MODEL BUILDING

Team ID	PNT2022TMID06799
Project Name	Crude Oil Price Prediction

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=50,batch_size=64,verbose=1)
```

```
Epoch 1/50
84/84 [==============================] - 10s 40ms/step - loss: 0.0018 - val_loss: 7.8491e-04
Epoch 2/50
84/84 [==============================] - 2s 23ms/step - loss: 1.2458e-04 - val_loss: 8.2808e-04
Epoch 3/50
84/84 [==============================] - 2s 26ms/step - loss: 1.1971e-04 - val_loss: 7.3377e-04
Epoch 4/50
84/84 [==============================] - 2s 22ms/step - loss: 1.2129e-04 - val_loss: 8.7799e-04
Epoch 5/50
84/84 [==============================] - 2s 24ms/step - loss: 1.1870e-04 - val_loss: 8.8529e-04
Epoch 6/50
84/84 [==============================] - 2s 26ms/step - loss: 1.2588e-04 - val_loss: 0.0010
Epoch 7/50
84/84 [==============================] - 2s 24ms/step - loss: 1.2309e-04 - val_loss: 0.0011
Epoch 8/50
84/84 [==============================] - 2s 21ms/step - loss: 1.1149e-04 - val_loss: 8.1880e-04
Epoch 9/50
84/84 [==============================] - 2s 19ms/step - loss: 1.1051e-04 - val_loss: 7.3024e-04
Epoch 10/50
84/84 [==============================] - 2s 19ms/step - loss: 1.0838e-04 - val_loss: 6.5473e-04
Epoch 11/50
84/84 [==============================] - 2s 19ms/step - loss: 1.2032e-04 - val_loss: 0.0016
Epoch 12/50
84/84 [==============================] - 2s 19ms/step - loss: 1.1879e-04 - val_loss: 6.1331e-04
Epoch 13/50
84/84 [==============================] - 2s 19ms/step - loss: 1.0408e-04 - val_loss: 5.8768e-04
Epoch 14/50
84/84 [==============================] - 2s 20ms/step - loss: 9.5711e-05 - val_loss: 5.9183e-04
Epoch 15/50
84/84 [==============================] - 2s 20ms/step - loss: 9.1507e-05 - val_loss: 5.4513e-04
Epoch 16/50

Epoch 17/50
84/84 [==============================] - 2s 20ms/step - loss: 9.1008e-05 - val_loss: 7.0263e-04
Epoch 18/50
84/84 [==============================] - 2s 21ms/step - loss: 8.8793e-05 - val_loss: 4.3665e-04
Epoch 19/50
84/84 [==============================] - 2s 23ms/step - loss: 9.0948e-05 - val_loss: 6.0063e-04
Epoch 20/50
84/84 [==============================] - 2s 22ms/step - loss: 8.7479e-05 - val_loss: 6.1548e-04
Epoch 21/50
84/84 [==============================] - 2s 21ms/step - loss: 9.1391e-05 - val_loss: 0.0010
Epoch 22/50
84/84 [==============================] - 2s 21ms/step - loss: 7.4975e-05 - val_loss: 4.2512e-04
Epoch 23/50
84/84 [==============================] - 2s 21ms/step - loss: 9.9814e-05 - val_loss: 7.6774e-04
Epoch 24/50
84/84 [==============================] - 2s 22ms/step - loss: 7.7256e-05 - val_loss: 9.2345e-04
Epoch 25/50
84/84 [==============================] - 2s 21ms/step - loss: 7.0041e-05 - val_loss: 3.3666e-04
Epoch 26/50
84/84 [==============================] - 2s 24ms/step - loss: 6.2571e-05 - val_loss: 3.3346e-04
Epoch 27/50
84/84 [==============================] - 2s 28ms/step - loss: 6.2461e-05 - val_loss: 2.9808e-04
Epoch 28/50
84/84 [==============================] - 2s 24ms/step - loss: 6.3408e-05 - val_loss: 3.0581e-04
Epoch 29/50
84/84 [==============================] - 2s 23ms/step - loss: 6.3040e-05 - val_loss: 2.6675e-04
Epoch 30/50
84/84 [==============================] - 2s 23ms/step - loss: 5.8238e-05 - val_loss: 2.6790e-04
Epoch 31/50
84/84 [==============================] - 2s 22ms/step - loss: 5.2839e-05 - val_loss: 3.2383e-04
Epoch 32/50
84/84 [==============================] - 2s 22ms/step - loss: 5.5354e-05 - val_loss: 2.4147e-04
Epoch 33/50
84/84 [==============================] - 2s 26ms/step - loss: 5.0568e-05 - val_loss: 2.5649e-04
Epoch 34/50
84/84 [==============================] - 2s 24ms/step - loss: 4.8728e-05 - val_loss: 2.1842e-04
Epoch 35/50
84/84 [==============================] - 2s 23ms/step - loss: 4.6476e-05 - val_loss: 5.8689e-04
Epoch 36/50
84/84 [==============================] - 2s 21ms/step - loss: 4.6695e-05 - val_loss: 4.8287e-04
Epoch 37/50
```

