JEPPIAAR ENGINEERING COLLEGE

CHENNAI - 600119.

NALAYA THIRAN PROJECT

CRUDE OIL PRICE PREDICTION

Category: Artificial Intelligence

Team ID: PNT2022TMID26965

Team Members:

Mohammed Sufiyan F (310819104054)

Mathimithran T (310819104711)

Prembabu C (310819104061)

Jack Melony G (310819104038)

TABLE OF CONTENTS

CHAPTER 1

INTRODUCTION

Crude oil is a yellow black naturally occurring liquid found in geological formations beneath the Earth's surface, it can be separated into various kinds of consumer fuels through the process of fractional distillation. Crude oil is the most important energy resources on the Earth right now. So far, it remains the world's leading fuel, with nearly one-third of global energy consumption. So, forecasting the price of crude oil is of great significance for energy policymakers, market participants, portfolio diversification, and energy risk management. There are many factors influencing the crude oil price, and the influence period of each factor on the crude oil prices is not consistent, so the crude oil prices have nonlinear characteristics.

However, identifying the formation process of crude oil prices is of significance for accurate prediction, but this process is complicated. Due to strong chain effects owned by this crude oil market, any changes in the factors involved will have exclusive impact to the price. Furthermore, the crude oil price contributes over 50% on the average price of petroleum and it is one of the most used commodities around the globe. Therefore, every increment and decrement that occurs to the crude oil price will then also give impact to the price of petroleum and later correspond to the global economy. A good prediction tool is crucial to be developed for this matter. Therefore, we try to use the machine learning methods to deal with the vague influence among various factors.

The formation process of crude oil prices can lead the traditional econometrics model to have a large error in crude oil price prediction, but the RNN and LSTM models can fit well. We have considered financialization of crude oil markets. The commodity attributes form the long-term trend of crude oil prices, and its financial attributes cause short-term fluctuation. In this paper, we try to forecast the price of crude oil from both spatial perspective and historical perspective.

2

1.1   PROJECT OVERREVIEW

Crude oil is amongst the most important resources in today's world, it is the chief fuel and its cost has a direct effect on the global habitat, our economy and oil exploration, exploitation and other activities. Prediction of oil prices has become the need of the hour; it is a boon to many large and small industries, individuals, and the government. The evaporative nature of crude oil, its price prediction becomes extremely difficult and it is hard to be precise with the same. Several different factors that affect crude oil prices. The main advantage of this crude oil price prediction using artificial intelligence is continuously captures the unstable pattern of the crude oil prices which have been incorporated by finding out the optimal lag and number of the delay effect that controls the prices of crude oil.

1.2   PURPOSE

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.

This 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.

CHAPTER 2

LITERATURE SURVEY

S.NO	Title	Authors	Publication Date	Methodology	Merits	Demerits
1	Brent crude oil price forecast utilizing Deep Neural Network Architectures	Amir Daneshvar and Maryam Ebrahimi	05 May 2022	Artificial Neural Network, Deep Learning	The LSTM layers results in more accurate result.	Crude oil price signals exhibit highly nonlinear and complex behavior.
2.	Crude oil prices and volatility prediction by a hybrid model based on kernel extreme learning machine	Hongli Niu and Yazhi Zhao	17 September 2021	VMD-KELM	The VMD-KELM model shows a more powerful ability than other models in improving the precision of forecasting crude oil volatility.	-
3.	Crude oil price prediction using ANN	Nalini Gupta and Shobhit Nigam	January 2020	Artificial Neural Network	ANN model is effective. This capture the changing pattern of prices. Prediction is accurate.	Market trends have to be planned, then the ANN model will perform.
4.	Crude oil price prediction using complex network and deep learning algorithms	Makumbonori Bristone, Rajesh Prasad, Adamu Ali Abubakar	19 June 2019	Artificial Neural Network, Deep Learning	The appropriate number of LSTM layers can effectively improve the model.	The other factors that affect the crude oil price volatilities such as economic growth, exchange rate demand are not considered.
5.	Daily crude oil price forecasting using Hybridizing wavelet and Artificial Neural Network Model	Ani Shabri and Ruhaidah Samsudin	16 July 2014	Artificial Neural Network	The hybrid model showed a great improvement in crude oil price modeling and	-

4

				produced better forecasts than ANN model alone.	
6.	Machine Learning Approach for crude oil price prediction with Artificial Neural Networks-Quantitative (ANN-Q) model	Abdullah	-	Artificial Neural Network	Returns function had successfully proved to cleanse and uniform the data from errors and noises hence, the crisp prediction result.	
7.	A novel look back N feature approach towards prediction of crude oil price	Rudra Kalyan Nayak	-	ARIMA, LBNF Algorithm	Attained better training and accuracy by shifting the dataset into n class problem and more scope to classifier.	-

2.1 EXISTING PROBLEM

Several machine learning techniques were proposed for oil price prediction, such as artificial neural networks and support vector machine These are nonlinear models which may produce more accurate predictions if the oil price data are strongly nonlinear. However, these machine learning techniques, like other traditional machine learning techniques, rely on a fixed set of training data to train a machine learning model and then apply the model to a test set. Such an approach works well if the training data and the test data are generated from a stationary process but may not be effective for non-stationary time series data such as oil price data

5

2.2 REFERENCES

1. Zhao, Y., Li, J. and Yu, L., 2017. A deep learning ensemble approach for crude oil price forecasting. Energy Economics, 66, pp.9-16.

2. Xie, W., Yu, L., Xu, S. and Wang, S., 2006, May. A new method for crude oil price forecasting based on support vector machines. In International conference on computational science (pp. 444-451). Springer, Berlin, Heidelberg.

3. Zhang, J.L., Zhang, Y.J. and Zhang, L., 2015. A novel hybrid method for crude oil price forecasting. Energy Economics, 49, pp.649-659.

4. Bashiri Behmiri, N. and Pires Manso, J.R., 2013. Crude oil price forecasting techniques: a comprehensive review of literature. Available at SSRN 2275428.

5. Li, X., Shang, W. and Wang, S., 2019. Text-based crude oil price forecasting: A deep learning approach. International Journal of Forecasting, 35(4), pp.1548 - 1560.

2.3 PROBLEM STATEMENT DEFINITION

The crude oil price prediction task is interesting as well as divides researchers and academics into two groups those who believe that we can devise mechanisms to predict the market and those who believe that the market is efficient and whenever new information comes up the market absorbs it by correcting itself, thus there is no space for prediction.

CHAPTER 3

IDEATION AND PROPOSED SOLUTION

3.1 EMPATHY MAP AND CANVAS
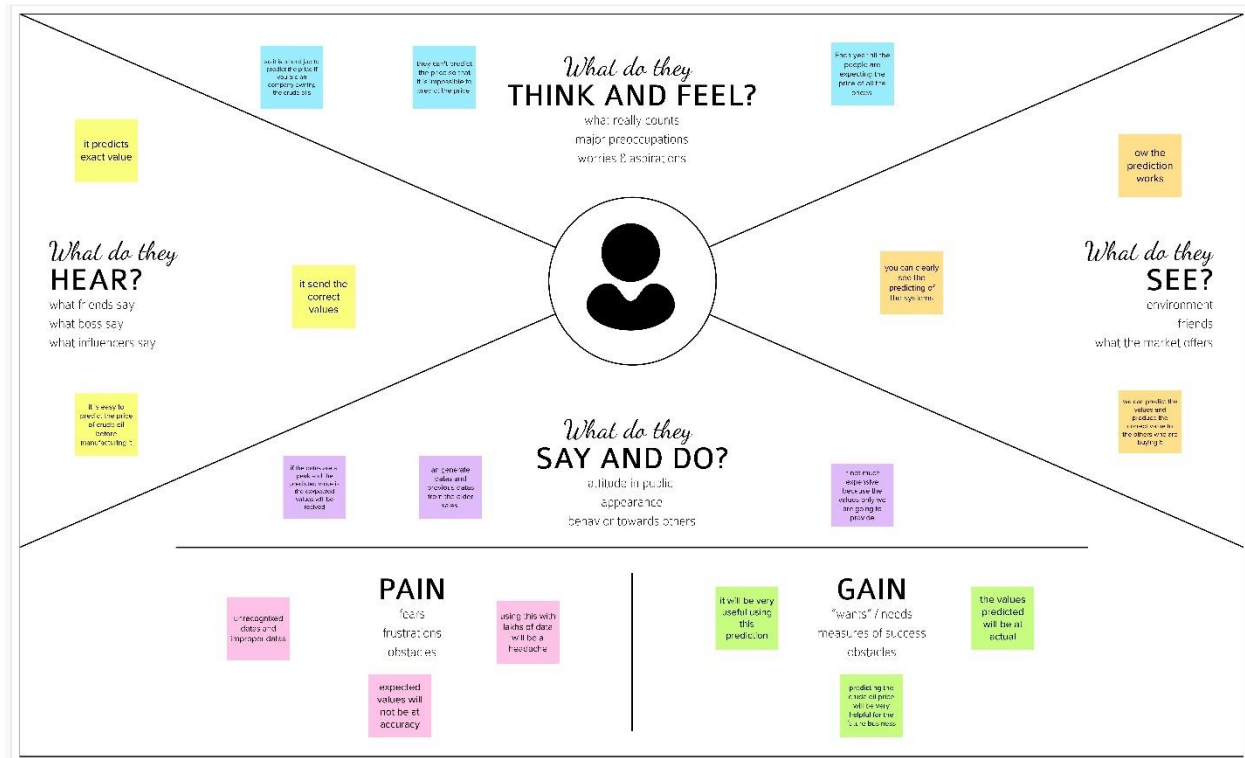
3.2 IDEATION AND BRAINSROMING

Brainstorm & idea prioritization

Use this template in your own brainstorming sessions so your team can unleash their imagination and start shaping concepts even if you're not sitting in the same room.

🕐 **10 minutes** to prepare
⏳ **1 hour** to collaborate
👤 **2-8 people** recommended

Share template feedback

Before you collaborate

A little bit of preparation goes a long way with this session. Here's what you need to do to get going.

🕐 **10 minutes**

A **Team gathering**
Define who should participate in the session and send an invite. Share relevant information or pre-work ahead.

B **Set the goal**
Think about the problem you'll be focusing on solving in the brainstorming session.

C **Learn how to use the facilitation tools**
Use the Facilitation Superpowers to run a happy and productive session.

Open article →

① Define your problem statement

What problem are you trying to solve? Frame your problem as a How Might We statement. This will be the focus of your brainstorm.

🕐 **5 minutes**

PROBLEM
How crude oil price can be predicted?
What are the ways to predict and what are the impacts?

Key rules of brainstorming
To run a smooth and productive session

Stay in topic. Encourage wild ideas.
Defer judgment. Listen to others.
Go for volume. If possible, be visual.

2

Brainstorm

Write down any ideas that come to mind that address your problem statement.

🕐 10 minutes

Mohammed Sufiyan.

Prembabu

Jack Melony

Mathimithran

3

Group ideas

Take turns sharing your ideas while clustering similar or related notes as you go. Once all sticky notes have been grouped, give each cluster a sentence-like label. If a cluster is bigger than six sticky notes, try and see if you and break it up into smaller sub-groups.

