

## **1 INTRODUCTION**

### 1.1 Overview

Oil demand is inelastic, therefore the rise in price is good news for producers because they will see an increase in their revenue. Oil importers, however, will experience increased costs of purchasing oil. Because oil is the largest traded commodity, the effects are quite significant. A rising oil price can even shift economic/political power from oil importers to oil exporters. The crude oil price movements are subject to diverse influencing factors.

### 1.2 Purpose

This Guided Project mainly focuses on applying Neural Networks to predict the Crude Oil Price. This decision helps us to buy crude oil at the proper time. Time series analysis is the best option for this kind of prediction because we are using the Previous history of crude oil prices to predict future crude oil. So we would be implementing RNN(Recurrent Neural Network) with LSTM(Long Short Term Memory) to achieve the task

## **2 LITERATURE SURVEY**

### 2.1 Existing problem

Crude oil is the world's leading fuel, and its prices have a big impact on the global environment, economy as well as oil exploration and exploitation activities. Oil price forecasts are very useful to industries, governments and individuals. Although many methods have been developed for predicting oil prices, it remains one of the most challenging forecasting problems due to the high volatility of oil prices. So we propose a novel approach for crude oil price prediction based on applying neural networks to predict the crude oil price.

### 2.2 Proposed solution

In this propose system, we have used LSTM based recurrent neural networks for the purpose of crude oil price prediction.

Recurrent neural networks (RNN) have been proved to be one

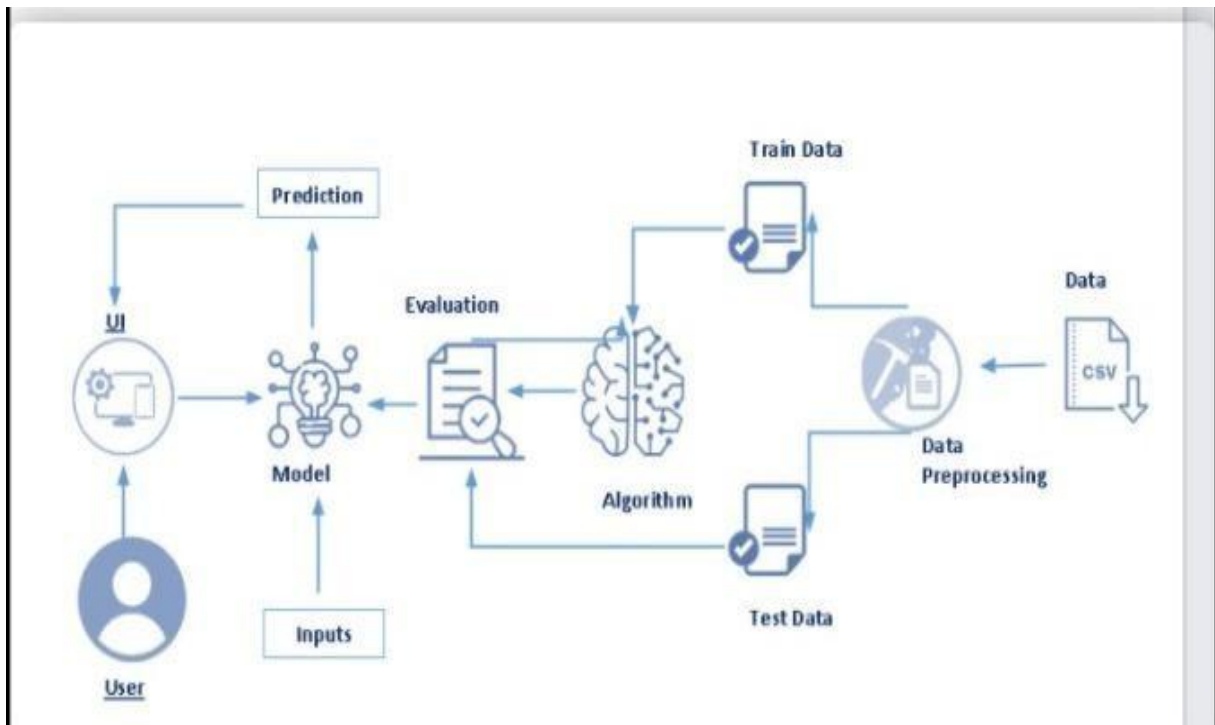
# Crude Oil Price Prediction

of the most powerful models for processing time-series based sequential data. LSTM is one of the most successful RNN. In this proposed system, the crude oil price of last 10 days are evaluated to predict the price of crude oil on the next day. The results indicate that large lookups do not necessarily improve

the accuracy of the predictions of crude oil prices. It has been found that lookups up to the value of 10 are ideal for crude oil price prediction purposes. It has also been found that just increasing the number of LSTM layers do not have much impact on the accuracy of the results

## 3 THEORITICALANALYSIS

### 3.1 Block diagram



## 3.2 Hardware / Software designing

### **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.

**Tensor flow:** TensorFlow is an end-to-end open-source platform for machine learning. It has a comprehensive, flexible ecosystem of tools, libraries, and community resources that lets researchers push the state-of-the-art in ML and developers can easily build and deploy ML-powered applications.

