# PROFESSIONAL TRAINING REPORT

**at**

**Sathyabama Institute of Science and Technology (Deemed to be University)**

Submitted in partial fulfillment of the requirements for the award of Bachelor of Engineering Degree in Computer Science and Engineering

By

**G.K.N.S.S.SHANKAR (Reg.No. 40110368)**



**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING SCHOOL OF COMPUTING**

**SATHYABAMA INSTITUTE OF SCIENCE AND TECHNOLOGY**

**JEPPIAAR NAGAR, RAJIV GANDHI SALAI, CHENNAI – 600119, TAMILNADU**

**APR 2023**

SATHYABAMA

**INSTITUTE OF SCIENCE AND TECHNOLOGY**

(DEEMED TO BE UNIVERSITY)

## Accredited with Grade “A” by NAAC

**(Established under Section 3 of UGC Act, 1956)**

**JEPPIAAR NAGAR, RAJIV GANDHI SALAI, CHENNAI– 600119**

[**www.sathyabama .ac.in**](http://www.sathyabama.ac.in/)

## DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

**BONAFIDE CERTIFICATE**

This is to certify that this Project Report is the bonafide work of who carried out the project entitled **“Crude Oil Price Prediction Using IBM Watson Studio”** under my supervision from March 2023 to .April 2023

**Internal Guide**

**Dr. Veena**

## Head of the Department

**Dr.L.LAKSHMANAN M.E., Ph.D.**



**Submitted for Viva voce Examination held on**

**Internal Examiner External Examiner**

## DECLARATION

I**, G K N S S Shankar** (Reg No: **40110368**) hereby declare that the Project Reportentitled **“Crude Oil Price Prediction Using IBM Watson Studio”** Done by me under the guidance of **Dr.Veena** and **Mahidhar Reddy** at **Smart Bridge(Smart Internz)** is submitted in Partial fulfilment of the requirments for the award of Bachelore of Engineering degree In Computer Science and Engineering

**DATE:**

**PLACE: CHENNAI SIGNATURE OF THE CANDIDATE**

**ACKNOWLEDGEMENT**

I am pleased to acknowledge my sincere thanks to **Board of Management** of **SATHYABAMA** for their kind encouragement in doing this project and for completing it successfully. I am grateful to them.

I convey my thanks to **Dr. T. Sasikala M.E., Ph.D.**, **Dean**, School of Computing, **Dr.L.Lakshmanan M.E., Ph.D.,** Head of the Department of Computer Science and Engineering for providing me necessary support and details at the right time during the progressive reviews.

I would like to express my sincere and deep sense of gratitude to my Project Guide **Dr.Veena,** for her valuable guidance, suggestions and constant encouragement paved the way for the successful completion of my projectwork.

I wish to express my thanks to all Teaching and Non-teaching staff members of the **Department of Computer Science and Engineering** who were helpful in many ways for the completion of the project.

**TRAINING CERTIFICATE**

**TABLE OF CONTENTS**

|  |  |  |
| --- | --- | --- |
| **CHAPTER NO.** | **TITLE** | **PAGE NO.** |
|  | ABSTRACT | vii |
|  | LIST OF FIGURES | ix |
| **1** | **INTRODUCTION** | 1 |
| **2** | **AIM AND SCOPE OF PRESENT INVESTIGATION** | 2 |
|  | 2.1 Aim | 2 |
|  | 2.2 Scope | 2 |
| **3** | **EXPERIMENTAL OR MATERIALS AND METHODS; ALGORITHMS USED** | 3 |
|  | 3.1 Pre Requisites | 3 |
|  | 3.2 Data Collection | 3 |
|  | *3.2.1 Download Dataset* | 3 |
|  | 3.3 Data Preprocessing | 4 |
|  | *3.3.1 Import The Libraries* | 4 |
| **4** | **RESULTS AND DISCUSSION, PERFORMANCE ANALYSIS** | 5 |
|  | 4.1 Importing The Dataset | 5 |
|  | 4.2 Analyze The Data | 5 |
|  | *4.2.1 Handling Missing Data* | 7 |
|  | *4.2.2 Feature Scaling* | 8 |
|  | *4.2.3 Data Visualization* | 8 |
|  | *4.2.4 Splitting The Dataset Into Train And Test* | 9 |
|  | 4.3 Creating A Dataset With Sliding Windows | 10 |
| **CHAPTER NO.** | **TITLE** | **PAGE NO.** |
| **5** | **MODEL BUILDING** | 12 |
|  | 5.1 Importing The Model Building Libraries | 12 |
|  | 5.2 Initialising The Model | 12 |
|  | 5.3 Adding Lstm Layers | 12 |
|  | 5.4 Adding Output Layers | 13 |
|  | 5.5 Configure The Learning Process | 13 |
|  | 5.6 Train The Model | 14 |
|  | 5.7 Model Evaluation | 14 |
|  | 5.8 Save The Model | 15 |
|  | 5.9 Test The Model | 15 |
| **6** | **APPLICATION BUILDING** | 21 |
|  | 6.1 Create An HTML File | 21 |
|  | 6.2 Build Python Code | 21 |
|  | 6.3 Run The App In Local Browser | 23 |
|  | 6.4 Showcasing Prediction On UI | 24 |
| **7** | **REFERENCES** | 26 |
| **8** | **SOURCE CODE** | 27 |

**ABSTRACT**

Prediction of future crude oil price is considered a significant challenge due to the extremely complex, chaotic, and dynamic nature of the market and stakeholder’s perception. The crude oil price changes every minute, and millions of shares ownerships are traded every day. The market price for commodity such as crude oil is influenced by many factors including news, supply-and-demand gap, labor costs, amount of remaining resources, as well as stakeholders’ perception. Therefore, various indicators for technical analysis have been utilized for the purpose of predicting the future crude oil price. Recently, many researchers have turned to machine learning approached to cater to this problem. This study demonstrated the use of RNN-LSTM networks for predicting the crude oil price based on historical data alongside other technical analysis indicators. This study aims to certify the capability of a prediction model built based on the RNN-LSTM network to predict the future price of crude oil. The developed model is trained and evaluated against accuracy matrices to assess the capability of the network to provide an improvement of the accuracy of crude oil price prediction as compared to other strategies. The result obtained from the model shows a promising prediction capability of the RNN-LSTM algorithm for predicting crude oil price movement.