🕐 20 minutes

Basic level

- can be predicted using the raw data
- Artificial intelligence can be very helpful in prediction
- Samplings and records are useful for prediction

Advanced level

- Deep learning can be used for analysis
- History of prices can be used for references
- the integration of AI and Machine Learning will be useful
- Python can be used to represent the ideas
- The overall results can be recorded and can be used for prediction
- professionals can be really helpful in gaining knowledge about the use of prices
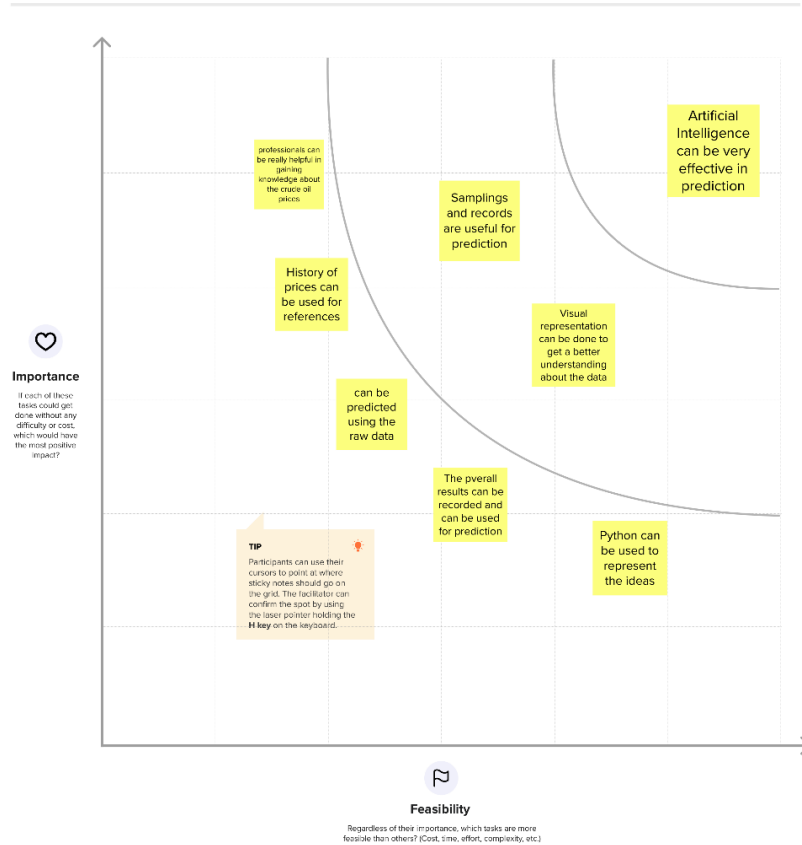- Price prediction can be done by comparing the data from the different time period
- Visual representation can be done to get a better understanding about the data

4

Prioritize

Your team should all be on the same page about what's important moving forward. Place your ideas on this grid to determine which ideas are important and which are feasible.

⏱ 20 minutes

Importance

If each of these tasks could get done without any difficulty or cost, which would have the most positive impact?

professionals can be really helpful in gaining knowledge about the crude oil prices.

Samplings and records are useful for prediction

Artificial Intelligence can be very effective in prediction

History of prices can be used for references

Visual representation can be done to get a better understanding about the data

can be predicted using the raw data

The overall results can be recorded and can be used for prediction

Python can be used to represent the ideas

TIP

Participants can use their cursors to point at where sticky notes should go on the grid. The facilitator can confirm the spot by using the laser pointer holding the **H key** on the keyboard.

Feasibility

Regardless of their importance, which tasks are more feasible than others? (Cost, time, effort, complexity, etc.)

→

After you collaborate

You can export the mural as an image or pdf to share with members of your company who might find it helpful.

Quick add-ons

A **Share the mural**
Share a view link to the mural with stakeholders to keep them in the loop about the outcomes of the session.

B **Export the mural**
Export a copy of the mural as a PNG or PDF to attach to emails, include in slides, or save in your drive.

Keep moving forward

Strategy blueprint
Define the components of a new idea or strategy.
Open the template →

Customer experience journey map
Understand customer needs, motivations, and obstacles for an experience.
Open the template →

Strengths, weaknesses, opportunities & threats
Identify strengths, weaknesses, opportunities, and threats (SWOT) to develop a plan.
Open the template →

🗨 Share template feedback

3.3 PROPOSED SOLUTION

S.No.	Parameter	Description
1.	Problem Statement (Problem to be solved)	As with the erratic changes in supply and demand and also the influence of geopolitics, it is very hard to predict the value of crude oil prices in the global market.
2.	Idea / Solution description	We are going to collect the dataset of the past oil prices with time so that by feeding those to the model and training it and compiling it and when it's achieved the optimal state we can implement it in the web application.
3.	Novelty / Uniqueness	It may be a traditional idea but the implementation of periodic training will have a better effect on it.
4.	Social Impact / Customer Satisfaction	By using the web app customer can gain knowledge of the crude oil price and get benefits financially.
5.	Business Model (Revenue Model)	It will be used by every individual at ease so that they can have an idea of the crude price so, that the use of the crude will be stable in the market
6.	Scalability of the Solution	The idea we proposed it take the input in the periodic and adjust and train through these so, that it will adapt to very different situations.

3.4 PROBLEM SOLUTION FIT

1. CUSTOMER SEGMENT(S) `CS`

Who is your customer?

Crude Oil Based Industries and companies for Business purposes. Handle transportation and storage in the Business.

6. CUSTOMER CONSTRAINTS `CC`

What constraints prevent your customers from taking action or limit their choices of solutions?

The risks and problems are the obstacles for the customers which limits them from proceeding further in the process.

5. AVAILABLE SOLUTIONS `AS`

Which solutions are available to the customers when they face the problem or need to get the job done? What have they tried in the past? What pros & cons do these solutions have?

The frustrations about the results can be avoid by providing a proper timeline and proper planning will be helpful in finishing it in time with the expected output.

2. JOBS-TO-BE-DONE / PROBLEMS `J&P`

Which jobs-to-be-done (or problems) do you address for your customers?

The difficulty in predicting the Crude Oil Price more accurately is one of the major problems

The information to be collected for providing the desired results may be a problem

9. PROBLEM ROOT CAUSE `RC`

What is the real reason that this problem exists? What is the back story behind the need to do this job?

It can both Man-made error or machine error which can sometimes go wrong.
This can cause a problem in proving an accurate or desired result.
This is the main root cause of this issue.

7. BEHAVIOUR `BE`

What does your customer do to address the problem and get the job done?

The problems faced by the customer can be reported in a form of a detailed document.
So that it can be properly addressed by the team and it can rectify.

3. TRIGGERS `TR`

What triggers customers to act?

The business ideas trigger customers for the crude oil price prediction for the benefits

4. EMOTIONS: BEFORE / AFTER `EM`

How do customers feel when they face a problem or a job and afterwards?

If the results are not up to the expected point, it makes them feel frustrated.

10. YOUR SOLUTION `SL`

If you are working on an existing business, write down your current solution first, fill in the canvas, and check how much it fits reality. If you are working on a new business proposition, then keep it blank until you fill in the canvas and come up with a solution that fits within customer limitations, solves a problem and matches customer.

To address this issue, it needs proper attention in carrying out this process for predicting the crude oil price. Both computer-aided prediction and human calculations should be carried out very carefully.

8. CHANNELS of BEHAVIOUR `CH`

8.1 ONLINE
What kind of actions do customers take online? Extract online channels from #7

Discount seekers
Wandering customers
Loyal customers

8.2 OFFLINE
What kind of actions do customers take offline? Extract offline channels from #7 and use them for customer development.

Reliable customers
Trustful customers

12

CHAPTER 4

REQUIREMENT ANALYSIS

4.1 FUNCTIONAL REQUIREMENTS

| FR No.	Functional Requirement (Epic)	Sub Requirement (Story / Sub-Task)
FR-1	User Registration	Registration through Form Registration through Gmail
FR-2	User Confirmation	Confirmation via Email Confirmation via OTP
FR-3	Graph	Showing graph by obtaining the data from the dataset
FR-4	Support	Providing answers for the queries asked by users.
FR-5	News	Information of the oil prices will be updated by admin
FR-6	Notification	Notification will be sent for the users price alert
Fr-7	Database	Information of the User will be stored

4.2 NON -FUNCTIONAL REQUIREMENTS

FR No.	Non-Functional Requirement	Description
NFR-1	**Usability**	It can use by wide variety of client as it is very simple to learn and not complex to proceed.
NFR-2	**Security**	We are using login for the user and the information will be hashed so that it will be very secure to use.
NFR-3	**Reliability**	It will be reliable that it can update with very time period so that the accuracy will be good.
NFR-4	**Performance**	It will be perform fast and secure even at the lower bandwidth.
NFR-5	**Availability**	Prediction will be available for every user but only for premium user news,database and price alert will be alert.
NFR-6	**Scalability**	It is scalable that we are going to use data in kb so that the quite amount of storage is satisfied.

CHAPTER 5

PROJECT DESIGN

5.1 DATAFLOW DIAGRAMS

DATA FLOW DIAGRAM
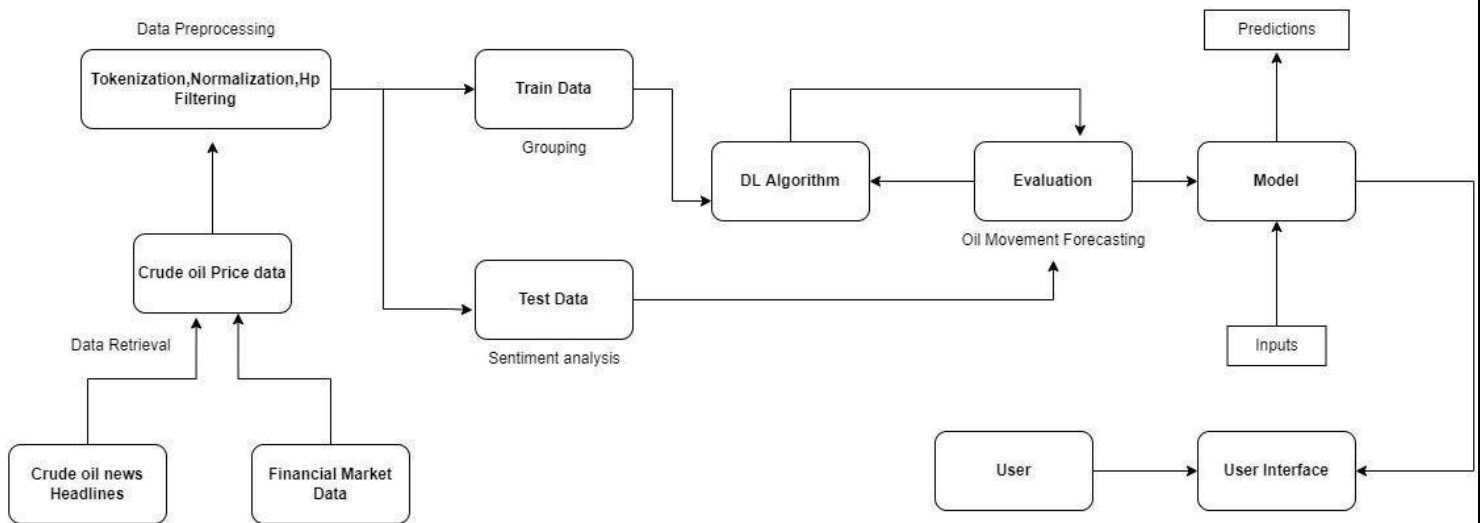
5.2 SOLUTION AND TECHNICAL ARCHITECTURE

Application Characteristics:

S.No	Characteristics	Description	Technology
1.	Open-Source Frameworks-1	Python,	Pandas, flask, numpy, tensorflow
2.	Open-Source Frameworks-2	JavaScript, Angular Js.	App module, component module
3.	Security Implementations	User data will be stored according to CIA model.	End to end encrpytion (SHA-256)
4.	Scalable Architecture	IBM cloud and firebase both used for better performance in storage and authentication.	IBM watson, Firebase, Mysql
5.	Availability	Handle huge requests,avoid DDOS and XSS attack.	Effective coding and restrictive user access based on need
6.	Performance	Handle more than 1000 users to use server at a time.	Flask

Components &Technologies:

S.No	Component	Description	Technology
1.	User Interface	Web application	HTML, CSS, JavaScript ,Angular Js
2.	Application Logic-1	Logic for a process in the Application	Python
3.	Application Logic-2	Logic for a process in the application	IBM Watson Assistant
4.	Database	Data Type, Configurations	MySQL
5.	Cloud Database	Database Service on Cloud	IBM cloud
6.	File Storage	File storage requirements	IBM Block Storage,Local Filesystem
7.	External API-1	Purpose of External API used in the Application	Firebase
8.	Machine Learning Model	Purpose of Machine Learning Model	Recurrent neural network & LSTM
9.	Infrastructure (Server / Cloud)	Application Deployment on Local System / Cloud Local Server Configuration: Cloud Server Configuration:	Local, Firebase.