**Keras:** Keras leverages various optimization techniques to make high-level neural network API easier and more performant. It supports the following features:

- Consistent, simple, and extensible API.
- Minimal structure - easy to achieve the result without any frills.
- It supports multiple platforms and backends.
- It is a user-friendly framework that runs on both CPU and GPU.
- Highly scalability of computation.

Flask: Web framework used for building Web applications

- IBM Watson Studio - IBM Watson Studio helps data scientists and analysts prepare data and build models at scale across any cloud.

- IBM Watson Machine Learning - IBM WatsonMachine Learning helpsdata scientists and developers accelerate AI and machine-learning deployment.
- IBM Cloud Object Storage - IBM Cloud Object Storage makes it possible to store practically limitless amounts of data, simply and cost effectively.

## 4 EXPERIMENTAL INVESTIGATION

### 1. Dataset 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.

We 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](#) to download the data set and to know about the dataset

This dataset contains two columns

➤ Date

It contains crude oil prices from 1988 to 2018.

we can collect dataset from different open sources like kaggle.com

Kaggle is the world's largest data science community with powerful tools and resources to help you achieve your data science goals..

The kaggle repository link is :

<https://www.kaggle.com/rockbottom73/crude-oil-prices>

### ii) Data Preprocessing :

Data Pre-processing includes the following main tasks

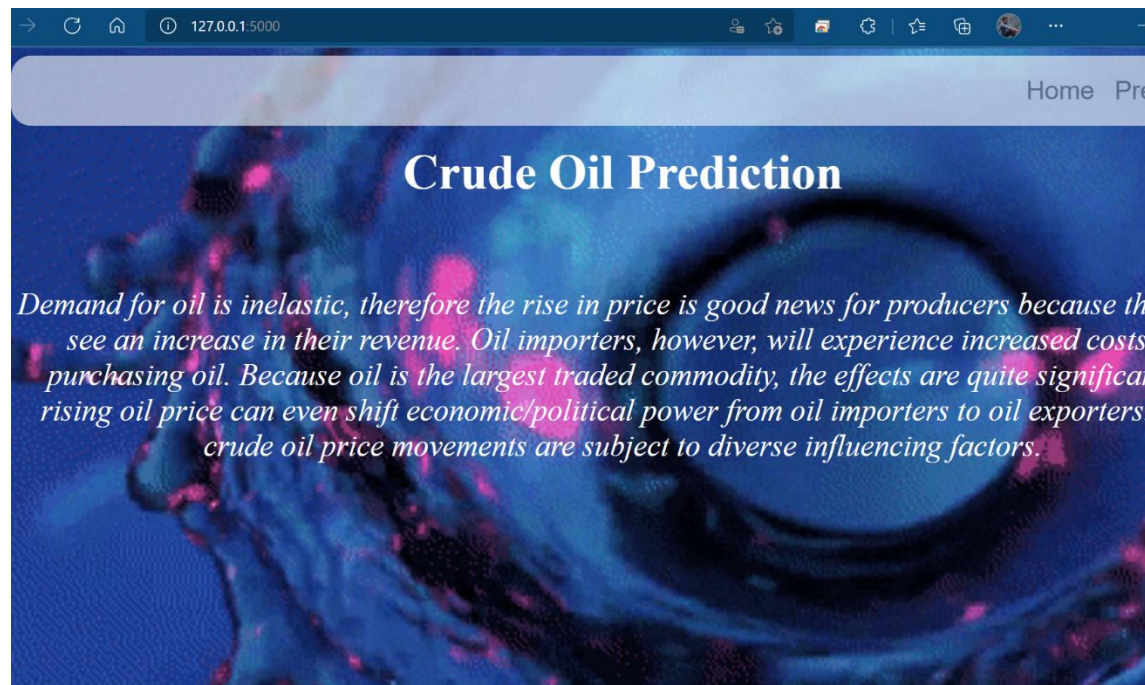
- Import the Libraries.
- Importing the dataset.

- Analyze the data
- Taking care of Missing Data
- Feature Scaling
- Data Visualization
- Splitting Data into Train and Test.
- Creating datasets with sliding windows.