```
Epoch 38/50
84/84 [==============================] - 2s 20ms/step - loss: 4.2573e-05 - val_loss: 2.0293e-04
Epoch 39/50
84/84 [==============================] - 2s 19ms/step - loss: 4.4277e-05 - val_loss: 2.0208e-04
Epoch 40/50
84/84 [==============================] - 2s 19ms/step - loss: 4.1741e-05 - val_loss: 2.6171e-04
Epoch 41/50
84/84 [==============================] - 2s 21ms/step - loss: 4.2183e-05 - val_loss: 1.8483e-04
Epoch 42/50
84/84 [==============================] - 2s 21ms/step - loss: 3.6736e-05 - val_loss: 2.1604e-04
Epoch 43/50
84/84 [==============================] - 2s 26ms/step - loss: 3.5417e-05 - val_loss: 1.9743e-04
Epoch 44/50
84/84 [==============================] - 2s 28ms/step - loss: 3.7554e-05 - val_loss: 1.8170e-04
Epoch 45/50
84/84 [==============================] - 2s 22ms/step - loss: 3.7558e-05 - val_loss: 1.8212e-04
Epoch 46/50
84/84 [==============================] - 2s 21ms/step - loss: 3.6515e-05 - val_loss: 7.0041e-04
Epoch 47/50
84/84 [==============================] - 2s 21ms/step - loss: 3.5327e-05 - val_loss: 4.7348e-04
Epoch 48/50
84/84 [==============================] - 2s 20ms/step - loss: 3.4312e-05 - val_loss: 2.0132e-04
Epoch 49/50
84/84 [==============================] - 2s 20ms/step - loss: 3.4950e-05 - val_loss: 2.2466e-04
Epoch 50/50
84/84 [==============================] - 2s 21ms/step - loss: 3.2850e-05 - val_loss: 1.8159e-04
```

]:

Model Evaluation

```
train_predict = model.predict(x_train)
test_predict = model.predict(x_test)
```

```
167/167 [==============================] - 2s 5ms/step
90/90 [==============================] - 1s 6ms/step
```

```
# model evaluation
train_predict=scaler.inverse_transform(train_predict)
test_predict=scaler.inverse_transform(test_predict)
```

```
#save the model
import math
from sklearn.metrics import mean_squared_error
math.sqrt(mean_squared_error(y_train,train_predict))
```

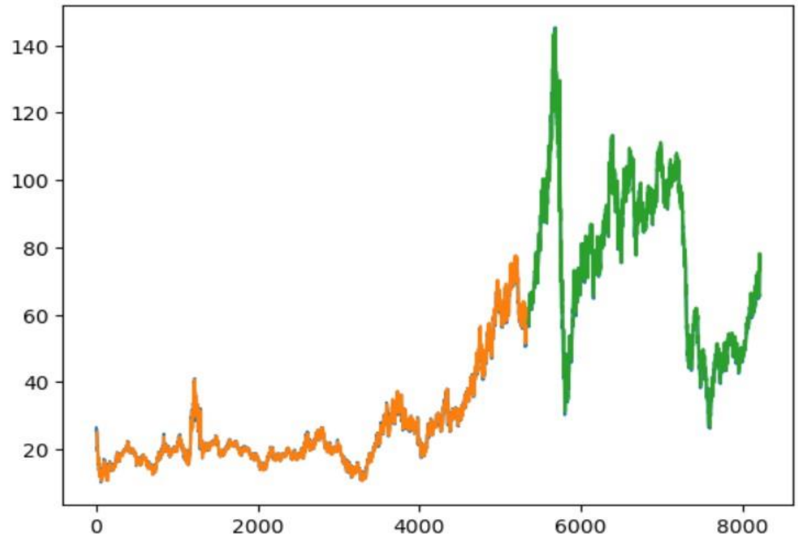
```
29.40603971934792
```

Save The Model

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

Test The Model

