

Delivery of Sprint-2

Date	31 October 2022
Team ID	PNT2022TMID21805
Project Name	Crude Oil Price Prediction

MODEL BUILDING

Importing The Model Building Libraries

```
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense
from tensorflow.keras.layers import LSTM
```

Initializing The Model

```
model = Sequential()
```

Adding LSTM Layers

```
model.add(LSTM(50, return_sequences = True, input_shape = (10,1)))
model.add(LSTM(50, return_sequences = True))
model.add(LSTM(50))
```

Adding Output Layers

```
model.add(Dense(1))
model.summary()
```

Model: "sequential"

Layer (type)	Output Shape	Param #
=====		
lstm (LSTM)	(None, 10, 50)	10400
lstm_1 (LSTM)	(None, 10, 50)	20200
lstm_2 (LSTM)	(None, 50)	20200
dense (Dense)	(None, 1)	51
=====		
Total params: 50,851		
Trainable params: 50,851		
Non-trainable params: 0		

```
Configure The Learning Process model.compile(loss='mean_squared_error',  
optimizer = 'adam')
```

Train The Model

```
model.fit(X_train, y_train, validation_data = (X_test, ytest), epochs  
= 10, batch_size = 64, verbose = 1)  
Epoch 1/10  
84/84 [=====] - 8s 33ms/step - loss: 0.0019 -  
val_loss: 9.9616e-04  
Epoch 2/10  
84/84 [=====] - 1s 17ms/step - loss: 1.2329e-  
04 - val_loss: 7.3913e-04  
Epoch 3/10  
84/84 [=====] - 1s 17ms/step - loss:  
1.2148e04 - val_loss: 0.0014  
Epoch 4/10  
84/84 [=====] - 1s 18ms/step - loss: 1.2151e-  
04 - val_loss: 7.6063e-04  
Epoch 5/10  
84/84 [=====] - 1s 17ms/step - loss:  
1.2602e04 - val_loss: 0.0020  
Epoch 6/10  
84/84 [=====] - 1s 17ms/step - loss:  
1.2412e04 - val_loss: 0.0011  
Epoch 7/10  
84/84 [=====] - 1s 17ms/step - loss: 1.1884e-  
04 - val_loss: 7.1855e-04  
Epoch 8/10  
84/84 [=====] - 1s 17ms/step - loss: 1.1737e-  
04 - val_loss: 7.6043e-04  
Epoch 9/10  
84/84 [=====] - 1s 17ms/step - loss: 1.1241e-  
04 - val_loss: 9.7294e-04  
Epoch 10/10  
84/84 [=====] - 1s 17ms/step - loss: 1.1236e-  
04 - val_loss: 6.5660e-04  
<keras.callbacks.History at 0x2505dbb7970>
```

Model Evaluation

```
train_predict=model.predict(X_train)  
test_predict=model.predict(X_test)
```

```
167/167 [=====] - 2s 3ms/step
90/90 [=====] - 0s 3ms/step
```

```
train_predict = scaler.inverse_transform(train_predict)
test_predict = scaler.inverse_transform(test_predict)
import math from sklearn.metrics import
mean_squared_error
math.sqrt(mean_squared_error(y_train,train_predict))
28.851078372476536
```

Save The Model

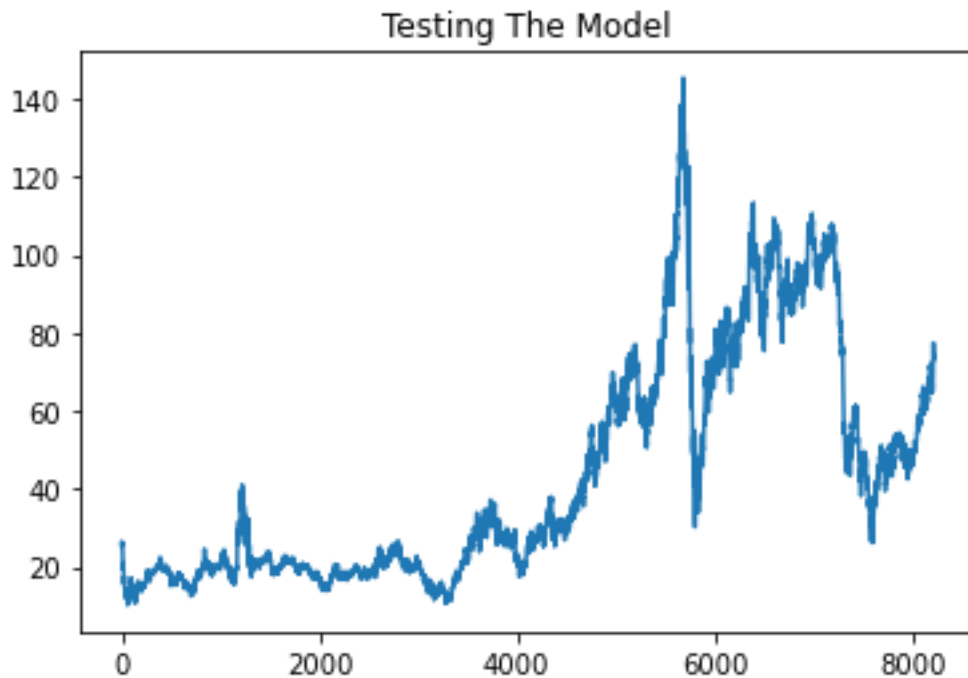
```
from tensorflow.keras.models import load_model
model.save("Crude_oil.h5")
```

