01.stockprediction.rev.01

August 14, 2024

[ ]:

```
import pandas as pd
import matplotlib.pyplot as plt
import numpy as np
import math
from sklearn.preprocessing import MinMaxScaler
from sklearn.metrics import mean_squared_error
from keras.models import Sequential
from keras.layers import Dense, Activation
from keras.layers import LSTM
import preprocessing
```

[ ]:

```
np.random.seed(7)
```

[ ]:

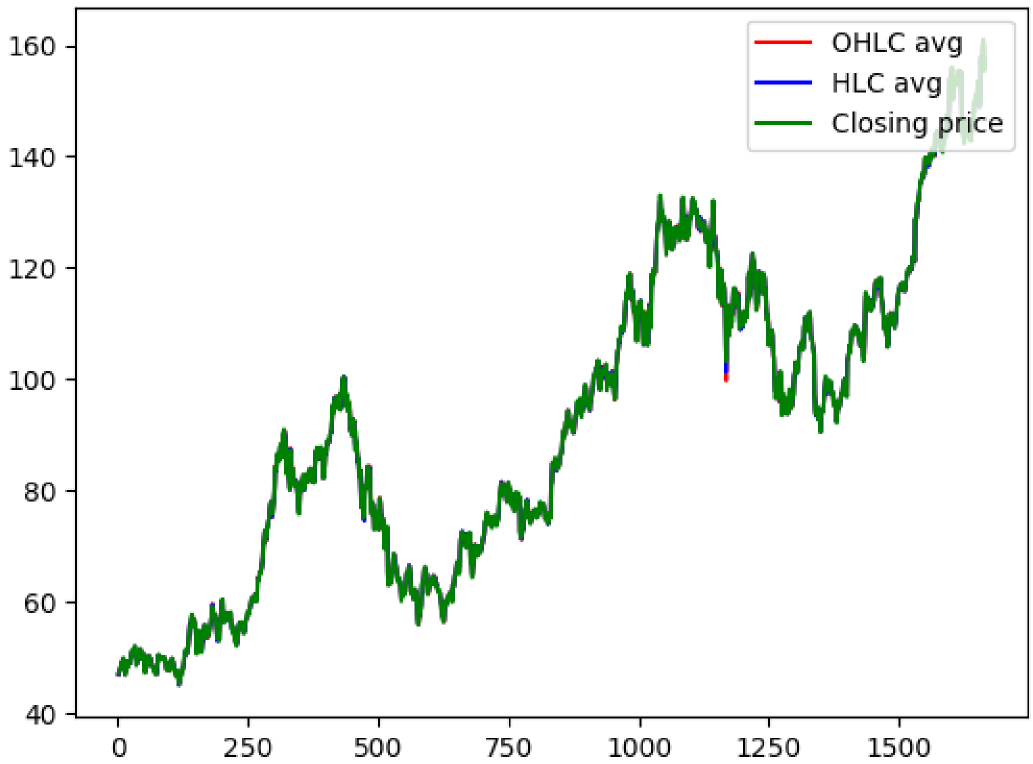
```
dataset = pd.read_csv('apple_share_price.csv', usecols=[1,2,3,4])
dataset = dataset.reindex(index = dataset.index[::-1])
```

[ ]:

```
obs = np.arange(1, len(dataset) + 1, 1)

OHLC_avg = dataset.mean(axis = 1)
HLC_avg = dataset[['High', 'Low', 'Close']].mean(axis = 1)
close_val = dataset[['Close']]
```

[ ]:

```
plt.plot(obs, OHLC_avg, 'r', label = 'OHLC avg')
plt.plot(obs, HLC_avg, 'b', label = 'HLC avg')
plt.plot(obs, close_val, 'g', label = 'Closing price')
plt.legend(loc = 'upper right')
plt.show()
```



[ ]:

```
OHLC_avg = np.reshape(OHLC_avg.values, (len(OHLC_avg),1)) # 1664
scaler = MinMaxScaler(feature_range=(0, 1))
OHLC_avg = scaler.fit_transform(OHLC_avg)
```