Technical Architecture

16
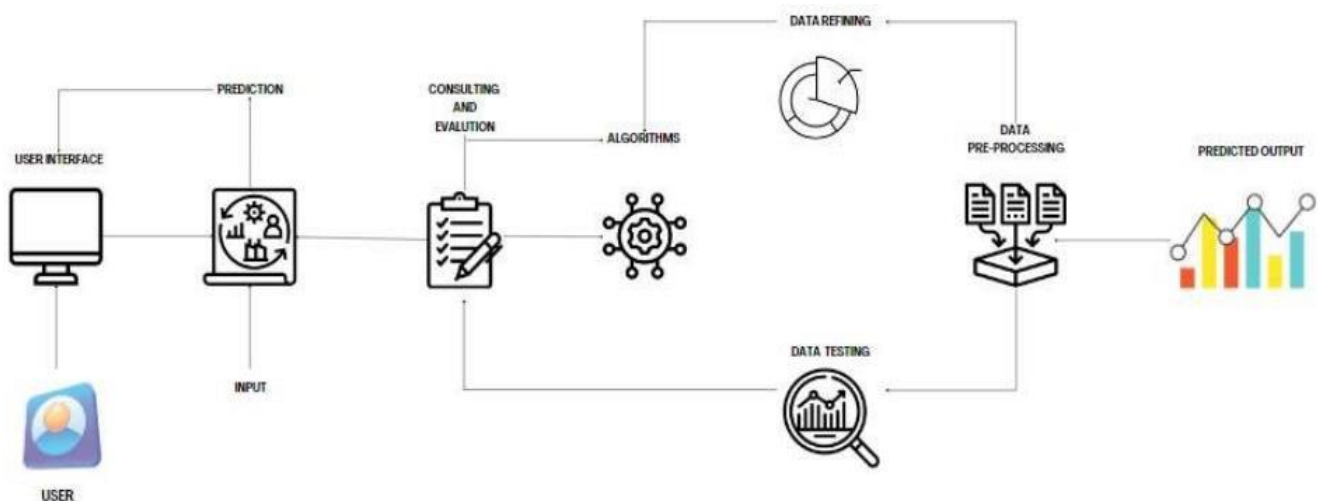
5.3 User stories

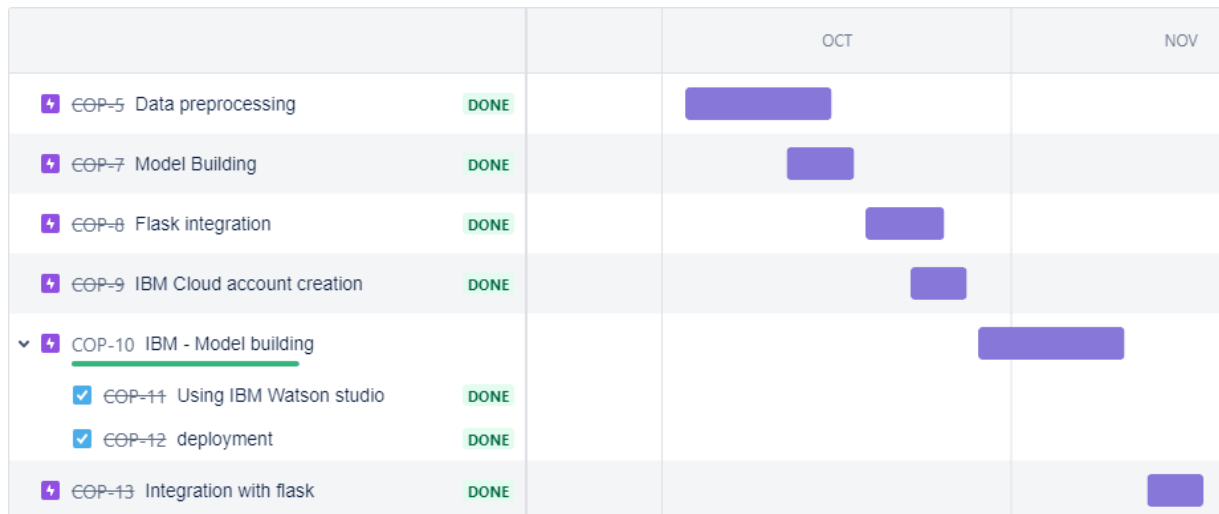| User Type	Functional Requirement (Epic)	User Story Number	User Story/ Task	Acceptance criteria	Priority	Release
Customer (Mobile User)	Registration	USN-1	As a user,I can register for the application by entering my email, password,and confirming my password.	I can access my account/ Displays Line gragh / Bar gragh.	High	Sprint-1
		USN-2	As a user,I will receive confirmation email once I have registered for the application	I can receive confirmation email & click confirm	High	Sprint-1
		USN-3	As a user,I can register for the application through Facebook	I can register & accessthe my Account	Low	Sprint-2
		USN-4	As a user,I can register for the application through Gmail	I can register through already logged in gmail account.	Medium	Sprint-1
	Login	USN-5	As a user,I can log into the application by entering email & password	After registration,I can log in by only email & password.	High	Sprint-1
	Line\Bar gragh		After entering the inputs,the model will display predictions in Line\Bar Gragh Format.	I can get the expected prediction in various formats.	High	Sprint-3
Customer (Web user)	Login	USN-1	As the web user,I can login simply by using Gmail or Facebook account.	Already created gmail can be used for Login.	Medium	Sprint-2
Customer Care Executive	Support		The Customer care service will provide solutions for any FAQ and also provide ChatBot.	I can solve the problems arised by Support.	Low	Sprint-3
Administ rator	News		Admin will give the recent news of Oil Prices.	Provide the recent oil prices.	High	Sprint-4
	Notificatio n		Admin will notify when the oil prices changes.	Notification by Gmail.	High	Sprint-4
	Access Control		Admin can control the access of users.	Access permission for Users.	High	Sprint-4
	Database		Admin can store the details of users.	Stores User details.	High	Sprint-4

17

Chapter 6

PROJECT PLANNING & SCHEDULING

6.1 Sprint Planning & Estimation

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members
Sprint-1	Registration	USN-1	As a user, I can register for the application by entering my email, password, and confirming my password.	10	High	Mohammed Sufiyan F
Sprint-1		USN-2	As a user, I will receive confirmation email once I have registered for the application	10	High	Prembabu C
Sprint-1	Login	USN-3	As a user, I can log into the application by entering email & password.	15	High	Jack Melony G
Sprint-2	Input Necessary Details	USN-4	As a user, I can give Input Details to Predict Likeliness of crude oil	15	High	Mathimithran T
Sprint-2	Data Pre-processing	USN-5	Transform raw data into suitable format for prediction.	15	High	Mohammed Sufiyan F
Sprint-3	Prediction of Crude Oil Price	USN-6	As a user, I can predict Crude oil using machine learning model.	20	High	Prembabu C
Sprint-3		USN-7	As a user, I can get accurate prediction of crude oil	5	Medium	Jack Melony G
Sprint-4	Review	USN-8	As a user, I can give feedback of the application.	20	High	Mathimithran T

6.2. Sprint Delivery Schedule

Sprint	Total Story Points	Duration	Sprint Start Date	Sprint End Date (Planned)	Story Points Completed (as on Planned End Date)	Sprint Release Date (Actual)
Sprint-1	20	6 Days	24 Oct 2022	29 Oct 2022	20	29 Oct 2022
Sprint-2	20	6 Days	31 Oct 2022	05 Nov 2022	20	05 Nov 2022
Sprint-3	20	6 Days	07 Nov 2022	12 Nov 2022	20	12 Nov 2022
Sprint-4	20	6 Days	14 Nov 2022	19 Nov 2022	20	19 Nov 2022

6.3 Reports from JIRA

CHAPTER 7

CODING & SOLUTIONING

7.1 Feature 1


```html
<html>
<head>
<!--    <meta charset="UTF-8">-->
<!--    <meta http-equiv="X-UA-Compatible" content="IE=edge">-->
<!--    <meta name="viewport" content="width=device-width, initial-scale=1.0">-->
    <link href="static/css/index.css" rel="stylesheet">
    <title>Index</title>
</head>
<body>
    <div class="container">
        <li><a href="new.html">Predict</a></li>
        <li><a href="index.html">Home</a></li>
    </div>
    <div class="pageFront">
        <h2>Crude Oil Price Prediction</h2>
        <p>Demand for oil is inelastic, therefore the rise in price is good news for producers because they will see an increa
        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 impor
        oil exporters. The crude oil price movements are subject to diverse influencing factors.</p>
    </div>
</body>
</html>
```
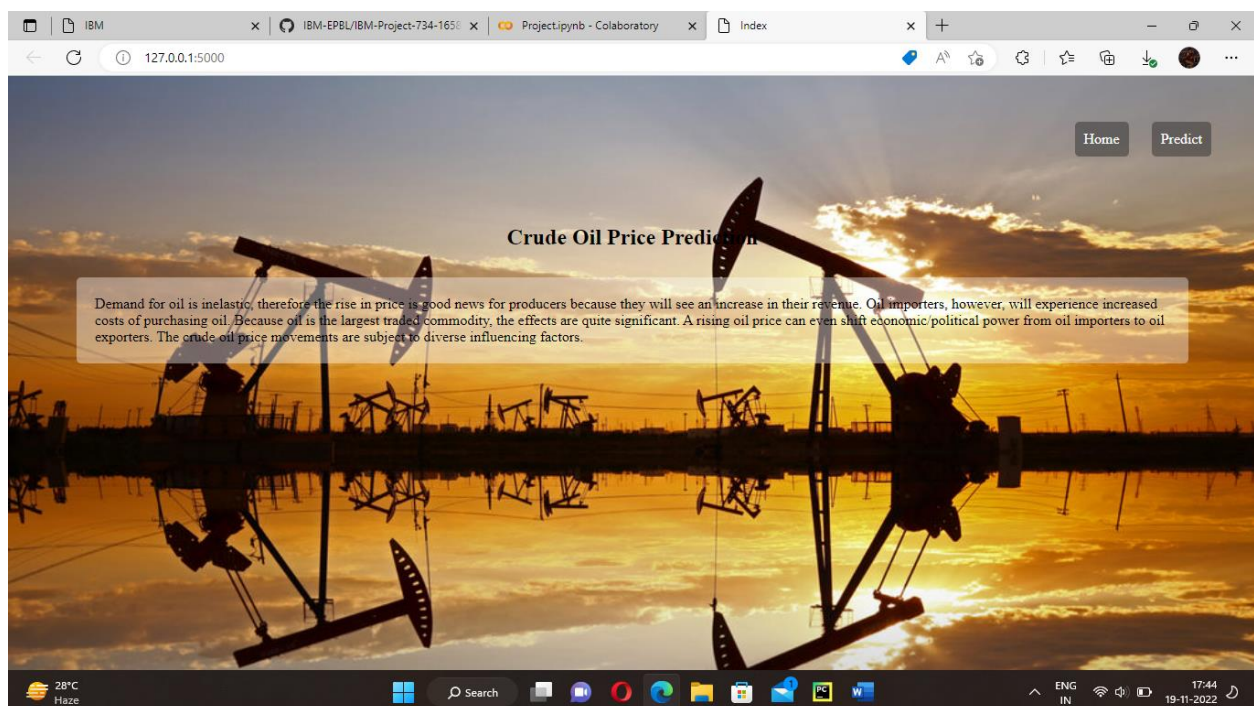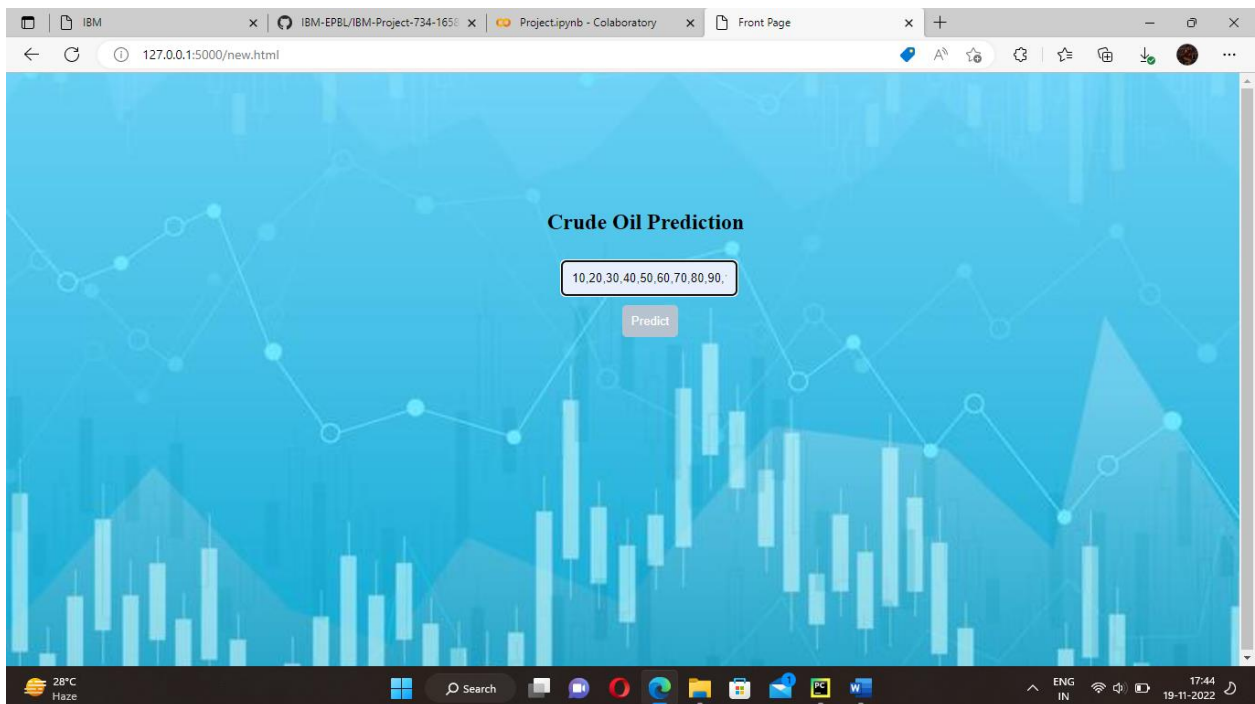
```html
<!DOCTYPE html>
<html lang="en">
<head>

    <link href="static/css/new.css" rel="stylesheet">
    <title>Front Page</title>
</head>
<body>
    <div class="container">
        <h2 class="topic">Crude Oil Prediction</h2><br>
    </div>
    <form action="/login" method="POST">
    <div class="box">
        <input type="text" name="year" class="value" id="value1" placeholder="Enter the crude oil prices for the first 10 days"
        <BUTTON TYPE="submit" class="submit" id="submit1">Predict</BUTTON><br>
    </div>
    </form>

</body>

</html>
```


