تمرین دوازدهم یادگیری عمیق

محمدعلی فراهت ۹۷۵۲۱۴۲۳

سوال ۱)

در ابتدا کتابخانه yfinance را با دستور زیر نصب می‌کنیم:

```
pip install yfinance --upgrade --no-cache-dir
```

حالا داده‌های قیمت بیتکوین را در زمان‌های خواسته شده دانلود می‌کنیم:

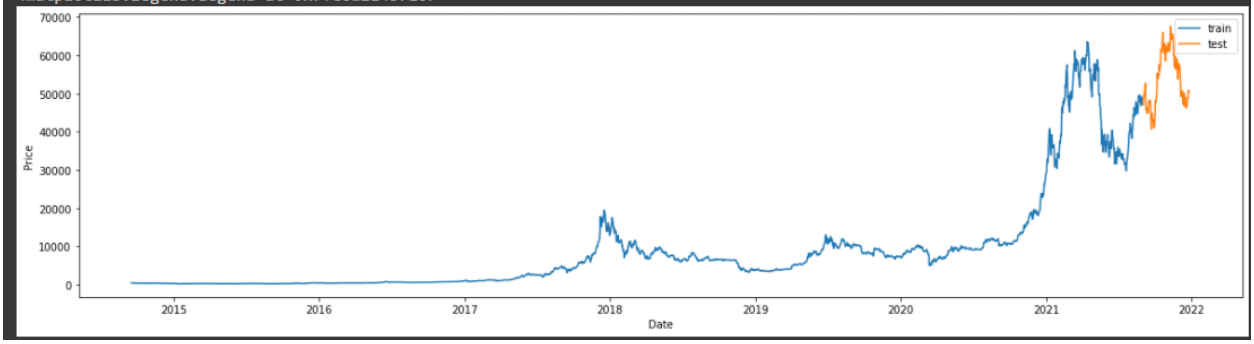
```
1 import yfinance as yf
2 data_train = yf.download("BTC-USD", start="2014-01-01", end="2021-09-01")
3 data_test = yf.download("BTC-USD", start="2021-09-01")
```

```
[**********100%**********]  1 of 1 completed
[**********100%**********]  1 of 1 completed
```

سپس هر دو داده را پشت هم رسم میکنیم که نمودار آن به صورت زیر خواهد شد:

```
1 import matplotlib.pyplot as plt
2
3 plt.figure(figsize=(20,5))
4
5 plt.xlabel('Date')
6 plt.ylabel('Price')
7
8 plt.plot(data_train['Open'], label = "train")
9 plt.plot(data_test['Open'], label = "test")
10 plt.legend()
11
```

```
<matplotlib.legend.Legend at 0x7fc6d2143710>
```



حالا داده آموزش و تست را با استفاده از تابع fit_transform در MinMaxScaler نرمالایز می‌کنیم:

```
from sklearn.preprocessing import MinMaxScaler
import numpy as np

train = data_train['Open'].values.reshape(-1, 1)
test = data_test['Open'].values.reshape(-1, 1)

scaler = MinMaxScaler()

normalized_train = np.squeeze(scaler.fit_transform(train))
normalized_test = np.squeeze(scaler.fit_transform(test))

print(normalized_train)
```

```
[0.00456166 0.00441952 0.00390242 ... 0.76811948 0.73953855 0.74072935]
```

حالا دیتا آموزش را برای ورودی دادن به مدل آماده میکنیم، برای اینکار x را برابر لیستی از ۶۰ داده پشت هم قرار میدهیم و y را برای همان داده ۶۱ امین دیتا قرار میدهیم. سپس یکی جلو میرویم و این کار را تکرار میکنیم همینطور تا آخرین داده‌ای که داریم جلو می‌رویم. در زیر نحوه انجام این کار را میبینیم:

```
sequence_train_x = []
sequence_train_y = []

for i in range(len(normalized_train) - 60):
    sequence_train_x.append(normalized_train[i:i+60])
    sequence_train_y.append(normalized_train[i+60])

sequence_train_x = np.array(sequence_train_x).reshape(2482, 60, 1)
sequence_train_y = np.array(sequence_train_y)

print(np.array(sequence_train_x).shape)
print(np.array(sequence_train_y).shape)
```

```
(2482, 60, 1)
(2482,)
```