***Keywords*** – RNN-LSTM, Chaotic, Stake Holder’s

**LIST OF FIGURES**

|  |  |  |
| --- | --- | --- |
| **FIGURE NO.** | **FIGURE NAME** | **PAGE NO** |
| 4.1 | Plotting Values | 9 |
| 5.1 | Testing The Model | 16 |
| 5.2 | Scalar Inverse Form Plotting | 19 |
| 5.3 | Ten Days Output Prediction | 19 |
| 5.4 | Inversing The Values | 20 |
| 6.1 | Output 1 | 24 |
| 6.2 | Output 2 | 24 |
| 6.3 | Output 3 | 25 |
| 6.4 | Output 4 | 25 |

# CHAPTER 1. INTRODUCTION

Crude oil has become increasingly important for the global economy, with nearly two-thirds of the world’s electricity generation relying on crude oil and natural gas As an extraordinary commodity, raw petroleum trades involve a wide variety of international players, including oil producing countries, oil companies, suppliers of treatment plants, oil trading countries, and theorists. Due to the importance of the commodity to the development of a country, a sharp shift in the value of crude oil can lead to a turmoil in monetary action and the economy of a country. The cost of crude oil can affect the economy of a country in two ways. A rapid rise in the cost of crude oil has antagonistic effects on financial growth and causes inflation to rise. By contrast, a drop in the cost of crude oil (such as in 1998) may pose serious financial shortfall challenges for oil exporting countries. Various studies have been conducted to visualize the impact of crude oil price changes on the economy of countries. For instance, a study by Sari has summarized the impact of world crude oil prices on the Malaysian economy— specifically on the country’s income as Malaysia is an oil producing country, and its economic growth depends on the price of crude oil. Therefore, an accurate crude oil price prediction mechanism is needed to allow researchers and stakeholders to understand and predict future crude oil prices. It is worth noting that the prediction of crude oil price is considered as a challenging and crucial topic given its high volatility nature. The crude oil spot value arrangement is perceived as nonlinear and non-stationary time arrangement. The crude oil spot value depends on various irregular variables such as weather, crude oil stock levels, GDP growth, political stability, and psychological expectations. Due to the challenging nature of the task, crude oil price prediction has become an interest to many.

# CHAPTER 2. AIM AND SCOPE OF THE PRESENT INVESTIGATION

***2.1 AIM:***

Crude oil has another name called black gold which has an essential role in evolution of global wealth and financial market. Therefore, dynamic information of future expected price will lead to enhancement of decision making at different levels.

***2.2 SCOPE:***

We have come across testing different versions of model using various lookback and alternative tuning methods. The conclusion derived from this study are promising and represent a more precise prediction for the crude oil price in coming days. Specifically, this is an attempt made to forecast price prediction using long short-term memory neural network rather than using convolutional neural network

# CHAPTER 3. EXPERIMENTAL OR MATERIALS AND METHODS

# ;ALGORITHMS USED

***3.1 Pre Requisites***

To complete this project you should have the following software  and packages

**Anaconda Navigator :**

Anaconda Navigator is a free and open-source distribution of the Python and R programming languages for data science and machine learning related applications. It can be installed on Windows, Linux, and macOS.Conda is an open-source, cross-platform,  package management system. Anaconda comes with so very nice tools like JupyterLab, Jupyter Notebook,QtConsole, Spyder, Glueviz, Orange, Rstudio, Visual Studio Code. For this project, we will be using Jupiter notebook and spyder.

If you are using **anaconda navigator**, follow the below steps to download the required packages:

* open anaconda prompt as administrator
* Type “pip install tensorflow” (make sure you are working on python 64 bit)
* Type “pip install flask”.
* Type "pip install keras

The above steps allow you to install  Keras and TensorFlow in the anaconda environment.

### 3.2 Data Collection

Deep Learning depends heavily on data, without data, a machine can't learn. It is the most crucial aspect that makes neural network training possible. In Deep Learning projects, we need a training data set. It is the actual data set used to train the model for performing various actions.

### 3.2.1 Download Dataset /Create Dataset

You can collect datasets from different open sources like kaggle.com, data.gov, UCI machine learning repository, etc.The dataset used for this project was obtained from Kaggle.  Please refer to the [link](https://www.kaggle.com/rockbottom73/crude-oil-prices) to download the data set and to know about the dataset.  
This dataset contains two columns

* Date
* Closing Value

It contains crude oil prices from 1988 to 2018.

### 3.3 Data Preprocessing

Data Pre-processing includes the following main tasks

* Import the Libraries.
* Importing the dataset.
* Analyze the data
* Taking care of Missing Data
* Feature Scaling
* Data Visualization
* Splitting Data into Train and Test.
* Creating datasets with sliding windows.

### 3.3.1 Import The Libraries

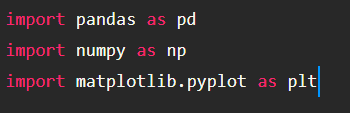
The first step is usually importing the libraries that will be needed in the program.The required libraries to be imported to  Python script are:

**Numpy:**

It is an open-source numerical Python library. It contains a multi-dimensional array and matrix data structures. It can be used to perform mathematical operations on arrays such as trigonometric, statistical, and algebraic routines.

**Pandas**- It is a fast, powerful, flexible, and easy to use open-source data analysis and manipulation tool, built on top of the Python programming language.

**Matplotlib**- Visualisation with python. It is a comprehensive library for creating static, animated, and interactive visualizations in Python.



***Note***: It’s conventional to refer to alias. When you add the alias name at the end of your import statement, your Jupyter Notebook understands that from this point on every time you type alias name, you are actually referring to the particular library.

# CHAPTER 4. RESULTS AND DISCUSSION, PERFORMANCE ANALYSIS

### 4.1 Importing The Dataset

* You might have your data in .csv files, .xlsx files

* Let’s load the excel data file into pandas using the **read\_excel()** function.We will need to locate the directory of the excel file at first (it’s more efficient to keep the dataset in the same directory as your program).



* If your dataset is in some other location, Then

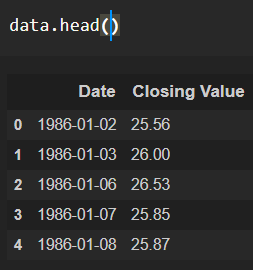
Data=pd.read\_excel(r”File\_location”)

**Note:** r stands for "raw" and will cause backslashes in the string to be interpreted as actual backslashes rather than special characters.