Test The Model

```
look_back = 0
trainPredictPlot = np.empty_like(data_oil) trainPredictPlot[:,
:] = np.nan
trainPredictPlot[look_back:len(train_predict) + look_back, :] =
train_predict

testPredictPlot = np.empty_like(data_oil) testPredictPlot[:,:]
= np.nan
testPredictPlot[len(train_predict)+(look_back*2)+1: len(data_oil)-1,
:] = test_predict

plt.plot(scaler.inverse_transform(data_oil))
plt.plot(trainPredictPlot)
plt.plot(testPredictPlot) plt.title("Testing
The Model") plt.show()
```



```
len(test_data)
2876
```

```
x_input = test_data[2866:].reshape(1,-1) x_input.shape
(1, 10)
```

```
temp_input = list(x_input)
temp_input = temp_input[0].tolist()
temp_input [0.44172960165852215,
0.48111950244335855,
0.49726047682511476,
0.4679401747371539,
0.4729749740855915,
0.47119798608026064,
0.47341922108692425,
0.4649785280616022,
0.4703835332444839,
0.47149415074781587]
```

```

lst_output = [] n_steps = 10 i=0 while(i<10):
if(len(temp_input)>10):          x_input =
np.array(temp_input[1:])          print("{} day
input {}".format(i,x_input))          x_input =
x_input.reshape(1,-1)          x_input =
x_input.reshape((1,n_steps, 1))

        yhat = model.predict(x_input, verbose = 0)
print("{} day output {}".format(i,yhat))
temp_input.extend(yhat[0].tolist())
temp_input = temp_input[1:]
lst_output.extend(yhat.tolist())          i=i+1
else:          x_input = x_input.reshape((1,
n_steps,1))          yhat = model.predict(x_input,
verbose = 0)          print(yhat[0])
        temp_input.extend(yhat[0].tolist())
print(len(temp_input))
lst_output.extend(yhat.tolist())          i=i+1
[0.47125974]
11
1 day input [0.4811195  0.49726048 0.46794017 0.47297497 0.47119799
0.47341922
0.46497853 0.47038353 0.47149415 0.47125974]
1 day output [[0.47265336]]
2 day input [0.49726048 0.46794017 0.47297497 0.47119799 0.47341922
0.46497853
0.47038353 0.47149415 0.47125974 0.47265336]
2 day output [[0.4715367]]
3 day input [0.46794017 0.47297497 0.47119799 0.47341922 0.46497853
0.47038353
0.47149415 0.47125974 0.47265336 0.4715367 ]
3 day output [[0.46978694]]
4 day input [0.47297497 0.47119799 0.47341922 0.46497853 0.47038353
0.47149415
0.47125974 0.47265336 0.4715367 0.46978694]
4 day output [[0.4700314]]
5 day input [0.47119799 0.47341922 0.46497853 0.47038353 0.47149415
0.47125974
0.47265336 0.4715367 0.46978694 0.47003141]
5 day output [[0.4699089]]
6 day input [0.47341922 0.46497853 0.47038353 0.47149415 0.47125974
0.47265336

```

```

0.4715367 0.46978694 0.47003141 0.46990889]
6 day output [[0.46986535]]
7 day input [0.46497853 0.47038353 0.47149415 0.47125974 0.47265336
0.4715367
0.46978694 0.47003141 0.46990889 0.46986535]
7 day output [[0.46965963]]
8 day input [0.47038353 0.47149415 0.47125974 0.47265336 0.4715367
0.46978694
0.47003141 0.46990889 0.46986535 0.46965963]
8 day output [[0.4699126]]
9 day input [0.47149415 0.47125974 0.47265336 0.4715367 0.46978694
0.47003141
0.46990889 0.46986535 0.46965963 0.46991259]
9 day output [[0.46976325]]