## 4 FLOWCHART

## 5 RESULT

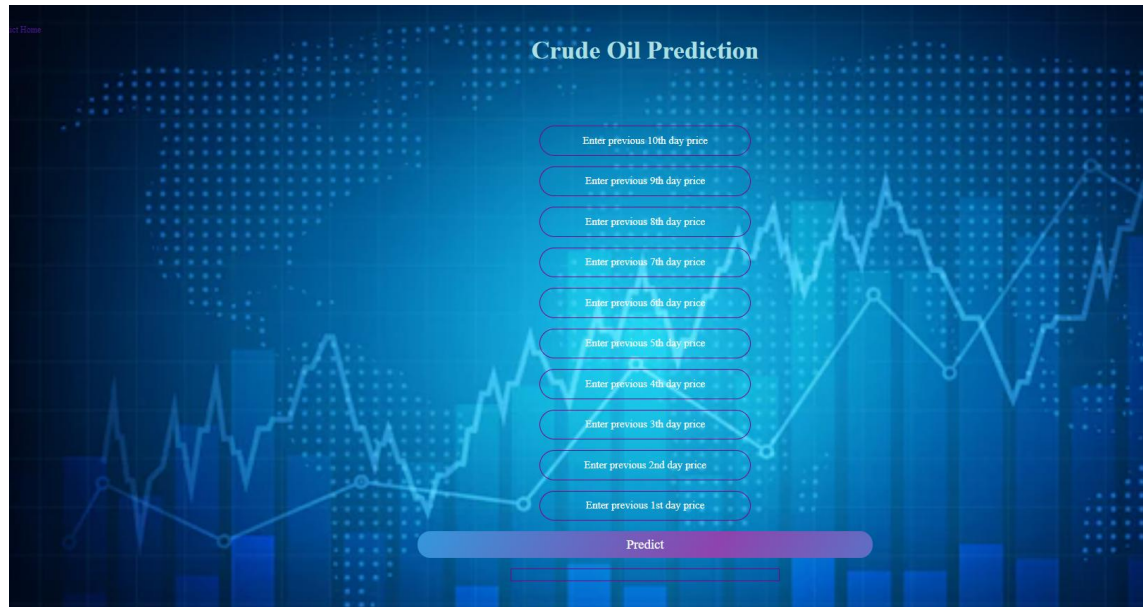
This is our home page where we get to know the summary of the project



As this 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



# Crude Oil Price Prediction



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



Here we got the prediction of next day crude oil price as

0.4815692603588104

## 6 ADVANTAGES & DISADVANTAGES

PREDICTION of crude oil prices has been a wide topic for ages. People use their intuition and lot of techniques to guess the prices of crude oil. It takes a lot of knowledge about

# *Crude Oil Price Prediction*

the crude oil to accurately predict it. Predicting the crude oil price is very significant in various economic, political and industrial areas, both for crude oil importer and exporter countries. Since crude oil is the most important strategic resource around the globe; it has become the crucial commodity for the world's economy. Thus, prediction of prices of crude oil has always been considered as a very exciting and challenging task which drew the curiosity of professionals, researchers and organizations all over the world . Moreover, crude oil volatility has a critical impact on macroeconomic parameters such as such as inflation, unemployment, exchange rate, economic growth of countries whose economy rely heavily on crude oil export or import. Thus, crude oil price prediction can help governments of countries of the world in economic policymaking and make quick and operative economic decisions to hedge against probable risk in these economic parameters . Therefore, forecasting of crude oil prices is quite useful and is also the objective of this paper.

## **7 APPLICATIONS**

Crude oil price prediction can help governments of countries of the world in economic policymaking and make quick and operative economic decisions to hedge against probable risk in these economic parameters .

## **8 CONCLUSION AND FUTURE SCOPE**

Crude oil price prediction plays a significant role in world economy and its accurate prediction has significant benefits for the economic conditions of a country. In this direction, an effort has been in this paper. This paper has proposed an LSTM based network for better prediction of crude oil prices.

# *Crude Oil Price Prediction*

The results obtained from the work are quite encouraging. The results indicate that large lookups do not necessarily improve the accuracy of the predictions of crude oil prices. It has been found that lookups up to the value of 10 are ideal for crude oil price prediction purposes. It has also been found that just increasing the number of LSTM layers do not have much impact on the accuracy of the results. In future work, current market and political conditions can also be taken into consideration in crude oil price forecasting for even better results.