```
#test the data
look_back=10
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.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
```

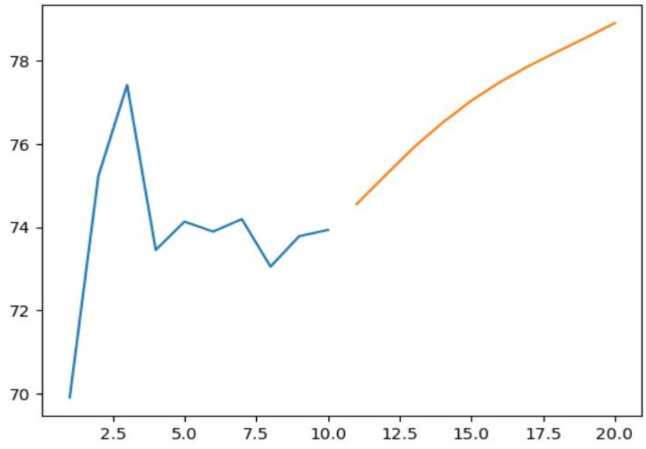
```
[0.47607496]
11
1 day input [0.4811195  0.49726048 0.46794017 0.47297497 0.47119799 0.47341922
 0.46497853 0.47038353 0.47149415 0.47607496]
1 day output [[0.48119003]]
2 day input [0.49726048 0.46794017 0.47297497 0.47119799 0.47341922 0.46497853
 0.47038353 0.47149415 0.47607496 0.48119003]
2 day output [[0.4861873]]
3 day input [0.46794017 0.47297497 0.47119799 0.47341922 0.46497853 0.47038353
 0.47149415 0.47607496 0.48119003 0.48618731]
3 day output [[0.49056533]]
4 day input [0.47297497 0.47119799 0.47341922 0.46497853 0.47038353 0.47149415
 0.47607496 0.48119003 0.48618731 0.49056533]
4 day output [[0.49446633]]
5 day input [0.47119799 0.47341922 0.46497853 0.47038353 0.47149415 0.47607496

 0.48119003 0.48618731 0.49056533 0.49446633]
5 day output [[0.49777645]]
6 day input [0.47341922 0.46497853 0.47038353 0.47149415 0.47607496 0.48119003
 0.48618731 0.49056533 0.49446633 0.49777645]
6 day output [[0.5006322]]
7 day input [0.46497853 0.47038353 0.47149415 0.47607496 0.48119003 0.48618731
 0.49056533 0.49446633 0.49777645 0.50063223]
7 day output [[0.50317526]]
8 day input [0.47038353 0.47149415 0.47607496 0.48119003 0.48618731 0.49056533
 0.49446633 0.49777645 0.50063223 0.50317526]
8 day output [[0.5056825]]
9 day input [0.47149415 0.47607496 0.48119003 0.48618731 0.49056533 0.49446633
 0.49777645 0.50063223 0.50317526 0.50568253]
9 day output [[0.50824463]]
```

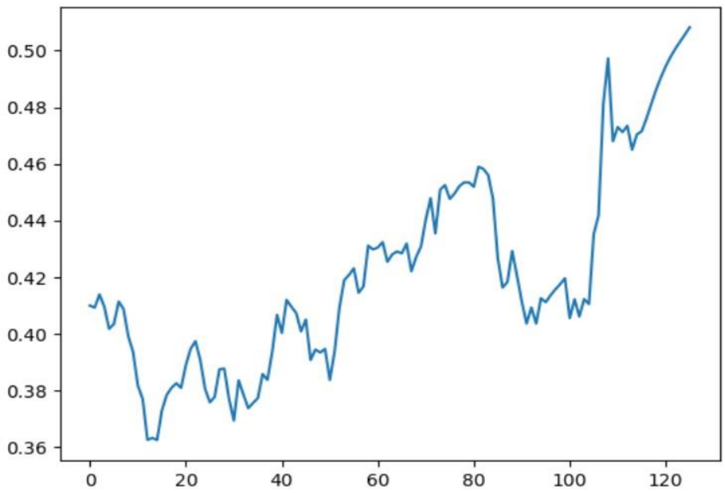
```
day_new=np.arange(1,11)
day_pred=np.arange(11,21)
len(data_oil)
```

```
8216
```

```
plt.plot(day_new,scaler.inverse_transform(data_oil[8206:]))
plt.plot(day_pred,scaler.inverse_transform(lst_output))
```




```
df3=data_oil.tolist()
df3.extend(lst_output)
plt.plot(df3[8100:])
```




```
df3=scaler.inverse_transform(df3).tolist()
```

```
plt.plot(df3)
```