Crude Oil Prediction

10,20,30,40,50,60,70,80,90,

Predict

7.2 Feature 2

CHAPTER 8

Testing

8.1 Test Cases

Test Case 1:

Input 1: 0.5677, 0.8765, 0.5678, 0.3456, 0.8765, 0.3456, 0.2456, 0.9876, 0.1673, 0.9876

Output: 0.573379635810852
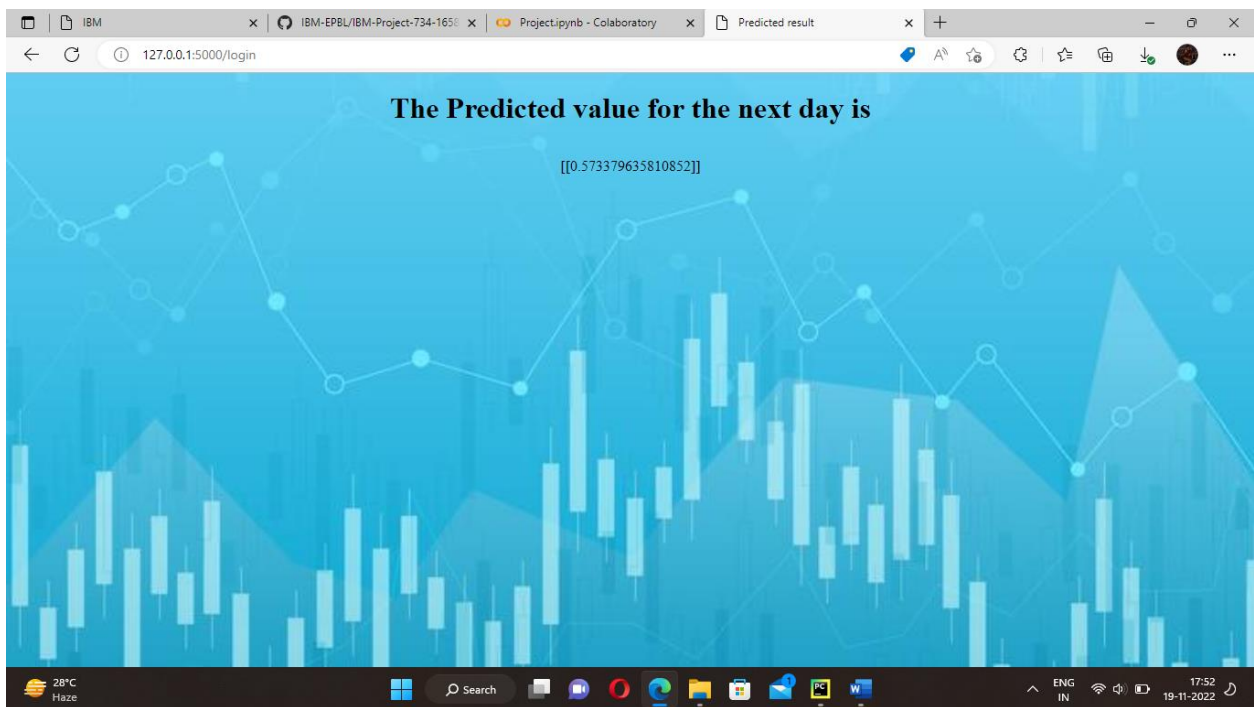
Test Case 2:

Input 2: 0.2314, 0.9876, 0.5643, 0.6566, 0.888, 0.4567, 0.3211, 0.9876, 0.5645, 0.7654

Output: 0.6405302286148071

Test case 3(Pycharm):

Input 3: 0.47567, 0.908889, 0.4783956 , 0.9876, 0.839099 , 0.43827895 , 0.473525 , 0.750590 , 0.567745 , 0.98766

Output: 0.9608331918716431

8.2 User Acceptance Testing

The purpose is to briefly explain the test coverage and open issues of the crude oil priceprediction project at the time of the release to user acceptance testing.

25

Defect Analysis:

Resolution	Severity 1	Severity 2	Severity 3	Severity 4	Subtotal
By Design	3	0	0	0	3
Duplicate	1	0	1	0	2
External	0	0	0	0	0
Fixed	4	0	1	1	6
Not Reproduced	0	0	0	0	0
Skipped	0	0	0	0	0
Won't fix	0	0	0	1	1
Totals	8	0	2	2	12

Test case analysis:

Section	Total Cases	Not Tested	Fail	Pass
ML Model	4	0	0	4
Flask Application	4	0	0	4
IBM Cloud	4	0	0	4
Exception Reporting	2	0	0	2
Final Report Output	4	0	0	4

CHAPTER 9

RESULT

9.1 Performance Metrics:

S.No.	Parameter	Values	Screenshot
1.	Model Summary	It accepts data in the shape of n,1 where n Is the number of inputs and predicts n number of outputs.	```train_predict=scaler.inverse_transform(train_data) test_predict=scaler.inverse_transform(test_data) ### Calculate RMSE performance metrics import math from sklearn.metrics import mean_squared_error math.sqrt(mean_squared_error(train_data,train_predict))``` 29.347830443269938
2.	Accuracy	Training Accuracy – 71.06% Validation Accuracy – 70.98%	```train_predict=scaler.inverse_transform(train_data) test_predict=scaler.inverse_transform(test_data) ### Calculate RMSE performance metrics import math from sklearn.metrics import mean_squared_error math.sqrt(mean_squared_error(train_data,train_predict))``` 29.347830443269938
3.	Confidence Score (Only Yolo Projects)	Class Detected - Confidence Score -	-------

We use different standard performance metrics in the oil price prediction literature for comparing different oil price prediction models. The first metric is Mean Squared Prediction Error (MSPE). MSPE of a prediction model measures the average of the squares of the prediction errors. The prediction error is the difference between the true value and the predicted value. Let $y1, y2, …, yn$ be the true oil prices and $y\hat{}1, y\hat{}2, …, y\hat{}n$ be the predicted oil prices under an oil price prediction model, and then the MSPE of that model is:

$$MSPE = 1n\sum i=1n \ (y\hat{}i - yi) \ 2$$

For comparison purposes, we use the no-change model as the baseline model and express the MSPE of another model as a ratio relative to the MSPE of the no-change model. If the MSPE ratio of a model is less than 1, then the model is more accurate than the no-change model in terms of MSPE.

CHAPTER 10

ADVANTAGES AND DISADVANTAGES

ADVANTAGES:

LSTM models have great advantages in terms of mining the long-term dependence of crude oil price sequence data. Furthermore, LSTM models can automatically search for nonlinear features and complex patterns of crude oil prices, which shows excellent forecasting performance in crude oil price prediction. As a very powerful prediction tool, LSTM has been widely used in prediction-related fields. Therefore, to forecast crude oil price more accurately, we have selected the LSTM model for this study.

The different gates inside LSTM boost its capability for capturing nonlinear relationships for forecasting. Causal factors generally have non-linear impact on demand. When these factors are used as part of the input variable, the LSTM could learn the nonlinear relationship for forecasting.

It is natural that events would impact demand on the day when it is happening as well as the days before and after the event is happening. For example, people would book more days of accommodation to attend a sports event. The LSTM could triage the impact patterns from different categories of events.

- No complexity (web app works to the point)

- Faster response (less latency)

- Can be scaled easily

- Simple user interface

- Light weight

DISADVANTAGES:

They became popular since they solved the issue of gradients disappearing. However, that they are unable to eliminate the problem. The issue lies in that data needs to be moved between cells for its analysis. Furthermore, the cell is becoming extremely complex with the addition of functions (such as the forget gate) that are now part of the picture.

LSTMs are affected by various random weights and behave similarly to neural networks that feed forward. They favour small initialization over large weights.

With the growing technology of data, mining scientists are searching for a system that can store past data for more extended periods of time than LSTMs. The motivation behind the development of such a model is the habit of humans of dividing a particular chunk of information into smaller parts to facilitate recollection.

- Flask can handle only smaller application.
- The user need to enter past 10 days crude oil price values, it may cause discomfort to the end users even though they can get the result immediately.
- Maintenance cost because of Flask.

CHAPTER 11

CONCLUSION

The web app will provide the best service to the end users (mainly INVESTORS) who are expert in investing and just looking for the crude oil price for their benefit as the web app will not let them wait and gives response in no time. Our Web app shows that our model achieves the highest accuracy in terms of both mean squared prediction error and directional accuracy ratio over a variety of forecast time horizons. The model used in our web app is LSTM (Long Short Term Memory). LSTM leads to more successful runs and learns much faster in compare to other algorithms like RTRN, BPTT, and RCC etc. LSTM also solves complex. The complexity to update each weight is reduced to O (1) with LSTM which is an added advantage. The LSTM cell adds long term memory in an even more performant way because it allows even more parameters to be learnt. This makes it the most powerful RNN (Recurrent Neural Network) to do forecasting, especially when you have a longer term trend in your data. IBM Watson provides tool to work collaborative and make work easier with data as well as in training the model. IBM Watson enables one-click deployment with Machine-Learning.

Immediate response:

The end users don't need to wait for longer seconds. The end users will get what they are looking   for in just a second. There is no latency.

Scalability:

As our web app is deployed on IBM cloud, the code can be managed and deployed easily. If the numbers of users are monotonically increased, our web application can be scaled in no time.

30

CHAPTER 12

FUTURE SCOPE

The current market situation, amid the Covid-19 outbreak, is not expected to last for the long- term; however, its long term effect on the markets may be felt for a few years to come, especially with the threat of a recession also coming from this outbreak. This could ripple out and affect the long term oil price forecast, and will need to be taken into consideration. There's also a delta variant spreading that is causing a return of lockdowns in some regions which could once again harm oil prices.

Oil price predictions long term are still vitally important to the oil investing market as the commodity, although quite volatile, is one that is often traded over longer periods of time. Oil is also a commodity that is still in high demand, and is finite, so it is expected to grow in demand over the long terms. Additionally, the prediction of oil in the long term is something that is important to different groups in the industry.

Compared with the ANN and ARIMA model, the average prediction accuracy of the LSTM model was 66.67% (33.33%) and 439587 (673.8) times higher, respectively. So, we can conclude that the LSTM model can improve the forecasting accuracy for both kinds of prices in the short term.

The number of investors in crude oil is keep on increasing as it is still in high demand. When youngsters are interested in investing but when they are in beginning stage surely our web app provide a greater service to make them understand the pattern everyday by giving them the prediction.

As price of crude oil is a complex changing pattern and new price comes every time, it is important to update our model. As we use LSTM model the complexity is very reduced to O(1) the model doesn't take more time to retrain the model with new data, helping us to easily serve for the customers with better accuracy.