[ ]:

```
train_OHLC = int(len(OHLC_avg) * 0.75)
test_OHLC = len(OHLC_avg) - train_OHLC
train_OHLC, test_OHLC = OHLC_avg[0:train_OHLC,:], OHLC_avg[train_OHLC:
 ↪len(OHLC_avg),:]
```

[ ]:

```
trainX, trainY = preprocessing.new_dataset(train_OHLC, 1)
testX, testY = preprocessing.new_dataset(test_OHLC, 1)
```

[ ]:

```
trainX = np.reshape(trainX, (trainX.shape[0], 1, trainX.shape[1]))
testX = np.reshape(testX, (testX.shape[0], 1, testX.shape[1]))
step_size = 1
```

[ ]:

```
model = Sequential()
model.add(LSTM(32, input_shape=(1, step_size), return_sequences = True))
model.add(LSTM(16))
model.add(Dense(1))
model.add(Activation('linear'))
```

[ ]:

```python
model.compile(loss='mean_squared_error', optimizer='adagrad') # Try SGD, adam, adagrad and compare!!!
model.fit(trainX, trainY, epochs=20, batch_size=1, verbose=1)
```

```
Epoch 1/20
1246/1246                5s 2ms/step -
loss: 0.1274
Epoch 2/20
1246/1246                2s 2ms/step -
loss: 0.0821
Epoch 3/20
1246/1246                2s 2ms/step -
loss: 0.0677
Epoch 4/20
1246/1246                2s 2ms/step -
loss: 0.0593
Epoch 5/20
1246/1246                2s 2ms/step -
loss: 0.0520
Epoch 6/20
1246/1246                2s 2ms/step -
loss: 0.0505
Epoch 7/20
1246/1246                2s 2ms/step -
loss: 0.0422
Epoch 8/20
1246/1246                2s 1ms/step -
loss: 0.0450
Epoch 9/20
1246/1246                2s 2ms/step -
loss: 0.0403
Epoch 10/20
1246/1246                3s 2ms/step -
loss: 0.0413
Epoch 11/20
1246/1246                3s 2ms/step -
loss: 0.0381
Epoch 12/20
1246/1246                2s 2ms/step -
loss: 0.0369
Epoch 13/20
1246/1246                3s 2ms/step -
loss: 0.0402
Epoch 14/20
1246/1246                2s 2ms/step -
loss: 0.0389
Epoch 15/20
1246/1246                2s 2ms/step -
```

```
loss: 0.0382
Epoch 16/20
1246/1246           3s 2ms/step -
loss: 0.0370
Epoch 17/20
1246/1246           3s 2ms/step -
loss: 0.0356
Epoch 18/20
1246/1246           2s 2ms/step -
loss: 0.0356
Epoch 19/20
1246/1246           2s 2ms/step -
loss: 0.0352
Epoch 20/20
1246/1246           2s 2ms/step -
loss: 0.0354
```

```
[ ]: <keras.src.callbacks.history.History at 0x1fb8774e2a0>
```

[ ]:

```python
trainPredict = model.predict(trainX)
testPredict = model.predict(testX)

trainPredict = scaler.inverse_transform(trainPredict)
trainY = scaler.inverse_transform([trainY])
testPredict = scaler.inverse_transform(testPredict)
testY = scaler.inverse_transform([testY])
```

```
39/39           1s 8ms/step
13/13           0s 5ms/step
```

[ ]:

```python

trainScore = math.sqrt(mean_squared_error(trainY[0], trainPredict[:,0]))
print('Train RMSE: %.2f' % (trainScore))

testScore = math.sqrt(mean_squared_error(testY[0], testPredict[:,0]))
print('Test RMSE: %.2f' % (testScore))

trainPredictPlot = np.empty_like(OHLC_avg)
trainPredictPlot[:, :] = np.nan
trainPredictPlot[step_size:len(trainPredict)+step_size, :] = trainPredict

testPredictPlot = np.empty_like(OHLC_avg)
testPredictPlot[:, :] = np.nan
testPredictPlot[len(trainPredict)+(step_size*2)+1:len(OHLC_avg)-1, :] =␣
 ↪testPredict

OHLC_avg = scaler.inverse_transform(OHLC_avg)
```

```
Train RMSE: 21.69
```