* If the dataset is in the same directory of your program, you can directly read it, without giving raw as r.

### 4.2 Analyze The Data

* head() method is used to return top n (5 by default) rows of a DataFrame or series.



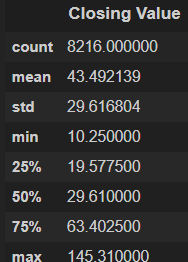
* tail() method is used to return bottom n (5 by default) rows of a DataFrame or series.



* describe() method computes a summary of statistics like count, mean, standard deviation, min, max, and quartile values.

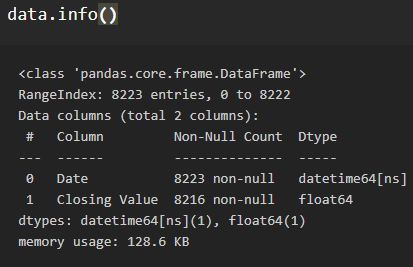


The output is as shown below



From the data we infer that there are 8216 records

* info() gives information about the data



### 4.2.1 Handling Missing Data

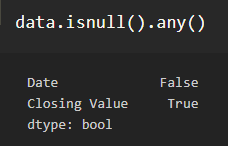
1. After loading the dataset, it is important to check the complete information of such as null values in a column or a row

2. Check whether any null values are there or not. if it is present then the following can be done,

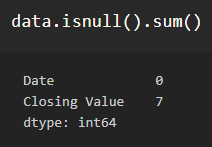
a.Imputing data using Imputation method in sklearn

b.Filling NaN values with mean, median, and mode using fillna() method.

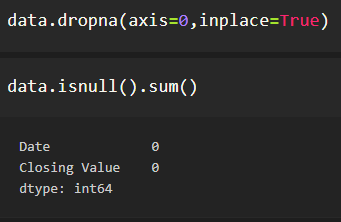
c.Delete the records



We can see that there are null values in the Closing Value Column.Let us check how many numbers of null records present in the Closing Value column using sum() function.



Let us drop the null records from the column.Axis=0 indicates that drop the rows The 'inplace=True' argument stands for the data frame has to make changes permanent

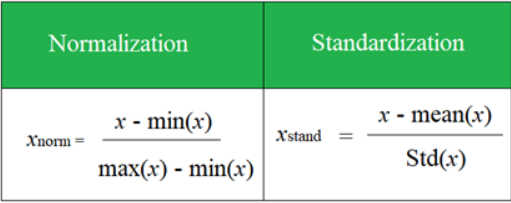
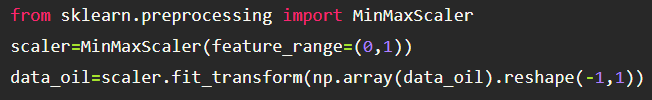


Let us consider Closing Value column in the dataset.reset\_index() is a method to reset index of a Data Frame. reset\_index() method sets a list of integer ranging from 0 to length of data as index.



### 4.2.2 Feature Scaling

Feature scaling is a method used to normalize the range of independent variables or features of data.

The next step is to scale the crude oil prices between (0, 1) to avoid intensive computation. Common methods include Standardization and Normalization .

LSTM are sensitive to the scale of the data. so we apply MinMax scaler

### 4.2.3 Data Visualization

* Data visualization is where a given data set is presented in a graphical format. It helps the detection of patterns, trends, and correlations that might go undetected in text-based data.
* Understanding your data and the relationship presents within it is just as important as any algorithm used to train your machine learning model. In fact, even the most sophisticated machine learning models will perform poorly on data that wasn’t visualized and understood properly.
* To visualize the dataset we need libraries called Matplotlib and Seaborn.
* The Matplotlib library is a Python 2D plotting library that allows you to generate plots, scatter plots, histograms, bar charts, etc.

Let’s visualize our data using the Matplotlib and seaborn library.

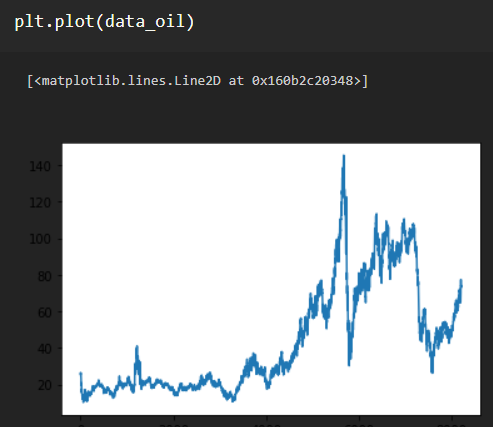
Before diving into the code, let's look at some of the basic properties we will be using when plotting.

xlabel: Set the label for the x-axis.

ylabel: Set the label for the y-axis.

title: Set a title for the axes.

              Legend: Place a legend on the axes.

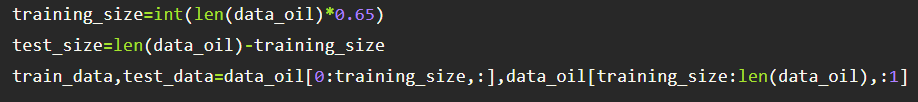


**4.1 Plotting Values**

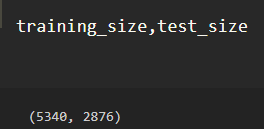
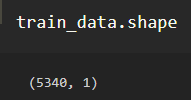
From the graph, we infer that crude oil prices are increasing day by day but, we cannot say that it is completely increasing as there is a significant drop in some years.

### 4.2.4 Splitting Data Into Train And Test

* When you are working on a model and you want to train it, you have a dataset. But after training, we have to test the model on some test dataset. For this, you will need a dataset that is different from the training set you used earlier. But it might not always be possible to have so much data during the development phase. In such cases, the solution is to split the dataset into two sets, one for training and the other for testing.
* But the question is, how do you split the data?
* For time-series data, the sequence of values is important. A simple method that we can use is to split the ordered dataset into train and test datasets. The code below calculates the index of the split point and separates the data into the training datasets with 65% of the observations that we can use to train our model, leaving the remaining 30% for testing the model.