همین کار را برای داده تست هم انجام می‌دهیم:

```
sequence_test_x = []
sequence_test_y = []

for i in range(len(normalized_test) - 60):
    sequence_test_x.append(normalized_test[i:i+60])
    sequence_test_y.append(normalized_test[i+60])

sequence_test_x = np.array(sequence_test_x).reshape(57, 60, 1)
sequence_test_y = np.array(sequence_test_y)

print(np.array(sequence_test_x).shape)
print(np.array(sequence_test_y).shape)
```

```
(57, 60, 1)
(57,)
```

بعد از آن باید مدل را بسازیم و آن را فیت کنیم (مدل از روی داک ساخته شده):

```
import tensorflow as tf
from tensorflow import keras
from keras.layers import Dense, Dropout, LSTM

model = keras.models.Sequential()

model.add(LSTM(50, input_shape=(60, 1), return_sequences=True))
model.add(Dropout(0.2))
model.add(LSTM(50, return_sequences=True))
model.add(Dropout(0.2))
model.add(LSTM(50, return_sequences=True))
model.add(Dropout(0.2))
model.add(LSTM(50))
model.add(Dense(1))

loss = "mean_squared_error"
opt = "adam"
model.compile(loss=loss, optimizer=opt)
h = model.fit(sequence_train_x, sequence_train_y, epochs=100, batch_size=32)
```

مدل در ۱۰۰ ایپاک آموزش دید و تمام شد:

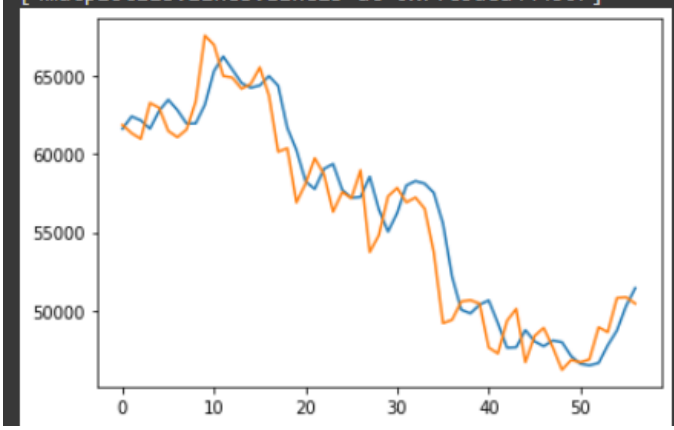
```
Epoch 98/100
78/78 [==============================] - 5s 61ms/step - loss: 2.4856e-04
Epoch 99/100
78/78 [==============================] - 5s 61ms/step - loss: 2.5120e-04
Epoch 100/100
78/78 [==============================] - 5s 62ms/step - loss: 2.4041e-04
```

سپس مدل را با داده آزمایشی تست میکنیم و نمودار آن را رسم میکنیم:

```
predict = model.predict(sequence_test_x)
real_predict = scaler.inverse_transform(predict)
real_label = scaler.inverse_transform([sequence_test_y])

plt.plot(real_predict)
plt.plot(np.squeeze(real_label))
```

```
[<matplotlib.lines.Line2D at 0x7fc5dea44450>]
```



بعد از آن قیمت بیتکوین را برای ۳ ماه بعد از آخرین تاریخ دیتا پیش‌بینی می‌کنیم:

```
last_60_days = list(normalized_test[-60:])
predict = []

for i in range(90):
    pred = model.predict(np.expand_dims(last_60_days, 0))
    predict.append(pred[0][0])
    last_60_days.append(pred[0][0])
    last_60_days.remove(last_60_days[0])

plt.plot(np.squeeze(scaler.inverse_transform(np.array(predict).reshape(-1, 1))))

```

```
[<matplotlib.lines.Line2D at 0x7fc5b70bc450>]
```