CHAPTER 13

APPENDIX

SOURCE CODE

Importing libraries

```
import numpy as np import
pandas as pd import datetime
from pylab import rcParams import
matplotlib.pyplot as plt import warnings
import itertools
import statsmodels.api as sm
from keras.models import Sequential from
keras.layers import Dense
from keras.layers import LSTM from keras.layers
import Dropout
from sklearn.metrics import mean_squared_error
from keras.callbacks import ReduceLROnPlateau, EarlyStopping,
ModelCheckpoint
from sklearn.metrics import mean_squared_error from sklearn.metrics
import mean_absolute_error from tensorflow.keras.models import
load_model import seaborn as sns
sns.set_context("paper", font_scale=1.3) sns.set_style('white')
import math
from sklearn.preprocessing import MinMaxScaler
warnings.filterwarnings("ignore") plt.style.use('fivethirtyeight')
import os
for dirname, _, filenames in os.walk('/kaggle/input'):
        for filename in filenames: print(os.path.join(dirname,
                filename))
```

Importing data

```
dateparse = lambda x: pd.datetime.strptime(x, '%b %d, %Y') from google.colab import
files
uploaded = files.upload()

<IPython.core.display.HTML object>

Saving Crude Oil Prices Daily.xlsx to Crude Oil Prices Daily.xlsx import io
df = pd.read_excel(io.BytesIO(uploaded['Crude Oil Prices
```

```
Daily.xlsx'])) df.head()
df[:10]
```

	Date	Closing	Value
0	1986-01-02		25.56
1	1986-01-03		26.00
2	1986-01-06		26.53
3	1986-01-07		25.85
4	1986-01-08		25.87
5	1986-01-09		26.03
6	1986-01-10		25.65
7	1986-01-13		25.08
8	1986-01-14		24.97
9	1986-01-15		25.18

#Sort dataset by column Date

```
df = df.sort_values('Date')
df = df.groupby('Date')['Closing Value'].sum().reset_index() df.set_index('Date',
inplace=True) df=df.loc[datetime.date(year=2000,month=1,day=1):]

df.head()
```

	Closing Value
Date	
2000-01-04	25.56
2000-01-05	24.65
2000-01-06	24.79
2000-01-07	24.79
2000-01-10	24.71

Data preprocessing

```
def DfInfo(df_initial):
    tab_info = pd.DataFrame(df_initial.dtypes).T.rename(index={0: 'column type'})
    tab_info = tab_info.append(pd.DataFrame(df_initial.isnull().sum()).T.rename(index
={0: 'null values (nb)'}))
    tab_info = tab_info.append(pd.DataFrame(df_initial.isnull().sum()
/ df_initial.shape[0] * 100).T.rename(index={0: 'null values (%)'}))
    return tab_info

DfInfo(df)
```

	Closing Value
column type	float64
null values (nb)	0
null values (%)	0.0

df.index

```
DatetimeIndex(['2000-01-04', '2000-01-05', '2000-01-06', '2000-01-07',
               '2000-01-10', '2000-01-11', '2000-01-12', '2000-01-13',
               '2000-01-14', '2000-01-18',
               ...
               '2018-06-26', '2018-06-27', '2018-06-28', '2018-06-29',
               '2018-07-02', '2018-07-03', '2018-07-04', '2018-07-05',
               '2018-07-06', '2018-07-09'],
              dtype='datetime64[ns]', name='Date', length=4673,
freq=None)
```

```python
y = df['Closing Value'].resample('MS').mean() y.plot(figsize=(15, 6))
plt.show()
```
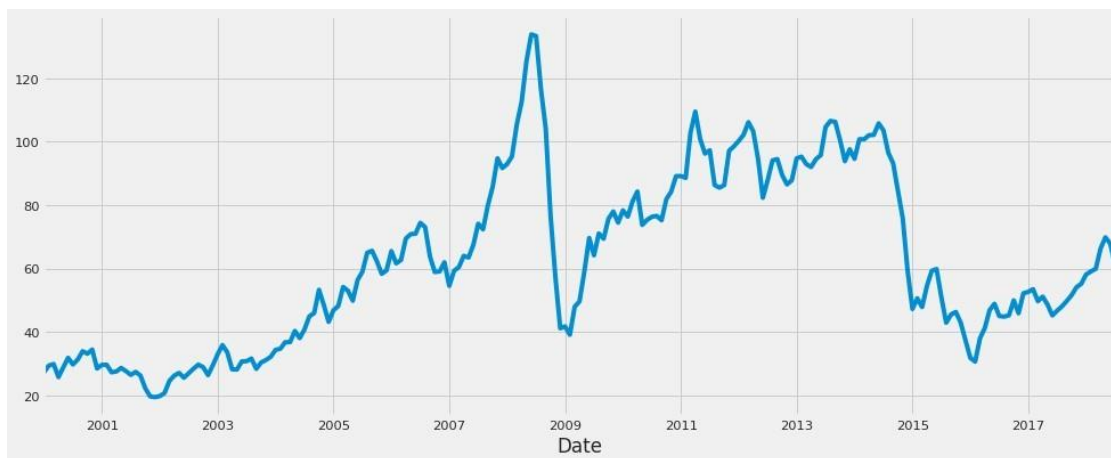


```python
rcParams['figure.figsize'] = 18, 8
decomposition = sm.tsa.seasonal_decompose(y, model='additive') fig = decomposition.plot()
plt.show()
```
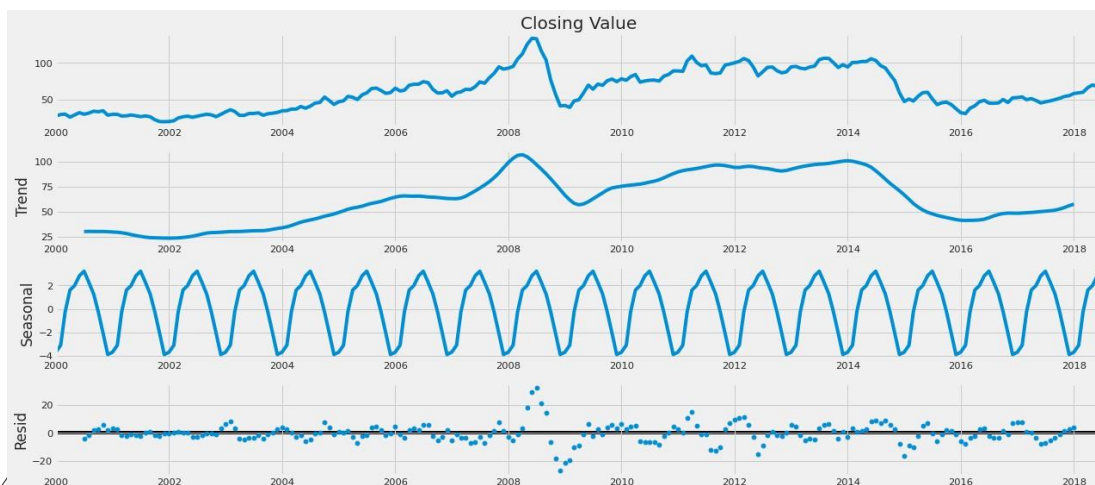
```
sc = MinMaxScaler(feature_range = (0, 1)) df =
sc.fit_transform(df)
```

Training and testing

```
train_size = int(len(df) * 0.70) test_size = len(df)
- train_size
train, test = df[0:train_size, :], df[train_size:len(df), :]

  def create_data_set(_data_set, _look_back=1): data_x, data_y = [],
        []
        for i in range(len(_data_set) - _look_back - 1): a = _data_set[i:(i +
            _look_back), 0] data_x.append(a)
            data_y.append(_data_set[i + _look_back, 0])
        return np.array(data_x), np.array(data_y)

look_back =90
X_train,Y_train,X_test,Ytest = [],[],[],[]
X_train,Y_train=create_data_set(train,look_back)
X_train = np.reshape(X_train, (X_train.shape[0], X_train.shape[1], 1))
X_test,Y_test=create_data_set(test,look_back)
X_test = np.reshape(X_test, (X_test.shape[0], X_test.shape[1], 1))
```

LSTM layer

```
regressor = Sequential()
regressor.add(LSTM(units = 60, return_sequences = True, input_shape = (X_train.shape[1], 1)))
regressor.add(Dropout(0.1))
regressor.add(LSTM(units = 60, return_sequences = True))
regressor.add(Dropout(0.1))
regressor.add(LSTM(units = 60))
regressor.add(Dropout(0.1))
regressor.add(Dense(units = 1))

regressor.compile(optimizer = 'adam', loss = 'mean_squared_error') reduce_lr =
ReduceLROnPlateau(monitor='val_loss',patience=5) history =regressor.fit(X_train, Y_train, epochs
= 20, batch_size = 15,validation_data=(X_test, Y_test), callbacks=[reduce_lr],shuffle=False)

Epoch 1/20
212/212 [==============================] - 31s 119ms/step - loss:
0.0049 - val_loss: 0.0220 - lr: 0.0010 Epoch 2/20
212/212 [==============================] - 24s 111ms/step - loss:
0.0112 - val_loss: 0.0487 - lr: 0.0010 Epoch 3/20
212/212 [==============================] - 23s 111ms/step - loss:
```

0.0124 - val_loss: 0.0549 - lr: 0.0010 Epoch 4/20
212/212 [==============================] - 24s 111ms/step - loss: 0.0164 - val_loss: 0.0484 - lr: 0.0010 Epoch 5/20
212/212 [==============================] - 23s 111ms/step - loss: 0.0199 - val_loss: 0.0546 - lr: 0.0010 Epoch 6/20
212/212 [==============================] - 23s 110ms/step - loss: 0.0179 - val_loss: 0.0516 - lr: 0.0010 Epoch 7/20
212/212 [==============================] - 24s 111ms/step - loss: 0.0203 - val_loss: 0.0034 - lr: 1.0000e-04 Epoch 8/20
212/212 [==============================] - 23s 111ms/step - loss: 0.0034 - val_loss: 0.0027 - lr: 1.0000e-04 Epoch 9/20
212/212 [==============================] - 23s 110ms/step - loss: 0.0026 - val_loss: 0.0021 - lr: 1.0000e-04 Epoch 10/20
212/212 [==============================] - 23s 110ms/step - loss: 0.0023 - val_loss: 0.0018 - lr: 1.0000e-04 Epoch 11/20
212/212 [==============================] - 23s 110ms/step - loss: 0.0019 - val_loss: 0.0018 - lr: 1.0000e-04 Epoch 12/20
212/212 [==============================] - 23s 111ms/step - loss: 0.0016 - val_loss: 0.0016 - lr: 1.0000e-04 Epoch 13/20
212/212 [==============================] - 24s 112ms/step - loss: 0.0014 - val_loss: 0.0015 - lr: 1.0000e-04 Epoch 14/20
212/212 [==============================] - 24s 113ms/step - loss: 0.0012 - val_loss: 0.0014 - lr: 1.0000e-04 Epoch 15/20
212/212 [==============================] - 24s 113ms/step - loss: 0.0011 - val_loss: 0.0013 - lr: 1.0000e-04 Epoch 16/20
212/212 [==============================] - 24s 115ms/step - loss: 0.0010 - val_loss: 0.0013 - lr: 1.0000e-04 Epoch 17/20
212/212 [==============================] - 24s 112ms/step - loss: 0.0010 - val_loss: 0.0013 - lr: 1.0000e-04 Epoch 18/20
212/212 [==============================] - 24s 113ms/step - loss: 0.0010 - val_loss: 0.0014 - lr: 1.0000e-04 Epoch 19/20
212/212 [==============================] - 9.6307e-04 - val_loss: 0.0014 - lr: 1.0000e-04 Epoch 20/20

212/212 [==============================] - 24s 112ms/step - loss:
9.6895e-04 - val_loss: 0.0013 - lr: 1.0000e-04

Model training

train_predict = regressor.predict(X_train) test_predict =
regressor.predict(X_test)

100/100 [==============================] - 3s 35ms/step
41/41 [==============================] - 1s 35ms/step

train_predict = sc.inverse_transform(train_predict) Y_train =
sc.inverse_transform([Y_train]) test_predict =
sc.inverse_transform(test_predict) Y_test = sc.inverse_transform([Y_test])

Prediction

print('Train Mean Absolute Error:', mean_absolute_error(Y_train[0], train_predict[:,0]))
print('Train Root Mean Squared Error:',np.sqrt(mean_squared_error(Y_train[0], train_predict[:,0])))
print('Test Mean Absolute Error:', mean_absolute_error(Y_test[0], test_predict[:,0]))
print('Test Root Mean Squared Error:',np.sqrt(mean_squared_error(Y_test[0], test_predict[:,0])))
plt.figure(figsize=(8,4))
plt.plot(history.history['loss'], label='Train Loss') plt.plot(history.history['val_loss'],
label='Test Loss') plt.title('model loss')
plt.ylabel('loss') plt.xlabel('epochs')
plt.legend(loc='upper right') plt.show();