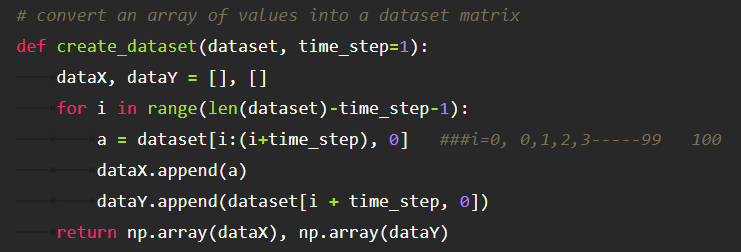
The size of train and test data after splitting



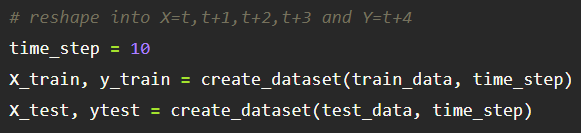
### 4.3 Creating A Dataset With Sliding Windows

A special data structure is needed to cover n-time stamps, based on which LSTM will predict the n +1 tt price. Here the number of past timestamps is set to 10.The function takes two arguments, the dataset which is a NumPy array that we want to convert into a dataset and the time\_step which is the number of previous time steps to use as input variables to predict the next time period, in this case, defaulted to 1.

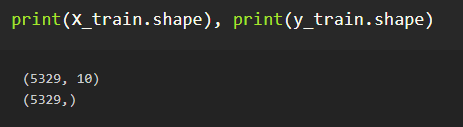
This default will create a dataset where X is the price of crude oil at a given time (t) and Y is the price of crude oil at the next time (t + 1).



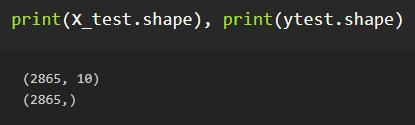
We are applying the function on training data and test data. Hence we get X\_train,y\_train and X\_test,ytest



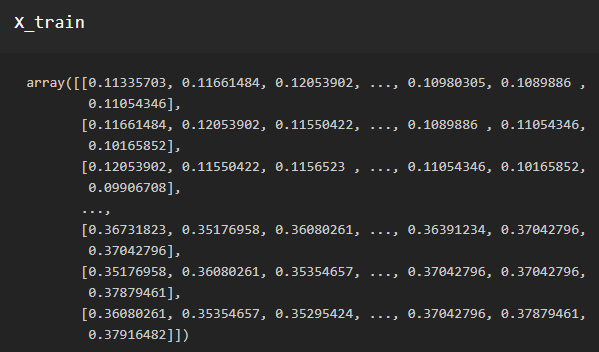
The shape of training data



The shape of test data

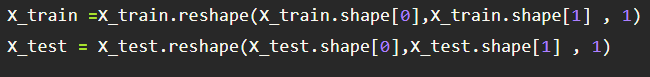


The data of X\_train is as follows



For LSTM model , it is necessary to reshape the X\_train and X\_test into 3 dimensional array before building the model.

# reshape input to be [samples, time steps, features] which is required for LSTM



### 5. MODEL BUILDING

Model Building Includes:

* Import the model building Libraries
* Initializing the model
* Adding LSTM Layers
* Adding Output Layer
* Configure the Learning Process
* Training the model
* Model Evaluation
* Save the Model
* Test the Model

### 5.1 Importing The Model Building Libraries



Importing the necessary libraries

### 5.2 Initializing The Model

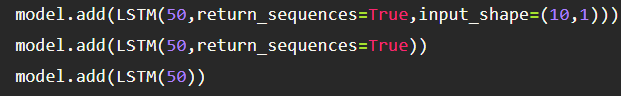
Sequential model is a linear stack of layers. You can create a Sequential model by passing a list of layer instances to the constructor from Keras. models import Sequential from Keras as follows.



### 5.3 Adding LSTM Layers

* Note for the LSTM layer, units is the number of LSTM neurons in the layer. 50 neurons will give the model high dimensionality, enough to capture the upwards and downward trends.
* return\_sequences is True as we need to add another LSTM layer after the current one. input\_shape corresponds to the number of time stamps and the number of indicators.

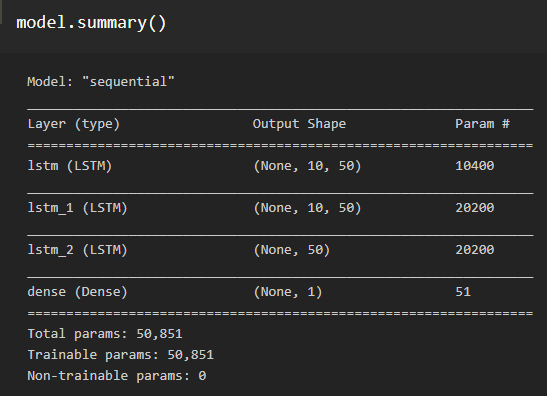
Following the above same method, add 2nd, 3rd LSTM layer



### 5.4 Adding Output Layers

The dense layer is a deeply connected neural network layer. It is the most common and frequently used layer.Finally, add the output layer. The output dimension is 1 since we are predicting 1 price each time.

  
Understanding the model is a very important phase to properly use it for training and prediction purposes. Keras provides a simple method, summary to get the full information about the model and its layers.



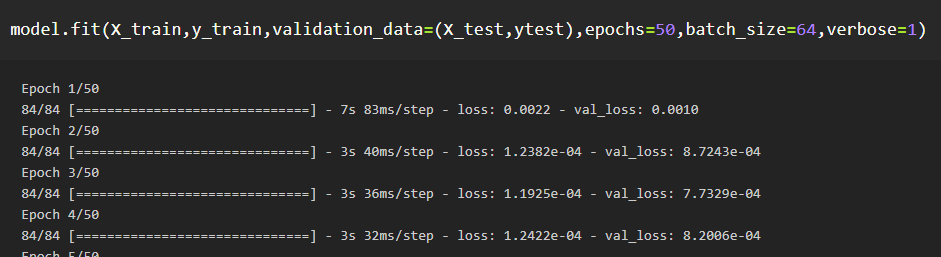
### 5.5 Configure The Learning Process

* The compilation is the final step in creating a model. Once the compilation is done, we can move on to the training phase. The loss function is used to find error or deviation in the learning process. Keras requires loss function during the model compilation process.
* Optimization is an important process that optimizes the input weights by comparing the prediction and the loss function. Here we are using adam optimizer
* Metrics are used to evaluate the performance of your model. It is similar to loss function, but not used in the training process



### 5.6 Train The Model

Now ,let us train our model RNN weights are updated every 64 stock prices with a batch size of 64. Try more batches and epochs if the loss of the model is not converging.