```

```

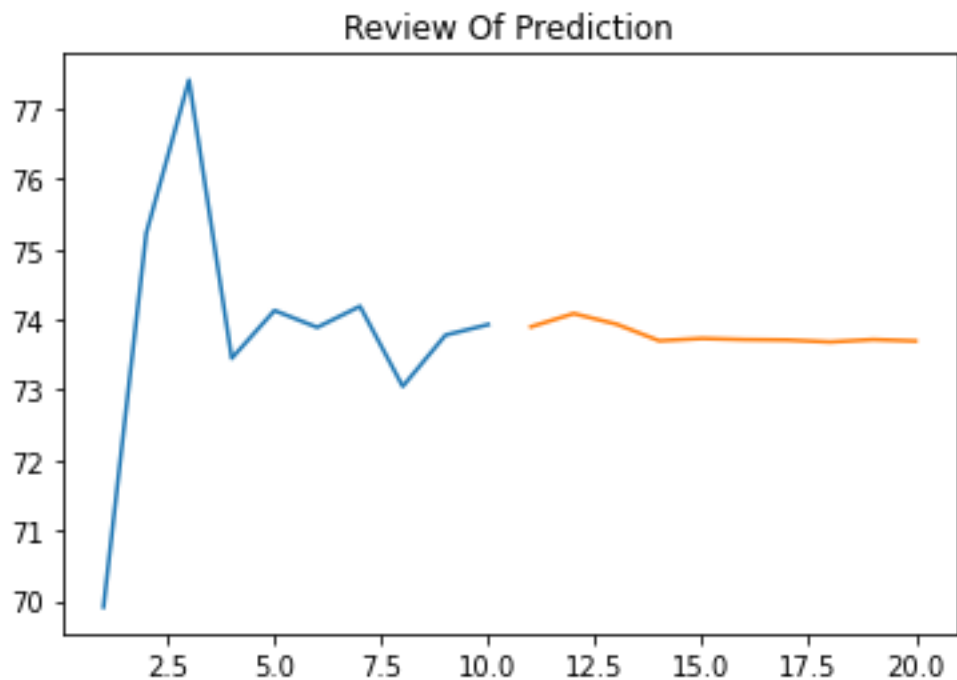
day_new = np.arange(1,11) day_pred
= np.arange(11,21) len(data_oil)
8216

```

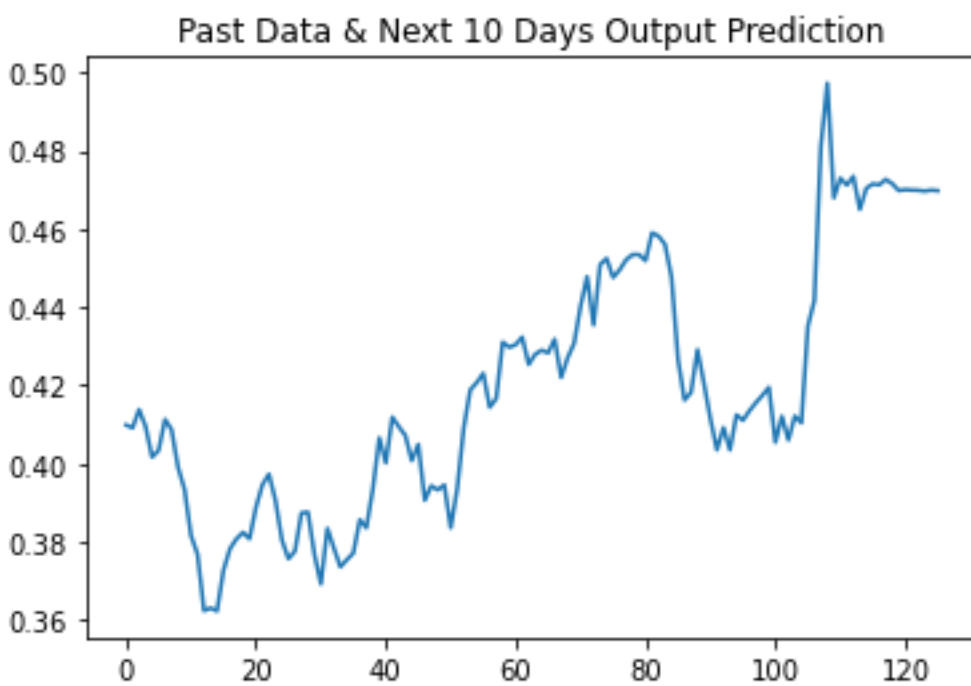
```

plt.plot(day_new,scaler.inverse_transform(data_oil[8206:]))
plt.title("Review Of Prediction")
plt.plot(day_pred,scaler.inverse_transform(1st_output)) plt.show()

```



```
df3 = data_oil.tolist() df3.extend(1st_output)
plt.title("Past Data & Next 10 Days Output Prediction")
plt.plot(df3[8100:])
[<matplotlib.lines.Line2D at 0x250696187c0>]
```



```
df3 = scaler.inverse_transform(df3).tolist()
plt.title("Past Data & Next 10 Days Output Prediction After Reversing
The Scaled Values") plt.plot(df3)
[<matplotlib.lines.Line2D at 0x25069758a30>]
```

Past Data & Next 10 Days Output Prediction After Reversing The Scaled Values