Train Mean Absolute Error: 2.42441411734527
Train Root Mean Squared Error: 3.3282435008105846 Test Mean
Absolute Error: 2.3375842822880166
Test Root Mean Squared Error: 5.285524069216685
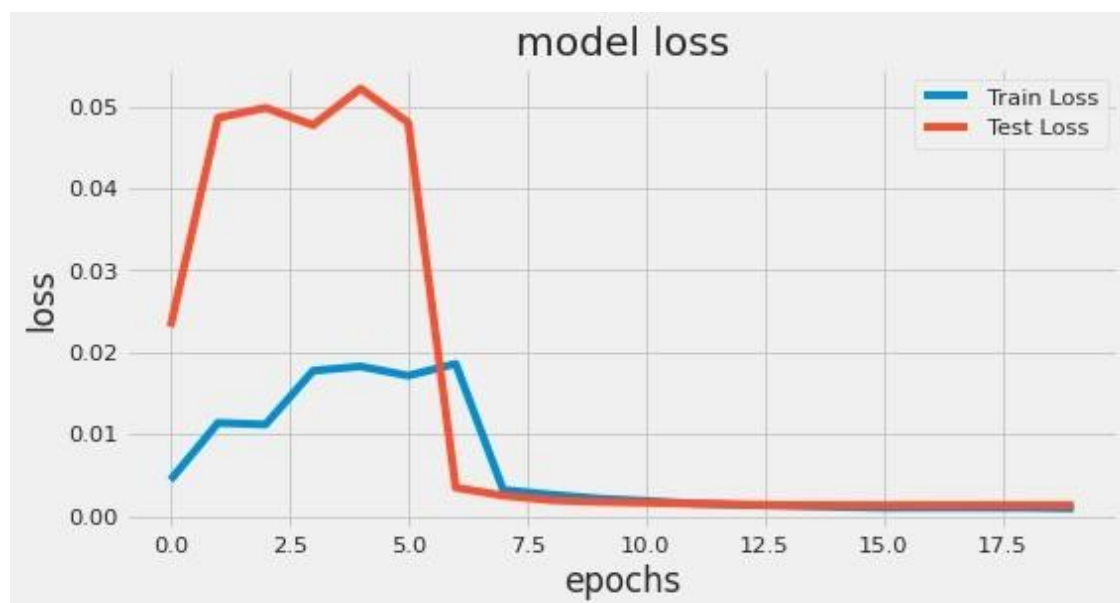
```
aa=[x for x in range(180)]
plt.figure(figsize=(8,4))
plt.plot(aa, Y_test[0][:180], marker='.', label="actual") plt.plot(aa, test_predict[:,0][:180], 'r',
label="prediction") plt.tight_layout()
sns.despine(top=True) plt.subplots_adjust(left=0.07) plt.ylabel('Price', size=15) plt.xlabel('Time step',
size=15) plt.legend(fontsize=15) plt.show();
```
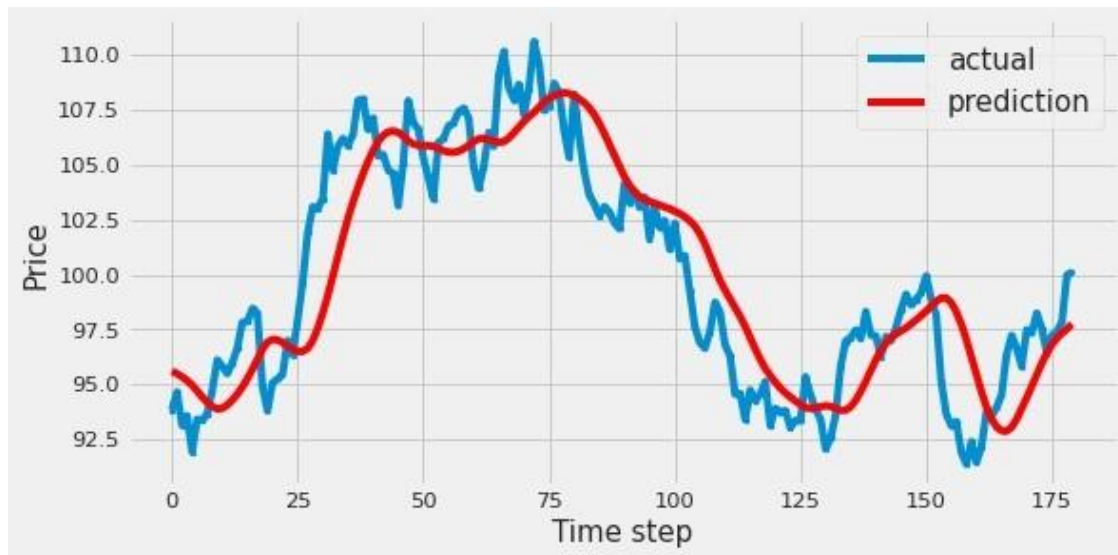

Flask integration:

```
#
app.py
file
        import numpy as np
        from flask import Flask, render_template, request
        from tensorflow.keras.models import load_model
        import os
        app = Flask(__name__)
        model = load_model('data_final.h5', )
        @app.route('/')
        def home():
            return render_template("index.html")
        @app.route('/index.html')
        def home1():
            return render_template("index.html")
        @app.route('/new.html')
```

```python
        def home2():
            return render_template("new.html")
        @app.route('/login',methods=['POST','GET'])
        def login():
         if request.method == 'POST':
            x_input=str(request.form['year'])
            x_input=x_input.split(',')
            print(x_input)
            for i in range(0, len(x_input)):
                x_input[i]=float(x_input[i])
            print(x_input)
            x_input=np.array(x_input).reshape(1, -1)
            temp_input=list(x_input)
            temp_input=temp_input[0].tolist()
            lst_output=[]
            n_steps=10
            i=0
            while(i<1):
             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
            print(lst_output)
            return render_template("result.html",result=str(lst_output))
        if __name__=='__main__':
          app.run(debug=True, port=5000)
```

HTML FILE :

39

INDEX.HTML

```html
<html>
        <head>
        <!--    <meta charset="UTF-8">-->
        <!--    <meta http-equiv="X-UA-Compatible" content="IE=edge">-->
        <!--    <meta name="viewport" content="width=device-width, initial-scale=1.0">-->
            <link href="static/css/index.css" rel="stylesheet">
            <title>Index</title>
        </head>
        <body>
            <div class="container">
                <li><a href="new.html">Predict</a></li>
                <li><a href="index.html">Home</a></li>
            </div>
            <div class="pageFront">
                <h2>Crude Oil Price Prediction</h2>
                <p>Demand for oil 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.</p>
            </div>
        </body>
        </html>
```

INDEX.CSS

```css
*{
        margin: 0;
        padding: 0;
    }
    body{
        background-image: url(crude.jpg);
        background-color: #ffffff;
        height: 500px;
        background-position: center;
        background-repeat: no-repeat;
        background-size: cover;
    }
```

40

```css
.container{
    padding: 20px;
    margin: 20px;
    display: flex;
    gap: 5px;
    flex-direction: row-reverse;
}
.container a{
    text-decoration: none;
    color: #fff;

}
.container li{
    list-style-type: none;
    margin: 10px;
    background-color: #36363689;
    padding: 10px;
    border: 0px solid;
    color: #fff;
    border-radius: 5px;
}
.pageFront{
    margin: 20px;
    padding: 25px;
    color:
}
.pageFront h2{
    text-align: center;
    color:
}
.pageFront p{

    margin: 30px;
    background-color: rgba(255, 255, 255, 0.4);
    padding: 20px;
    border: 0px solid;
    color: rgba(0, 0, 0, 1);
    border-radius: 5px;
}
```

NEW.HTML

```html
<!DOCTYPE
html>
```
41

```html
<html lang="en">
<head>
    <link href="static/css/new.css" rel="stylesheet">
    <title>Front Page</title>
</head>
<body>
    <div class="container">
        <h2 class="topic">Crude Oil Prediction</h2><br>
    </div>
    <form action="/login" method="POST">
    <div class="box">
        <input type="text" name="year" class="value" id="value1"
placeholder="Enter the crude oil prices for the first 10 days"
multiple></input><br>
        <BUTTON TYPE="submit" class="submit" id="submit1">Predict</BUTTON><br>
    </div>
    </form>
</body>
</html>
```

INDEX.CSS

```css
*{
    margin: 0;
    padding: 0;
}
body{
    background-image: url(predict.jpg);
    background-color: #fbfbfb;
    background-position: center;
    background-repeat: no-repeat;
    background-size: cover;
    height: 500px;
}
.container{
    text-align: center;
    margin-top: 150px;
    margin-left: 50px;
}
h4{
    display: flex;
    align-items: center;
    justify-content: center;
    width: 30%;
    text-align: center;
    margin: 10px;
```

42

```css
            background-color: #93919189;
            padding: 15px;
            border: 0px solid;
            color: #fff;
            border-radius: 10px;
        }
        .box{
            display: flex-column;
            align-items: center;
            justify-content: center;
            margin-left: 45%;
        }
        #submit1{
            padding: 10px;
            text-align: center;
            color: #fff;
            background-color: #B7C4CF;
            border:none;
            border-radius: 5px;
            margin-top: 10px;
            margin-left: 9%;
        }
        #value1{
            padding: 10px;
            text-align: center;
            color: #fff;
            background-color: #EAEAEA;
            border-radius: 5px;
            margin-top: 10px;
        }
```

RESULT.HTML

```html
<!DOCTYPE
html>
        <html lang="en">
        <head>
            <meta charset="UTF-8">
            <title>Predicted result</title>
            <link href="static/css/result.css" rel="stylesheet">
        </head>
        <body class="result">
            <h1>The Predicted value for the next day is</h1><br>  
            <p>{{result}}</p>
```

43

```
        </body>
        </html>
```

RESULT.CSS

```css
*{
        margin:0;
        padding:o;
      }
      body{
        background-image: url(predict.jpg);
          background-color: #fbfbfb;
          background-position: center;
          background-repeat: no-repeat;
          background-size: cover;
          height: 500px;
      }
      .result{
        display: flex-column;
        padding:20px;
        text-align: center;
      }
```

TRAIN THE MODEL IN IBM CLOUD:

IBM Deployment

!pip install watson-machine-learning-client

Looking in indexes: https://pypi.org/simple, https://us- python.pkg.dev/colab-wheels/public/simple/
Collecting watson-machine-learning-client
    Downloading watson_machine_learning_client-1.0.391-py3-none-any.whl (538 kB)
ent already satisfied: urllib3 in /usr/local/lib/python3.7/dist- packages (from watson-machine-learning-client) (1.24.3) Requirement already satisfied: tabulate in
/usr/local/lib/python3.7/dist-packages (from watson-machine-learning- client) (0.8.10)
Collecting lomond
    Downloading lomond-0.3.3-py2.py3-none-any.whl (35 kB) Requirement already
satisfied: certifi in
/usr/local/lib/python3.7/dist-packages (from watson-machine-learning- client) (2022.9.24)
Collecting boto3
    Downloading boto3-1.26.7-py3-none-any.whl (132 kB)

44

ent already satisfied: requests in /usr/local/lib/python3.7/dist- packages (from watson-machine-learning-client) (2.23.0) Requirement already satisfied: pandas in /usr/local/lib/python3.7/dist-packages (from watson-machine-learning- client) (1.3.5)
Requirement already satisfied: tqdm in /usr/local/lib/python3.7/dist- packages (from watson-machine-learning-client) (4.64.1)
Collecting ibm-cos-sdk
   Downloading ibm-cos-sdk-2.12.0.tar.gz (55 kB)
espath<2.0.0,>=0.7.1
   Downloading jmespath-1.0.1-py3-none-any.whl (20 kB) Collecting s3transfer<0.7.0,>=0.6.0
   Downloading s3transfer-0.6.0-py3-none-any.whl (79 kB) ent already satisfied: python-dateutil<3.0.0,>=2.1 in /usr/local/lib/python3.7/dist-packages (from botocore<1.30.0,>=1.29.7->boto3->watson-machine-learning-client) (2.8.2) Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.7/dist-packages (from python- dateutil<3.0.0,>=2.1->botocore<1.30.0,>=1.29.7->boto3->watson-machine- learning-client) (1.15.0)
Collecting ibm-cos-sdk-core==2.12.0
    Downloading ibm-cos-sdk-core-2.12.0.tar.gz (956 kB)
-cos-sdk-s3transfer==2.12.0
   Downloading ibm-cos-sdk-s3transfer-2.12.0.tar.gz (135 kB) espath<2.0.0,>=0.7.1
   Downloading jmespath-0.10.0-py2.py3-none-any.whl (24 kB) Collecting requests