**Arguments:**

* Epochs: an integer and number of epochs we want to train our model for.
* validation\_data can be either:
  + an inputs and targets list
  + a generator
  + an inputs, targets, and sample\_weights list which can be used to evaluate the loss and metrics for any model after any epoch has ended.

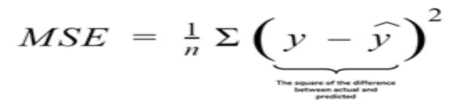
### 5.7 Model Evaluation

Finally, we need to check to see how well our model is performing on the test data.

**Regression Evaluation Metrics:**

### 1.Mean Squared Error (MSE):

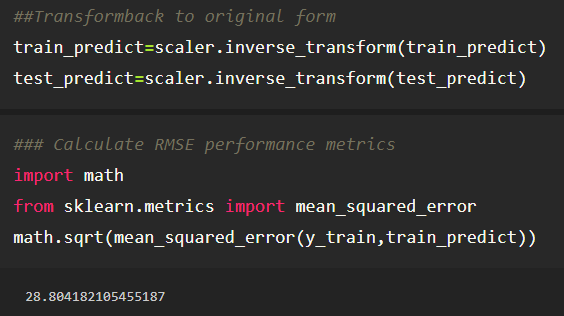
MSE or Mean Squared Error is one of the most preferred metrics for regression problems. It is simply the average of the squared difference between the target value and the value predicted by the regression model. As it squares the differences, it penalizes even a small error which leads to over-estimation of how bad the model is. It is preferred more than other metrics because it is differentiable and hence can be optimized better.



### 2.RMSE:Root Mean Square Error:

### RMSE is the square root of the averaged squared difference between the target value and the value predicted by the model. It is preferred more in some cases because the errors are first squared before averaging which poses a high penalty on large errors. This implies that RMSE is useful when large errors are undesired.

### 



### 5.8 Save The Model

The model is saved with .h5 extension as follows An H5 file is a data file saved in the Hierarchical Data Format (HDF). It contains multidimensional arrays of scientific data.

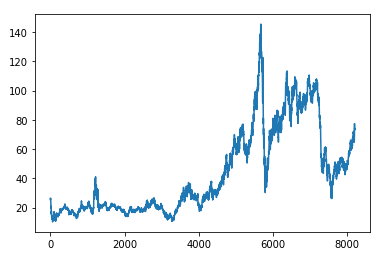




### 5.9 Test The Model

Finally, we can generate predictions using the model for both the train and test to visualize the model. We must shift the predictions so that they aline on the x-axis with the original dataset. Once prepared, the data is plotted, showing the original dataset in blue, the predictions for the train dataset in green the predictions on the unseen test dataset in orange.

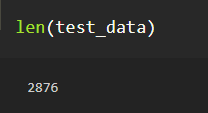




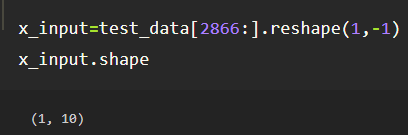
**5.1 Testing the Model**

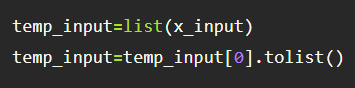
Prediction for next 10 days:

Now let us predict the price of crude oil for next 10 days.As the length of the test data is 2876, We are taking previous 10 days input i.e., from index 2866 -2876 to predict 2867 th day output

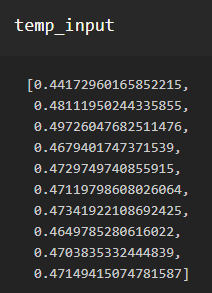


Create the input and reshape it and convert it into list





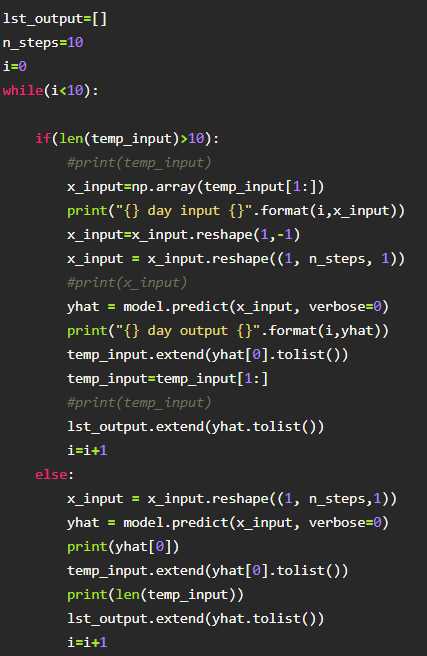
We can see temp\_input contains last 10 days price list



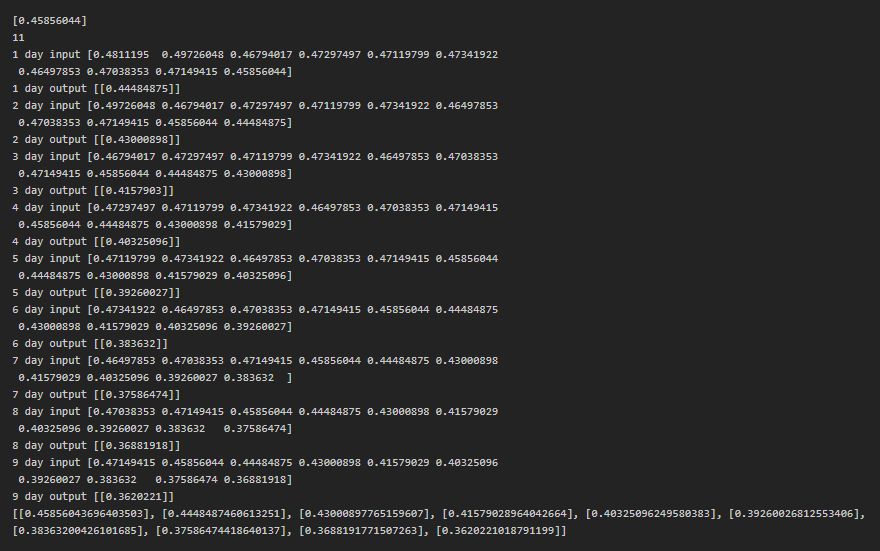
For predicting next 10 days crude oil prices we consider n\_steps=10

We create the input for prediction, index starting from the date 10 days before the first date in the test dataset. Then, reshape the inputs to have only 1 column and predict using model\_predict predefined function.

