CRUDEOILPRICEPREDICTIONUSINGARTIFICIALINTELLIGENCE

**TEAM ID : PNT2022TMID44721**

**TEAM MEMBERS : DHARMENDHIRA G G (732519106002)**

**GOKUL K (732519106003)**

**JEBASTIN J (732519106006)**

**MEENAKSHI S (732519106013)**

**DEPARTMENT : ELECTRONICS & COMMUNICATION ENGINEERING**

**COLLEGE NAME :SHREE VENKATESHWARA HITECH ENGINEERING**

**COLLEGE, GOBI**

**1.INTRODUCTION**

**1.1 PROJECT 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.

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**

**S. N. Abdullah,X. Zeng,** ON **Machine Learning Approach for Crude Oil Price Prediction with Artificial Neural Networks-Quantitative (ANN-Q) Model:**The volatility of crude oil market and its chain effects to the world economy augmented the interest and fear of individuals, public and private sectors. Previous statistical and econometric techniques used for prediction, offer good results when dealing with linear data. Nevertheless, crude oil price series deal with high nonlinearity and irregular events. The continuous usage of statistical and econometric techniques for crude oil price prediction might demonstrate demotions to the prediction performance. Machine Learning and Computational Intelligence approach through combination of historical quantitative data with qualitative data from experts’ view and news is a remedy proposed to predict this. This paper will discuss the first part of the research, focusing on to (i) the development of Hierarchical Conceptual (HC) model and (ii) the development of Artificial Neural Networks-Quantitative (ANN-Q) model.

**B. Abramson, A. Finizza**, **“Probabilistic forecasts from probabilistic models: A case study in the oil market,”**at **International Journal of Forecasting** : Probabilistic forecasts, probabilistic models, and contingent policy recommendations are inextricably intertwined. This article describes a case study in the use of inherently probabilistic belief network models to produce probabilistic forecasts of average annual oil prices. Belief networks are flexible enough to capture both standard, data-driven economic variables, and quantified expert judgements about the politics of the oil market (particularly the production and capacity policies of key OPEC members). These variables are interrelated by a combination of algebraic formulas, conditional probabilities, and econometric relations. The resultant network is used to test the impact of a variety of different scenarios. The probabilistic forecasts generated by running Monte Carlo analyses on these scenario networks provide corporate decision-makers with useful insights and recommendations.

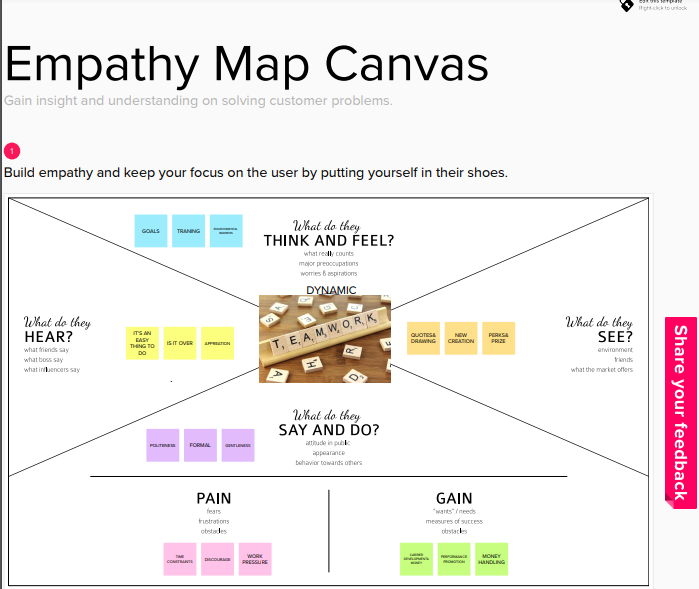
**CLAUDIO MORANA** on **A SEMIPARAMETRIC APPROACH TO SHORT-TERM OIL PRICE FORECASTING** :In this paper it is shown how the GARCH properties of oil price changes can be employed to forecast the oil price distribution over short-term horizons. The forecasting methodology is semiparametric and it is based on the bootstrap approach. The results of an out-of-sample forecasting exercise, carried out using the Brent oil price series, suggest that the forecasting approach can be used to obtain a performance measure for the forward price, in addition to compute interval forecasts for the oil price.

**S.H., Kang** and **S.M Yoon**on **ForecastingVolatility of Crude Oil**

**Markets:** This article investigates the efficacy of a volatility model for three crude oil markets -- Brent, Dubai, and West Texas Intermediate (WTI) -- with regard to its ability to forecast and identify voltility stylized facts, in particular volatility persistence or long memory. In this context, we assess persistence in the volatility of the three crude oil prices using conditional volatility models. The CGARCH and FIGARCH models are better equipped to capture persistence than are the GARCH and IGARCH models. The CGARCH and FIGARCH models also provide superior performance in out-of-sample volatility forecasts. We conclude that the CGARCH and FIGARCH models are useful for modeling and forecasting persistence in the volatility of crude oil prices.