## 9 BIBLIOGRAPHY

<https://publications.waset.org/10008992/crude-oil-price-prediction-using-lstm-networks>

<https://www.kaggle.com/rockbottom73/crude-oil-prices>

## 11 APPENDIX

### Source Code

```
pip install tensorflow==2.2.0
pip install keras==2.2.5
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import os, types
import pandas as pd
from botocore.client import Config
import ibm_boto3
def __iter__(self): return 0
# @hidden_cell
# The following code accesses a file in your IBM Cloud Object Storage.
# It includes your credentials.
# You might want to remove those credentials before you share the
# notebook.
client_39015f7cbe0c4760900af6d9649f8c3b =
ibm_boto3.client(service_name='s3',
```



## Crude Oil Price Prediction

```
ibm_api_key_id='Xj8fhZA5dSZuQYL1yxcDK5LtU1MhDXHacSH654
OFut_K',
ibm_auth_endpoint="https://iam.cloud.ibm.com/oidc/token",
    config=Config(signature_version='oauth'),
    endpoint_url='https://s3.private.us.cloud-object-
storage.appdomain.cloud')
body =
client_39015f7cbe0c4760900af6d9649f8c3b.get_object(Bucket='new-
donotdelete-pr-hv21clbef0djyz',Key='Crude Oil Prices
Daily.xlsx')['Body']
data = pd.read_excel(body.read())
data.head()
data.tail()
data.describe()
data.info()
data.isnull().any()
data.isnull().sum()
data.dropna(axis=0,inplace=True)
data.isnull().sum()
data.shape
import tensorflow as tf
tf.__version__
import tensorflow.keras
tensorflow.keras.__version__
data_oil=data.reset_index()['Closing Value']
data_oil
plt.plot(data_oil)
from sklearn.preprocessing import MinMaxScaler
scaler=MinMaxScaler(feature_range=(0,1))
data_oil=scaler.fit_transform(np.array(data_oil).reshape(-1,1))
print(data_oil)
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:le
n(data_oil),:1]
training_size,test_size
train_data
train_data.shape
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]   ###i=0, 0,1,2,3-----99   100
    dataX.append(a)
    dataY.append(dataset[i + time_step, 0])
return np.array(dataX), np.array(dataY)
time_step = 10
X_train, y_train = create_dataset(train_data, time_step)
```

## *Crude Oil Price Prediction*

```
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
y_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)
from tensorflow.keras.layers import Dense
from tensorflow.keras.layers import LSTM
from tensorflow.keras.models import Sequential, load_model
model = Sequential()
model.add(LSTM(50, return_sequences=True, input_shape=(10, 1)))
model.add(LSTM(50, return_sequences=True))
model.add(LSTM(50))
model.add(Dense(1))
model.compile(loss='mean_squared_error', optimizer='adam')
model.summary()
model.fit(X_train, y_train, validation_data=(X_test, ytest), epochs=50, batch_size=64, verbose=1)
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))
math.sqrt(mean_squared_error(ytest, test_predict))
look_back = 10
trainPredictPlot = np.empty_like(data_oil)
trainPredictPlot[:, :] = np.nan
trainPredictPlot[look_back:len(train_predict)+look_back, :] =
train_predict
# shift test predictions for plotting
testPredictPlot = np.empty_like(data_oil)
testPredictPlot[:, :] = np.nan
testPredictPlot[len(train_predict)+(look_back*2)+1:len(data_oil)-1, :] =
test_predict
# plot baseline and predictions
plt.plot(scaler.inverse_transform(data_oil))
plt.plot(trainPredictPlot)
plt.plot(testPredictPlot)
plt.show()
model.save("crude_oil.h5")
!tar -zcvf crude_oil.tgz crude_oil.h5
ls -l
len(test_data)
x_input = test_data[2866:].reshape(1, -1)
```

## Crude Oil Price Prediction

```
x_input.shape
temp_input=list(x_input)
temp_input=temp_input[0].tolist()
temp_input
len(temp_input)
from numpy import array
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
print(lst_output)
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)
ls -l
!pip install watson-machine-learning-client --upgrade
!pip install --upgrade "ibm-watson>=5.2.3"
from ibm_watson_machine_learning import APIClient
```

## *Crude Oil Price Prediction*

```
wml_credentials = { "url": "https://us-south.ml.cloud.ibm.com",
                    "apikey":
                    "59VBsHIzN8nA4LJxHCKmWxQzXuvGU9FCdZdA6FBWGltY"
                    }
client=APIClient(wml_credentials)
def guide_from_space_name(client,space_name):
    space = client.spaces.get_details()
    return(next(item for item in space['resources']if
item['entity']['name']== space_name)['metadata']['id'])
space_uid = guide_from_space_name(client,'models')
print("Space UID = " + space_uid)
client.set.default_space(space_uid)
client.software_specifications.list()
software_spec_uid =
client.software_specifications.get_uid_by_name("tensorflow_2.4-py3.7-
horovod")
software_spec_uid
!pip install watson-machine-learning-client
model_details = client.repository.store_model(model="crude_oil.tgz",
                                              meta_props= {

client.repository.ModelMetaNames.NAME:"models",

client.repository.ModelMetaNames.TYPE:"keras_2.2.4",

client.repository.ModelMetaNames.SOFTWARE_SPEC_UID:software
_spec_uid

        })
model_id = client.repository.get_model_uid(model_details)
model_id
ypred=model.predict(X_test)
ypred
pwd
model = load_model('crude_oil.h5')
```