This can be done using the below code



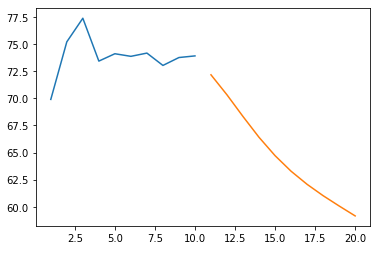
The output is as shown follows We can infer that it is taking 10 inputs and predicting the day 11 th output.



let’s create a visualization plot to easily review the prediction.

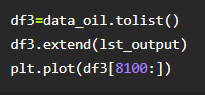


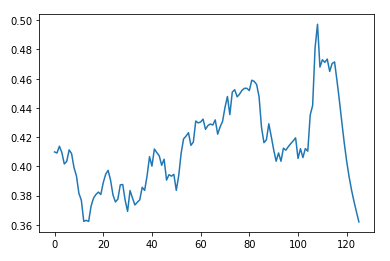
Don’t forget, what we have predicted is the scaled values, so we need to reverse the prediction



**5.2 Scalar Inverse Form Plotting**

Let us merge the the past data and next 10 days output prediction

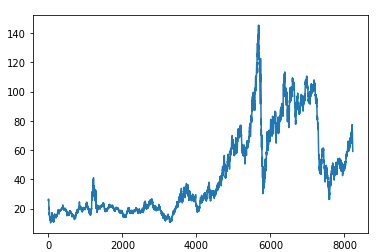


****

**5.3 Ten Days Output Prediction**

Don’t forget, what we have predicted is the scaled values, so we need to reverse the prediction





### 5.4 Inversing the Values

### 

### 6. APPLICATION BUILDING

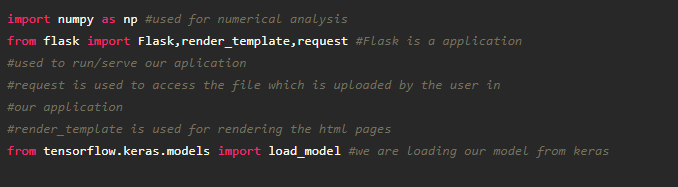
### 6.1 Create An HTML File

* We use HTML to create the front end part of the web page.
* Here, we created 2 HTML pages- index.html, web.html.
* index.html displays the home page.
* web.html accepts the values from the input and displays the prediction.

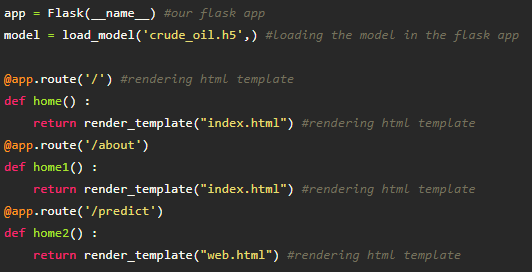
**CREATE** index.html and web.html

### 6.2 Build Python Code

* Let us build a flask file ‘app.py’ which is a web framework written in python for server-
* side scripting. Let’s see step by step procedure for building the backend application.
* The app starts running when the “\_\_name\_\_” constructor is called in main.
* render\_template is used to return HTML files.
* “GET” method is used to take input from the user.
* “POST” method is used to display the output to the user.
* Importing Libraries

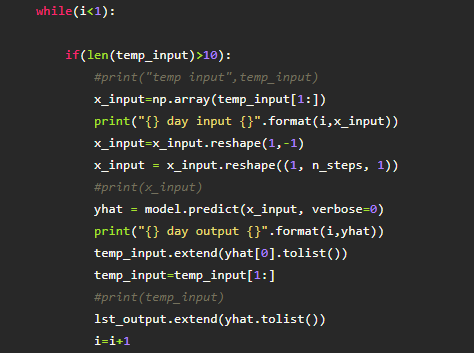


* Routing to the html Page



* For predicting next day’s crude oil prices we consider n\_steps=10.

We take the input for prediction, index starting from the date 10 days before the first date in the test dataset. Then, reshape the inputs to have only 1 column and predict using model\_predict predefined function.



### 

### 

### 6.3 Run The App In Local Browser

* Open anaconda prompt from the start menu
* Navigate to the folder where your python script is.
* Now type “python app.py” command
* Navigate to the localhost where you can view your web page