45

Downloading requests-2.28.1-py3-none-any.whl (62 kB)
ent already satisfied: idna<4,>=2.5 in /usr/local/lib/python3.7/dist- packages (from requests->watson-machine-learning-client) (2.10) Requirement already satisfied: charset-normalizer<3,>=2 in /usr/local/lib/python3.7/dist-packages (from requests->watson-machine- learning-client) (2.1.1)
Requirement already satisfied: pytz>=2017.3 in
/usr/local/lib/python3.7/dist-packages (from pandas->watson-machine- learning-client) (2022.6)
Requirement already satisfied: numpy>=1.17.3 in
/usr/local/lib/python3.7/dist-packages (from pandas->watson-machine- learning-client) (1.21.6)
Building wheels for collected packages: ibm-cos-sdk, ibm-cos-sdk-core, ibm-cos-sdk-s3transfer
    Building wheel for ibm-cos-sdk (setup.py) ... -cos-sdk: filename=ibm_cos_sdk-2.12.0-py3-none-any.whl size=73931 sha256=40521eb23c69343f6cb0ff09ca1de338cd0f97721f6a23c547e435e1c31aaf8 d
      Stored in directory:
/root/.cache/pip/wheels/ec/94/29/2b57327cf00664b6614304f7958abd29d77ea 0e5bbece2ea57
    Building wheel for ibm-cos-sdk-core (setup.py) ... -cos-sdk-core: filename=ibm_cos_sdk_core-2.12.0-py3-none-any.whl size=562962
sha256=17b57b548ee5e5cdac2ad1380f3c82f6f242f1a26bac985f3ef67c0294305dd 0
      Stored in directory:
/root/.cache/pip/wheels/64/56/fb/5cd6f4f40406c828a5289b95b2752a4d142a9 afb359244ed8d
    Building wheel for ibm-cos-sdk-s3transfer (setup.py) ... -cos-sdk- s3transfer:
filename=ibm_cos_sdk_s3transfer-2.12.0-py3-none-any.whl size=89778
sha256=9a6dcd72513363e86b5542b137def2c8a591af72c8d28b0514d240e78826394 d
      Stored in directory:
/root/.cache/pip/wheels/57/79/6a/ffe3370ed7ebc00604f9f76766e1e0348dcdc ad2b2e32df9e1
Successfully built ibm-cos-sdk ibm-cos-sdk-core ibm-cos-sdk-s3transfer Installing collected packages: urllib3, requests, jmespath, ibm-cos- sdk-core, botocore, s3transfer, ibm-cos-sdk-s3transfer, lomond, ibm-cos-sdk, boto3, watson-machine-learning-client
    Attempting uninstall: urllib3
      Found existing installation: urllib3 1.24.3 Uninstalling urllib3-
      1.24.3:
        Successfully uninstalled urllib3-1.24.3 Attempting uninstall:
    requests
      Found existing installation: requests 2.23.0 Uninstalling requests-
      2.23.0:
        Successfully uninstalled requests-2.23.0
Successfully installed boto3-1.26.7 botocore-1.29.7 ibm-cos-sdk-2.12.0 ibm-cos-sdk-core-2.12.0 ibm-cos-sdk-s3transfer-2.12.0 jmespath-0.10.0

lomond-0.3.3 requests-2.28.1 s3transfer-0.6.0 urllib3-1.26.12 watson- machine-learning-client-1.0.391

{"pip_warning":{"packages":["requests","urllib3"]}}

!pip install ibm_watson_machine_learning

Looking in indexes: https://pypi.org/simple, https://us- python.pkg.dev/colab-wheels/public/simple/
Collecting ibm_watson_machine_learning
    Downloading ibm_watson_machine_learning-1.0.257-py3-none-any.whl (1.8 MB)
ent already satisfied: pandas<1.5.0,>=0.24.2 in
/usr/local/lib/python3.7/dist-packages (from
ibm_watson_machine_learning) (1.3.5) Requirement already
satisfied: tabulate in
/usr/local/lib/python3.7/dist-packages (from
ibm_watson_machine_learning) (0.8.10) Requirement already
satisfied: requests in
/usr/local/lib/python3.7/dist-packages (from
ibm_watson_machine_learning) (2.28.1) Requirement already
satisfied: lomond in
/usr/local/lib/python3.7/dist-packages (from
ibm_watson_machine_learning) (0.3.3) Requirement already
satisfied: certifi in
/usr/local/lib/python3.7/dist-packages (from
ibm_watson_machine_learning) (2022.9.24) Requirement already
satisfied: urllib3 in
/usr/local/lib/python3.7/dist-packages (from
ibm_watson_machine_learning) (1.26.12) Collecting ibm-cos-sdk==2.7.*
    Downloading ibm-cos-sdk-2.7.0.tar.gz (51 kB) ent already
satisfied: importlib-metadata in
/usr/local/lib/python3.7/dist-packages (from
ibm_watson_machine_learning) (4.13.0) Requirement already
satisfied: packaging in
/usr/local/lib/python3.7/dist-packages (from
ibm_watson_machine_learning) (21.3) Collecting ibm-cos-sdk-core==2.7.0
    Downloading ibm-cos-sdk-core-2.7.0.tar.gz (824 kB)
-cos-sdk-s3transfer==2.7.0
    Downloading ibm-cos-sdk-s3transfer-2.7.0.tar.gz (133 kB) ent already satisfied:
jmespath<1.0.0,>=0.7.1 in
/usr/local/lib/python3.7/dist-packages (from ibm-cos-sdk==2.7.*-
>ibm_watson_machine_learning) (0.10.0) Collecting
docutils<0.16,>=0.10
    Downloading docutils-0.15.2-py3-none-any.whl (547 kB) ent already satisfied:
python-dateutil<3.0.0,>=2.1 in
/usr/local/lib/python3.7/dist-packages (from ibm-cos-sdk-core==2.7.0-
>ibm-cos-sdk==2.7.*->ibm_watson_machine_learning) (2.8.2) Requirement already
satisfied: numpy>=1.17.3 in

/usr/local/lib/python3.7/dist-packages (from pandas<1.5.0,>=0.24.2->ibm_watson_machine_learning) (1.21.6) Requirement already satisfied: pytz>=2017.3 in /usr/local/lib/python3.7/dist-packages (from pandas<1.5.0,>=0.24.2->ibm_watson_machine_learning) (2022.6) Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.7/dist-packages (from python- dateutil<3.0.0,>=2.1->ibm-cos-sdk-core==2.7.0->ibm-cos-sdk==2.7.*->ibm_watson_machine_learning) (1.15.0)

Requirement already satisfied: charset-normalizer<3,>=2 in /usr/local/lib/python3.7/dist-packages (from requests->ibm_watson_machine_learning) (2.1.1) Requirement already satisfied: idna<4,>=2.5 in /usr/local/lib/python3.7/dist-packages (from requests->ibm_watson_machine_learning) (2.10)

Requirement already satisfied: typing-extensions>=3.6.4 in /usr/local/lib/python3.7/dist-packages (from importlib-metadata->ibm_watson_machine_learning) (4.1.1) Requirement already satisfied: zipp>=0.5 in /usr/local/lib/python3.7/dist-packages (from importlib-metadata->ibm_watson_machine_learning) (3.10.0)

Requirement already satisfied: pyparsing!=3.0.5,>=2.0.2 in /usr/local/lib/python3.7/dist-packages (from packaging->ibm_watson_machine_learning) (3.0.9)

Building wheels for collected packages: ibm-cos-sdk, ibm-cos-sdk-core, ibm-cos-sdk-s3transfer

Building wheel for ibm-cos-sdk (setup.py) ... -cos-sdk: filename=ibm_cos_sdk-2.7.0-py2.py3-none-any.whl size=72563 sha256=883aff6a2eb4f64d9725c3360d1fd6aef738ae8d47e4b1dc28c821f5addd01f 2

Stored in directory: /root/.cache/pip/wheels/47/22/bf/e1154ff0f5de93cc477acd0ca69abfbb8b799 c5b28a66b44c2

Building wheel for ibm-cos-sdk-core (setup.py) ... -cos-sdk-core: filename=ibm_cos_sdk_core-2.7.0-py2.py3-none-any.whl size=501013 sha256=e2a695e1a2fbdf2b3a09b1fd4ce25506bd957e6ee42589f47c7a68d37e82edb e

Stored in directory: /root/.cache/pip/wheels/6c/a2/e4/c16d02f809a3ea998e17cfd02c13369281f3d 232aaf5902c19

Building wheel for ibm-cos-sdk-s3transfer (setup.py) ... -cos-sdk- s3transfer: filename=ibm_cos_sdk_s3transfer-2.7.0-py2.py3-none-any.whl size=88622 sha256=60711740f2411fa7b564df2dd6ac2f326bcb75aacea1807fde89935938e80ce 4

Stored in directory: /root/.cache/pip/wheels/5f/b7/14/fbe02bc1ef1af890650c7e51743d1c8389085 2e598d164b9da

Successfully built ibm-cos-sdk ibm-cos-sdk-core ibm-cos-sdk-s3transfer Installing collected packages: docutils, ibm-cos-sdk-core, ibm-cos-

48

sdk-s3transfer, ibm-cos-sdk, ibm-watson-machine-learning Attempting uninstall:
    docutils
        Found existing installation: docutils 0.17.1 Uninstalling docutils-
        0.17.1:
            Successfully uninstalled docutils-0.17.1 Attempting
    uninstall: ibm-cos-sdk-core
        Found existing installation: ibm-cos-sdk-core 2.12.0 Uninstalling ibm-cos-
        sdk-core-2.12.0:
            Successfully uninstalled ibm-cos-sdk-core-2.12.0 Attempting uninstall:
    ibm-cos-sdk-s3transfer
        Found existing installation: ibm-cos-sdk-s3transfer 2.12.0 Uninstalling ibm-cos-sdk-
        s3transfer-2.12.0:
            Successfully uninstalled ibm-cos-sdk-s3transfer-2.12.0 Attempting uninstall:
    ibm-cos-sdk
        Found existing installation: ibm-cos-sdk 2.12.0 Uninstalling ibm-cos-
        sdk-2.12.0:
            Successfully uninstalled ibm-cos-sdk-2.12.0
Successfully installed docutils-0.15.2 ibm-cos-sdk-2.7.0 ibm-cos-sdk- core-2.7.0 ibm-cos-sdk-s3transfer-
2.7.0 ibm-watson-machine-learning- 1.0.257

from ibm_watson_machine_learning import APIClient

```
wml_credentials = {
        "url":"https://eu-gb.ml.cloud.ibm.com",
        "apikey":"OegDZFbgmd2nJVPQW7buYfvlXEQRhY64_lkDLz_kHpLF"
}
```

client = APIClient(wml_credentials)

Python 3.7 and 3.8 frameworks are deprecated and will be removed in a future release. Use Python 3.9
framework instead.

client

<ibm_watson_machine_learning.client.APIClient at 0x7f8a6bc7a250> client.spaces.get_details()

{'resources': [{'entity': {'compute': [{'crn': 'crn:v1:bluemix:public:pm-20:eu-
gb:a/223be9e216084bc6831c3b6c556758f9: 756bb1ab-f9dd-48ea-bf89-de831b79e2ac::',
        'guid': '756bb1ab-f9dd-48ea-bf89-de831b79e2ac', 'name': 'Watson
        Machine Learning-nr',
        'type': 'machine_learning'}], 'description': '',
    'name': 'crude oil',
    'scope': {'bss_account_id': '223be9e216084bc6831c3b6c556758f9'}, 'stage': {'production':
    False},
    'status': {'state': 'active'},
    'storage': {'properties': {'bucket_name': '4ff2d233-098c-4391-