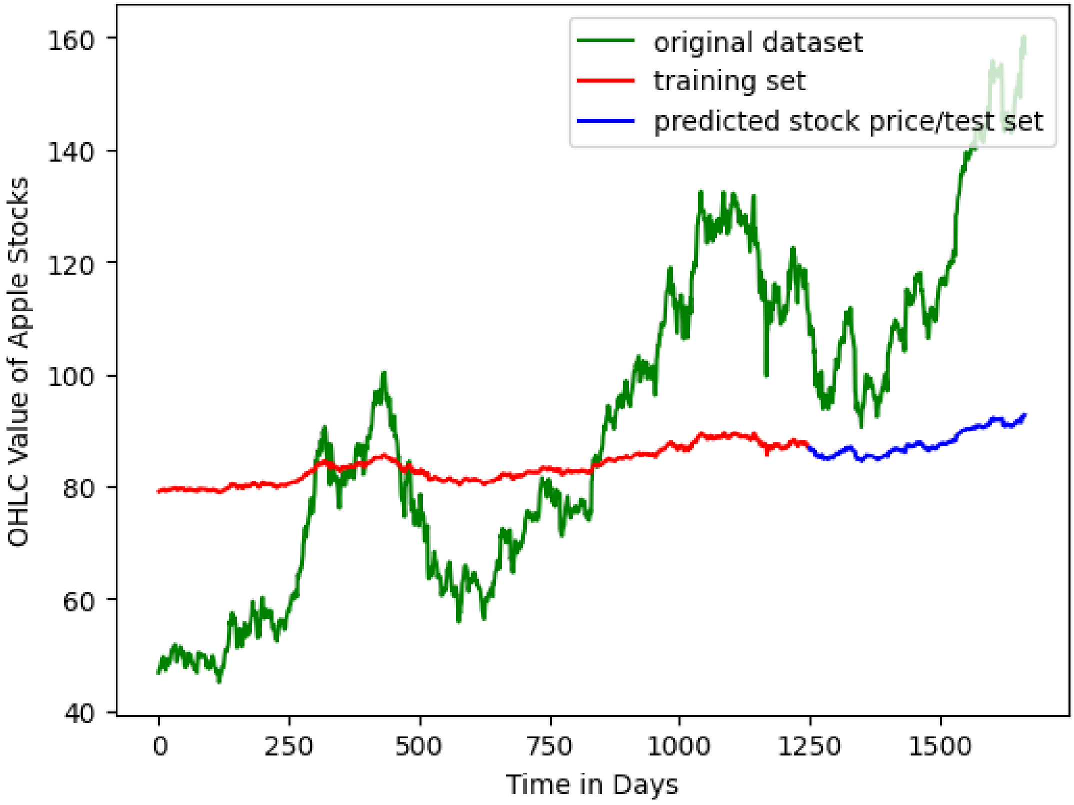
```
Test RMSE: 35.35
```

[ ]:

```python
import numpy

def patch_asscalar(a):
    return a.item()

setattr(numpy, "asscalar", patch_asscalar)
```

[ ]:

```python
plt.plot(OHLC_avg, 'g', label = 'original dataset')
plt.plot(trainPredictPlot, 'r', label = 'training set')
plt.plot(testPredictPlot, 'b', label = 'predicted stock price/test set')
plt.legend(loc = 'upper right')
plt.xlabel('Time in Days')
plt.ylabel('OHLC Value of Apple Stocks')
plt.show()


last_val = testPredict[-1]
last_val_scaled = last_val/last_val
next_val = model.predict(np.reshape(last_val_scaled, (1,1,1)))
print("Last Day Value:", np.asscalar(last_val))
print("Next Day Value:", np.asscalar(last_val*next_val))
```



```
1/1             0s 29ms/step
Last Day Value: 92.63001251220703
Next Day Value: 38.290130615234375
1/1             0s 29ms/step
Last Day Value: 92.63001251220703
Next Day Value: 38.290130615234375
```