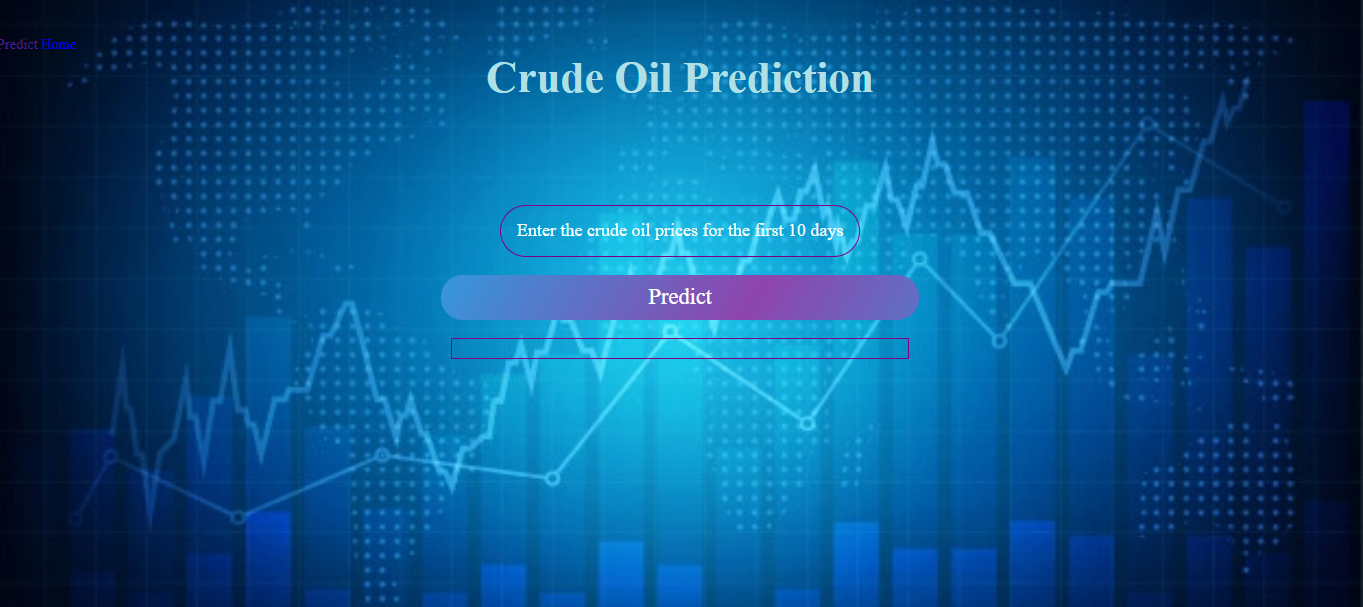
<http://localhost:8080/>

### 6.4 Showcasing Prediction On UI



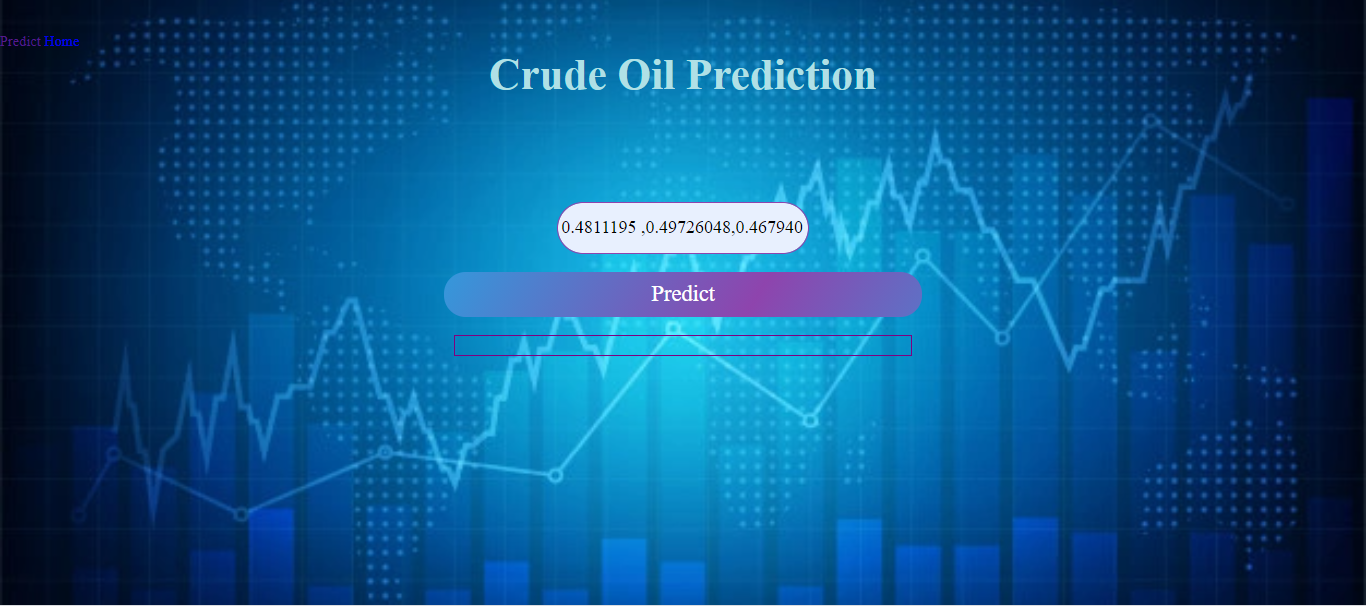
**6.1 Output 1**

This is our home page where we get to know the summary of the project.



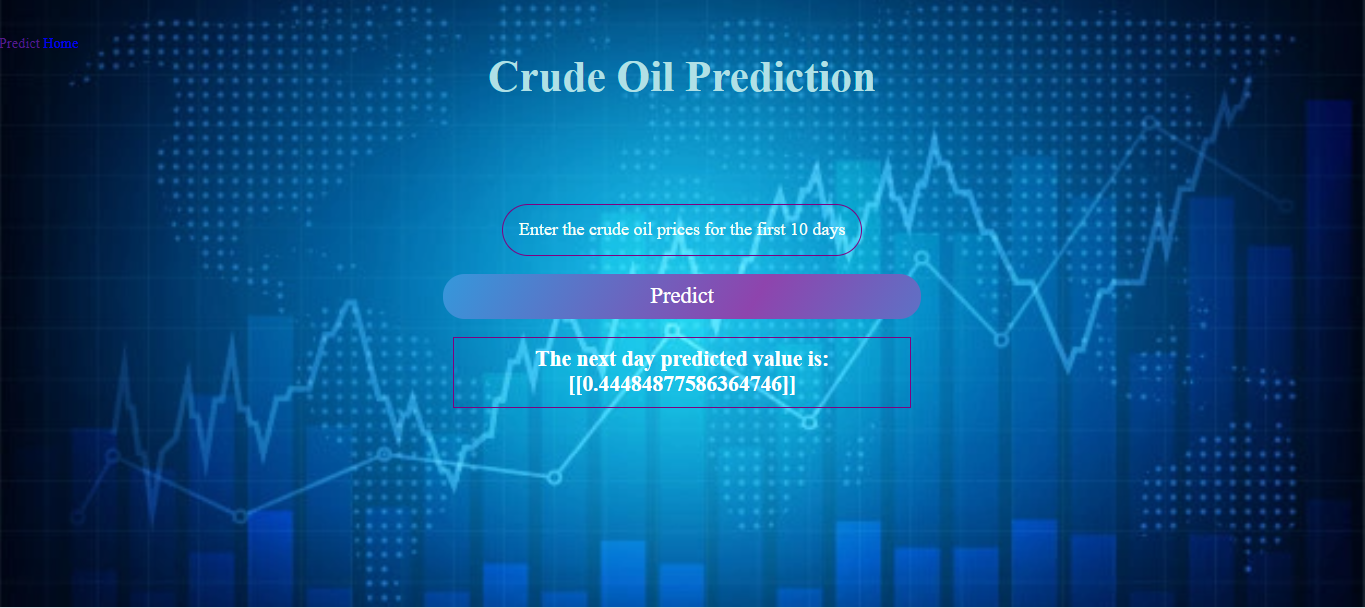
**6.2 Output 2**

As this is is a time series approach, the user has to give past 10 days crude oil prices as input to predict the future crude oil price.