8449-57d3b94013be',
             'bucket_region': 'eu-gb-standard', 'credentials': {'admin':
             {'access_key_id':
'0e38cf6e31944f938daa9652d79eb602',
                 'api_key': 'AE90AJ0_-7FpZ7LGtI5WHTAxiHjDDh7pnJ1VeWjTlnL5',
                 'secret_access_key':
  '2ea791ab1804e89f786e59b4b5a7835ffbd7fffad6e00aa2', 'service_id': 'ServiceId-
                 59cf7746-ff50-4ca8-89d3-
2cf2e5d7f45c'},
             'editor': {'access_key_id': '074aa8d8552d4c2e8e5f7911eccff9fc', 'api_key':
                 'NRIvdwOcdoHLA4vcmrqDJYrGlZOrbKryOTenSBKX5GIo', 'resource_key_crn':
                 'crn:v1:bluemix:public:cloud-object-
storage:global:a/223be9e216084bc6831c3b6c556758f9:f11a9507-df7d-41fc- 8b4e-4b913e22d2e3::',
             'secret_access_key':
'0d93d72466f97abf01611c2e37e2a1e1e7f0db4eb0f5fe4b',
             'service_id': 'ServiceId-a1e5087b-ec41-42c3-af4a- 212d1d5e4464'},
             'viewer': {'access_key_id': 'e2ab1c2855fc48fab2086f196493a241', 'api_key':
                 'cAyzIXrDLUmfVyyQpjNb-GB0sg1IPTspM24A04TfpWk9', 'resource_key_crn':
                 'crn:v1:bluemix:public:cloud-object-
storage:global:a/223be9e216084bc6831c3b6c556758f9:f11a9507-df7d-41fc- 8b4e-4b913e22d2e3::',
             'secret_access_key':
'7de3c08ec568240c79cde117ba40d04282b31ac9c91c06af',
             'service_id': 'ServiceId-dac6f800-269c-42e0-821e- 07f1e84a061c'}},
         'endpoint_url': 'https://s3.eu-gb.cloud-object- storage.appdomain.cloud',
          'guid': 'f11a9507-df7d-41fc-8b4e-4b913e22d2e3', 'resource_crn':
          'crn:v1:bluemix:public:cloud-object-
storage:global:a/223be9e216084bc6831c3b6c556758f9:f11a9507-df7d-41fc- 8b4e-4b913e22d2e3::'},
         'type': 'bmcos_object_storage'}},
     'metadata': {'created_at': '2022-11-07T10:20:17.909Z', 'creator_id': 'IBMid-
      668000ETEJ',
      'id': '8920acc3-e00b-4c42-9e35-627fbd388e49', 'updated_at': '2022-
      11-07T10:20:35.258Z',
     'url': '/v2/spaces/8920acc3-e00b-4c42-9e35-627fbd388e49'}}]} client.spaces.list()

Note: 'limit' is not provided. Only first 50 records will be displayed if the number of records exceed 50

```
 ----------------------------------------------- ----------
ID                                              NAME        CREATED
8920acc3-e00b-4c42-9e35-627fbd388e49            crude oil   2022-11-
07T10:20:17.909Z
 ----------------------------------------------------------- ----------
 -------------------------
```

space_uid = "8920acc3-e00b-4c42-9e35-627fbd388e49" space_uid

{"type":"string"} client.set.default_space(space_uid)

{"type":"string"} client.software_specifications.list()

```
NAME                              ASSET_ID
--------------------------------  ---------------------------------------
-----
TYPE
default_py3.6                     0062b8c9-8b7d-44a0-a9b9-46c416adcbd9 base
kernel-spark3.2-scala2.12         020d69ce-7ac1-5e68-ac1a-31189867356a base
pytorch-onnx_1.3-py3.7-edt        069ea134-3346-5748-b513-49120e15d288 base
scikit-learn_0.20-py3.6           09c5a1d0-9c1e-4473-a344-eb7b665ff687 base
spark-mllib_3.0-scala_2.12        09f4cff0-90a7-5899-b9ed-1ef348aebdee base
pytorch-onnx_rt22.1-py3.9         0b848dd4-e681-5599-be41-b5f6fccc6471 base
ai-function_0.1-py3.6             0cdb0f1e-5376-4f4d-92dd-da3b69aa9bda base
shiny-r3.6                        0e6e79df-875e-4f24-8ae9-62dcc2148306 base
tensorflow_2.4-py3.7-horovod      1092590a-307d-563d-9b62-4eb7d64b3f22 base
pytorch_1.1-py3.6                 10ac12d6-6b30-4ccd-8392-3e922c096a92 base
tensorflow_1.15-py3.6-ddl         111e41b3-de2d-5422-a4d6-bf776828c4b7 base
autoai-kb_rt22.2-py3.10           125b6d9a-5b1f-5e8d-972a-b251688ccf40 base
runtime-22.1-py3.9                12b83a17-24d8-5082-900f-0ab31fbfd3cb base
scikit-learn_0.22-py3.6           154010fa-5b3b-4ac1-82af-4d5ee5abbc85 base
default_r3.6                      1b70aec3-ab34-4b87-8aa0-a4a3c8296a36 base
pytorch-onnx_1.3-py3.6            1bc6029a-cc97-56da-b8e0-39c3880dbbe7 base
kernel-spark3.3-r3.6              1c9e5454-f216-59dd-a20e-474a5cdf5988 base
pytorch-onnx_rt22.1-py3.9-edt     1d362186-7ad5-5b59-8b6c-9d0880bde37f base
```

tensorflow_2.1-py3.6	1eb25b84-d6ed-5dde-b6a5-3fbdf1665666 base
spark-mllib_3.2	20047f72-0a98-58c7-9ff5-a77b012eb8f5 base
tensorflow_2.4-py3.8-horovod	217c16f6-178f-56bf-824a-b19f20564c49 base
runtime-22.1-py3.9-cuda	26215f05-08c3-5a41-a1b0-da66306ce658 base
do_py3.8	295addb5-9ef9-547e-9bf4-92ae3563e720 base
autoai-ts_3.8-py3.8	2aa0c932-798f-5ae9-abd6-15e0c2402fb5 base
tensorflow_1.15-py3.6	2b73a275-7cbf-420b-a912-eae7f436e0bc base
kernel-spark3.3-py3.9	2b7961e2-e3b1-5a8c-a491-482c8368839a base
pytorch_1.2-py3.6	2c8ef57d-2687-4b7d-acce-01f94976dac1 base
spark-mllib_2.3	2e51f700-bca0-4b0d-88dc-5c6791338875 base
pytorch-onnx_1.1-py3.6-edt	32983cea-3f32-4400-8965-dde874a8d67e base
spark-mllib_3.0-py37	36507ebe-8770-55ba-ab2a-eafe787600e9 base
spark-mllib_2.4	390d21f8-e58b-4fac-9c55-d7ceda621326 base
autoai-ts_rt22.2-py3.10	396b2e83-0953-5b86-9a55-7ce1628a406f base
xgboost_0.82-py3.6	39e31acd-5f30-41dc-ae44-60233c80306e base
pytorch-onnx_1.2-py3.6-edt	40589d0e-7019-4e28-8daa-fb03b6f4fe12 base
pytorch-onnx_rt22.2-py3.10	40e73f55-783a-5535-b3fa-0c8b94291431 base
default_r36py38	41c247d3-45f8-5a71-b065-8580229facf0 base
autoai-ts_rt22.1-py3.9	4269d26e-07ba-5d40-8f66-2d495b0c71f7 base
autoai-obm_3.0	42b92e18-d9ab-567f-988a-4240ba1ed5f7 base
pmml-3.0_4.3	493bcb95-16f1-5bc5-bee8-81b8af80e9c7 base
spark-mllib_2.4-r_3.6	49403dff-92e9-4c87-a3d7-a42d0021c095 base
xgboost_0.90-py3.6	4ff8d6c2-1343-4c18-85e1-689c965304d3 base
pytorch-onnx_1.1-py3.6	50f95b2a-bc16-43bb-bc94-b0bed208c60b base
autoai-ts_3.9-py3.8	52c57136-80fa-572e-8728-a5e7cbb42cde base

```
spark-mllib_2.4-scala_2.11                   55a70f99-7320-4be5-9fb9-9edb5a443af5 base
spark-mllib_3.0                              5c1b0ca2-4977-5c2e-9439-ffd44ea8ffe9 base
autoai-obm_2.0                               5c2e37fa-80b8-5e77-840f-d912469614ee base
spss-modeler_18.1                            5c3cad7e-507f-4b2a-a9a3-ab53a21dee8b base
cuda-py3.8                                    5d3232bf-c86b-5df4-a2cd-7bb870a1cd4e base
autoai-kb_3.1-py3.7                           632d4b22-10aa-5180-88f0-f52dfb6444d7 base
pytorch-onnx_1.7-py3.8                        634d3cdc-b562-5bf9-a2d4-ea90a478456b base
```

```
-------------------------------  --------------------------------------
-----
```

Note: Only first 50 records were displayed. To display more use 'limit' parameter.

software_space_uid = client.software_specifications.get_uid_by_name("tensorflow_rt22.1- py3.9")
software_space_uid

{"type":"string"}

model_details = client.repository.store_model(model="crudeoil- prediction.tgz",
meta_props={
        client.repository.ModelMetaNames.NAME:"Crude Oil Model",
        client.repository.ModelMetaNames.TYPE:"tensorflow_2.7",

client.repository.ModelMetaNames.SOFTWARE_SPEC_UID:software_space_uid
})

model_details

  {'entity': {'hybrid_pipeline_software_specs': [], 'software_spec': {'id': 'acd9c798-6974-5d2f-a657-
    ce06e986df4d',
      'name': 'tensorflow_rt22.1-py3.9'}, 'type':
    'tensorflow_2.7'},
   'metadata': {'created_at': '2022-11-15T11:38:43.943Z', 'id': 'e4854139-835a-
    4dae-b61e-2c8644333c0d', 'modified_at': '2022-11-15T11:38:48.290Z',
    'name': 'Crude Oil Model', 'owner':
    'IBMid-668000ETEJ',
    'resource_key': '1bec88c6-df9a-4c62-93f8-13c3591666d0', 'space_id': '8920acc3-
    e00b-4c42-9e35-627fbd388e49'},
   'system': {'warnings': []}}

model_id = client.repository.get_model_id(model_details) model_id

53

client.repository.download(model_id,'Crude_oil_prediction.tgz') Successfully saved model content

to file: 'Crude_oil_prediction.tgz'

FLASK INTEGRATION – IBM CLOUD (app2.py)

```python
import
requests
import numpy as np
from flask import Flask, render_template, request, jsonify
# NOTE: you must manually set API_KEY below using information retrieved from
your IBM Cloud account.
API_KEY = "UPWRFTFhkvYA6wNCvrVT21hKgkcyyWYJTLut7wa6eSRu"
token_response = requests.post('https://iam.cloud.ibm.com/identity/token',
data={"apikey":
 API_KEY, "grant_type": 'urn:ibm:params:oauth:grant-type:apikey'})
mltoken = token_response.json()["access_token"]
header = {'Content-Type': 'application/json', 'Authorization': 'Bearer ' +
mltoken}
app = Flask(__name__)
@app.route('/')
def home():
    return render_template("index.html")
@app.route('/index.html')
def home1():
    return render_template("index.html")
@app.route('/new.html')
def home2():
    return render_template("new.html")
@app.route('/login',methods=['POST','GET'])
def login():
 if request.method == 'POST':
    x=str(request.form['year'])
    x=x.split(',')
    print(x)
    for w in range(0, len(x)):
        x[w]=float(x[w])
    print(x)
```

```
        t=[[ [x[0]], [x[1]], [x[2]], [x[3]], [x[4]], [x[5]], [x[6]], [x[7]], [x[8]],
[x[9]]]]
    payload_scoring = {
    "input_data": [{"field": [[["i1"], ["i2"], ["i3"], ["i4"], ["i5"], ["i6"],
["i7"], ["i8"], ["i9"], ["1i0"]]],
                       "values":t }]}
    response_scoring = requests.post('https://us-
south.ml.cloud.ibm.com/ml/v4/deployments/f73b8e94-628a-4cd8-8a84-
a5b527eb1468/predictions?version=2022-11-17', json=payload_scoring,
    headers={'Authorization': 'Bearer ' + mltoken})
    print("Scoring response")
    print(response_scoring.json())
    predictions=response_scoring.json()
    print("predicted value is")
    print(predictions['predictions'][0]['values'][0][0])
    pred=predictions['predictions'][0]['values'][0][0]
    return render_template("result.html",result=str(pred))
if __name__=='__main__':
  app.run(debug=True, port=5000)
```

DEMO LINK:

https://youtu.be/_Ui4rznV-o0

GITHUB LINK:

IBM-EPBL/IBM-Project-734-1658318512: Crude Oil Price Prediction (github.com)