and testing it against it's self. So, this is not forcasting. You should be using a previous time window as the test input, then predicting what the future time window is, then compare that to the true results.

32 Reply

Muhammad Hafizi
Apr 9, 2022

Hi Soubhil, thanks for the explaination in this article. I have a question how does the multivariate work here mean how does the input variable work like for example does all the input are being sum together? Hope you can reach me in future as soon as possible.thanks

21 Reply

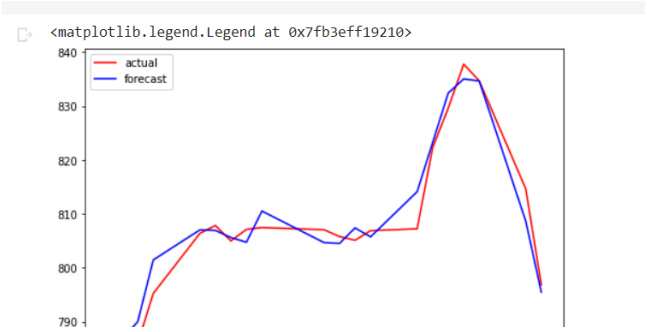
Andres Girardot
Feb 7, 2022

Hi Soubhik, thanks for doing this article. I was wondering if you had something similar but with a forecast of K steps ahead. So instead of forecasting tomorrows Open Price, we try to forecast the Open Price 5 days from now for example. Or if you... more

8 1 reply Reply

See all responses

More from Soubhik Khankary



Soubhik Khankary

Multivariate Time Series Forecasting using FBProphet

Maximilian Vogel

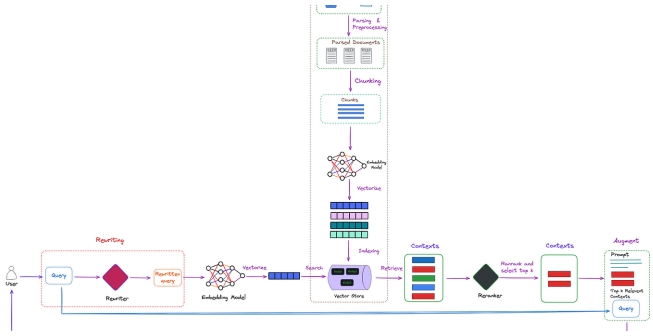
The ChatGPT list of lists: A collection of 3000+ prompts, GPT...

Hello Everyone, Hope you all are doing good. Today I have come up with a post which woul...

Jan 30, 2022 355 5

Updated Feb-16, 2025. Added New Introductions, Prompts, Lists and Tools

Feb 7, 2023 13.2K 162

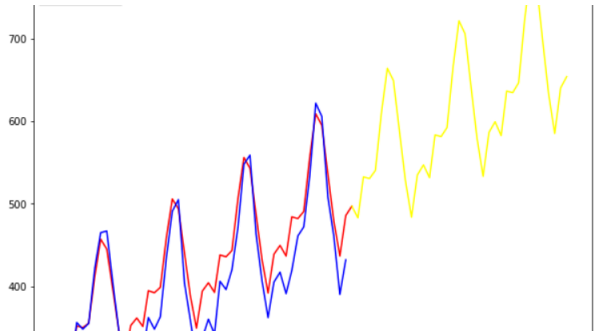


Florian June

Advanced RAG 06: Exploring Query Rewriting

A key technique for aligning the semantics of queries and documents

Mar 4, 2024 1.1K 6



Soubhik Khankary

Time Series forecasting using SARIMAX

Hello Everyone, In one of my previous post we discussed about how to forecast a variable...

Jan 31, 2022 35

See all from Soubhik Khankary

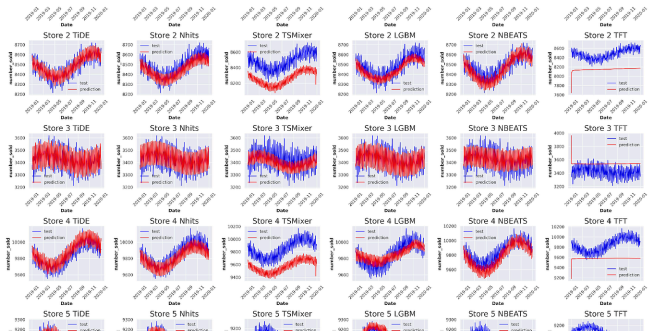
Recommended from Medium

Ajay Verma

Unlocking the Secrets of Time: A Deep Dive into Time Series...

Time series data—sequences of data points collected at specific points in time—are...

Oct 6, 2024 12

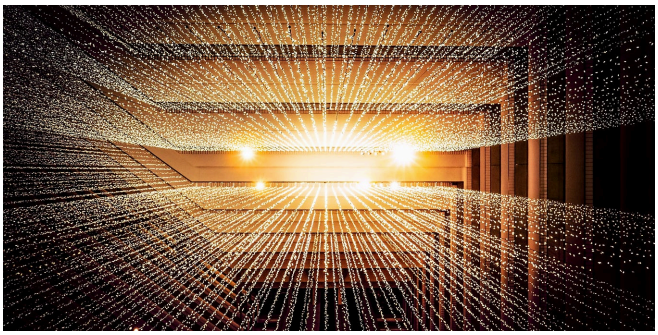


E Esther Cifuentes

Comparing Time Series Algorithms

Evaluating Leading Time Series Algorithm with Darts.

Oct 16, 2024 443 10



Ryan Martin

Variational Autoencoders for Timeseries Data Generation

An Innovative Approach to Generating Synthetic Timeseries Data through VAEs

Dec 9, 2024 2

Data Science In TDS Archive by Marco Peixeiro

iTransformer: The Latest Breakthrough in Time Series...

Discover the architecture of iTransformer and apply the model in a small experiment using...

Apr 9, 2024 830 9

shashank Jain

Forecasting Stock Prices with TSLANet: A Deep Dive into Time...

Introduction In today's data-driven world, accurately forecasting stock prices is...

Sep 30, 2024 7 2

In InsiderFinance Wire by Alexzap

Taker Buy/Sell Volume Ratio Forecasting with 30 Darts Models

A Comparative Analysis of Various Darts Time Series Forecasting & Hyperparameter...

Nov 25, 2024 155

See more recommendations