**6.3 Output 3**

Here we give the input- First ten days and then we click on the predict button to predict the next day’s price.



**6.4 Output 4**

As we see the predicted output is displayed on the User Interface

### 7. REFERENCES

### [1] Mohammad Reza Mahdiani and Ehsan Khamehchi, “A modified neural network model for predicting the crude oil price”, Intellectual Economics, vol. 10, no. 2, pp. 71-77, Aug. 2016.

### [2] Manel Hamdi and Chaker Aloui, \"Forecasting Crude Oil Price Using Artificial NeuralNetworks: A Literature Survey,\" Economics Bulletin, AccessEcon, vol. 35, no. 2, pp. 1339-1359, 2015.

### [3] Yu Runfang, Du Jiangze and Liu Xiaotao, “Improved Forecast Ability of Oil Market Volatility Based on combined Markov Switching and GARCH-class Model, Procedia Computer Science, vol. 122, pp. 415-422, 2017.

### [4] K. Greff, R. K. Srivastava, J. Koutník, B. R. Steunebrink and J. Schmidhuber, \"LSTM: A Search Space Odyssey,\" IEEE Transactions on Neural Networks and Learning Systems, vol. 28, no. 10, pp. 2222-2232,Oct. 2017.

### [5] S. Hochreiter and J. Schmidhuber, “Long short-term memory,” Neural Comput., vol. 9, no. 8, pp. 1735–1780, Nov. 1997.

### [6] Jammazi, R., Aloui, C.: Crude oil price forecasting: experimental evidence from wavelet decomposition and neural network modeling. Energy Econ. 34(3), 828–841 (2012)

### [7] S. Moshiri, and F. Foroutan, “Forecasting nonlinear crude oil futures prices,” The Energy Journal vol. 27, pp. 81-95, 2005.

### [8] Siddhi Vinayak Kulkarni and Imad Haidar, Forecasting Model for Crude Oil Price Using Artificial Neural Networks and Commodity Futures Prices. International Journal of Computer Science and Information Security, vol. 2, no.1, June 2009.

### [9] Hamdi and Aloui, \"Machine learning approach for crude oil price prediction with Artificial Neural Networks-Quantitative (ANN-Q) model,\" The 2010 International JointConference on Neural Networks (IJCNN), Barcelona, pp. 1-8, 2010.

### [10] Abdullah and Zeng.: Exploring the core factors and its dynamic effects on oil price: An application on path analysis and BVAR-TVP model. Energy Policy 39(12), 8022–8036 (2011)

### 8.SOURCE CODE

### DATA PREPROCESSING

### Import the Libraries

### import pandas as pd

### import numpy as np

### import matplotlib.pyplot as plt

### Importing the Datase

### data=pd.read\_excel(r"D:\Crude Oil Prices Daily.xlsx")

### Analyze the Data

### data.head()

### data.tail()

### data.describe()

### data.info()

### Handling Missing Data

### data.isnull().any()

### data.isnull().sum()

### data.dropna(axis=0,inplace = True)

### data.isnull().sum()

### data\_oil=data.reset\_index()['Closing Value']

### data\_oil

### Feature Scaling

### from sklearn.preprocessing import MinMaxScaler

### scaler=MinMaxScaler(feature\_range=(0,1))

### data\_oil=scaler.fit\_transform(np.array(data\_oil).reshape(-1,1))

### Data Visualization

### plt.plot(data\_oil)

### Splitting Data Into Train And Test

### training\_size=int(len(data\_oil)\*0.65)

### test\_size=len(data\_oil)-training\_size

### train\_data,test\_data=data\_oil[0:training\_size,:],data\_oil[training\_size:len(data\_oil),:1]

### training\_size,test\_size

### train\_data.shape

### Creating A Dataset With Sliding Windows

### # convert an array of values into a dataset matrix

### def create\_dataset (dataset, time\_step=1):

### dataX, dataY = [], []

### for i in range(len(dataset)-time\_step-1):

### a = dataset[i:(i+time\_step), 0]

### dataX.append(a)

### dataY.append(dataset[i + time\_step, 0])

### return np.array(dataX), np.array(dataY)

### # reshape into x=t,t+1,t+2,t+3 and Y=t+4

### time\_step = 10

### x\_train, y\_train = create\_dataset(train\_data, time\_step)

### x\_test, ytest = create\_dataset(test\_data, time\_step)

### print (x\_train.shape), print(y\_train.shape)

### print (x\_test.shape), print(ytest.shape)

### x\_train

### x\_train =x\_train.reshape(x\_train.shape[0],x\_train.shape[1], 1)

### x\_test =x\_test.reshape(x\_test.shape[0],x\_test.shape[1], 1)

### MODEL BUILDING

### Importing The Model Building Libraries

### import tensorflow as tf

### import tensorflow\_datasets as tfds

### from tensorflow.keras.models import Sequential

### Initializing The Model

### model = Sequential()

### Adding LSTM Layers

### from tensorflow.keras.layers import LSTM

### from tensorflow.keras.layers import Dense

### 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()

### 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)

### Model Evaluation

### train\_predict = model.predict(x\_train)

### test\_predict = model.predict(x\_test)

### 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))

### Save The Model

### from tensorflow.keras.models import load\_model

### model.save("crude\_oil.h5")

### Test The Model

### 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)

### x\_input=test\_data[2866:].reshape(1,-1)

### x\_input.shape

### temp\_input=list(x\_input)

### temp\_input=temp\_input[0].tolist()

### temp\_input

### lst\_output=[]

### n\_steps=10

### i=0

### while(i<10):

### if(len(temp\_input)>10):

### #print(temp\_input)

### 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)) #print(x\_input)

### yhat = model.predict(x\_input, verbose=0)

### print("{} day output {}".format(i,yhat))

### temp\_input.extend(yhat[0].tolist())

### temp\_input=temp\_input[1:] #print(temp\_input)

### 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

### day\_new=np.arange(1,11)

### day\_pred=np.arange(11,21)

### len(data\_oil)

### 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)