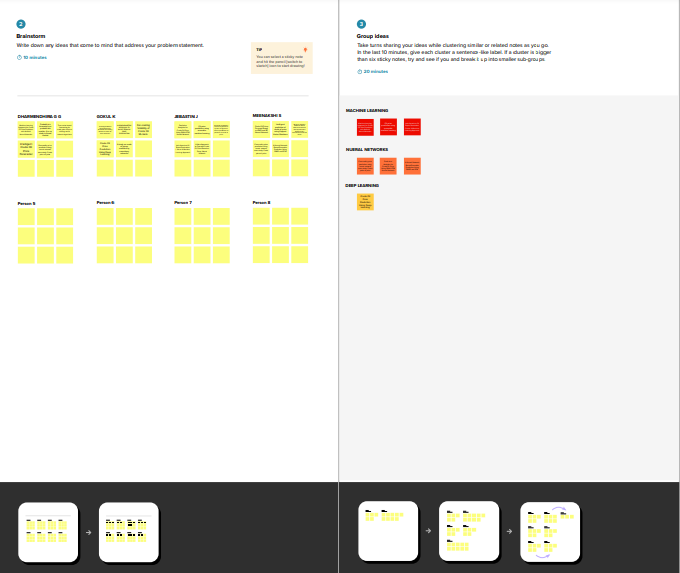
**3. IDEATION AND PROPOSED SOLUTION**

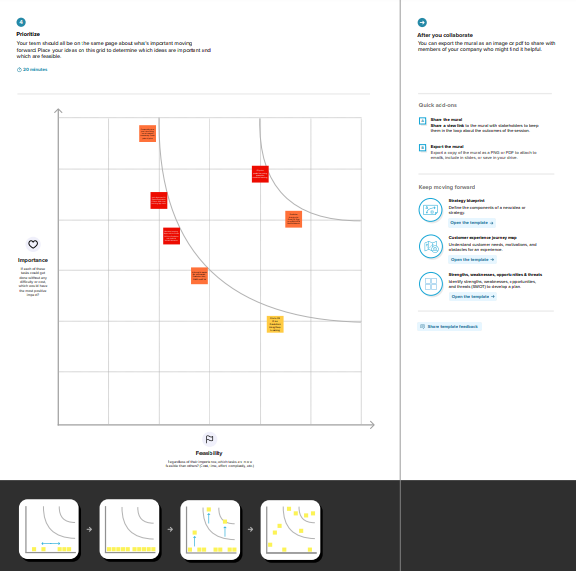
**3.1 EMPATHY MAP CANVAS:**

.****

**3.2 IDEATION & BRAINSTROMING:**

****

****

****

**3.3 PROPOSED SOLUTION:**

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.

**NOVELTY:**

There has been a renewed interest in accurately forecasting the price of crude oil and its fluctuations. Buying crude oil at a proper time is crucial to avoid risk of losses. Time series analyses 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 prices. Time-series data will be collected and pre-processed as needed, and two architectures of computational neural networks will be tested: Recurrent Neural Network(RNN) and long-short term memory (LSTM) neural networks. The findings suggest that LSTM networks are the best architectures to predict the crude oil price. The outcomes of this project could potentially help in making the oil price prediction mechanism a more tractable task and in assisting decision-makers to improve macroeconomic policies, generate enhanced macroeconomic projections, and better assess macroeconomic risks.

**SOCIAL IMPACTS:**

Crude oil is considered one of the most important fuel sources and contributes to over a third of the world's energy consumption thus making the global oil industry a multitrillion sector. The surge in the crude oil prices since 2002 has renewed interest in determining what variables affect the price of crude oil and has highlighted the importance of the ability to accurately predict the evolution of its prices. Using Artificial intelligence will provide more accurate prediction of prices which in turn will reduce losses when investing in crude oil market.

**BUSINESS MODEL:**

Artificial intelligence is drifting out of R&D labs and into the business world. Millions of industries across the globe and top-notch companies are fitting together the power of AI and Applied artificial intelligence (AAI). Most of the business industries spot the scams using machine learning algorithms in nanoseconds to improve customer satisfactions. A vivid rise in the machine learning tools, business platforms, and applications-based tools were developed to quench the business satisfactions. These state-of-the-art technologies not only compressed the quality of the internet and the software industry but also other verticals such as built-up, healthcare system, legal, automobile, and agriculture as well as in safety.

**SCALABILITY:**

Currently, nearly half of all companies rely on artificial intelligence (AI) for handling data quality. This powerful tool can be used to quickly and effectively predict investment outcomes, as well as to devise strategies or establish long-term goals. Scalable AI pertains to how data models, infrastructures, and algorithms are able to increase or decrease their complexity, speed, or size at scale in order to best handle the requirements of the situation at hand. As improvements continue with data storage capacities as well as computing resources, AI models can be created with billions of parameters. It’s extremely helpful for extracting value from large data sets and spotting patterns or trends that would be difficult or impossible for a human to notice. Load scalability pertains to software that can speed up its performance with regard to the available computing power.

**3.4 PROBLEM SOLUTION FIT:**

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. There has been a renewed interest in accurately forecasting the price of crude oil and its fluctuations. Buying crude oil at a proper time is crucial to avoid risk of losses. Time series analyses 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 prices. Time-series data will be collected and pre-processed as needed, and two architectures of computational neural networks will be tested: Recurrent Neural Network (RNN) and long-short term memory (LSTM) neural networks. The findings suggest that LSTM networks are the best architectures to predict the crude oil price. The outcomes of this project could potentially help in making the oil price prediction mechanism a more tractable task and in assisting decision-makers to improve macroeconomic policies, generate enhanced macroeconomic projections, and better assess macroeconomic risks.

**4. REQUIREMENT ANALYSIS**

**4.1 FUNCTIONAL REQUIREMENT:**

Following are the functional requirements of the proposed solution.

|  |  |  |
| --- | --- | --- |
| **FR.NO** | **Functionalrequirement** | **Subrequirement(story/subtask)** |
| FR-1 | Userregistration | |  | | --- | | Registration through Form. | |
| FR-2 | Userconfirmation | ConfirmationviaSMS. |
| FR-3 | Fetching input data | Give the model the input data. |
| FR-4 | Generating results | Prediction of oil prices |
|  |  |  |

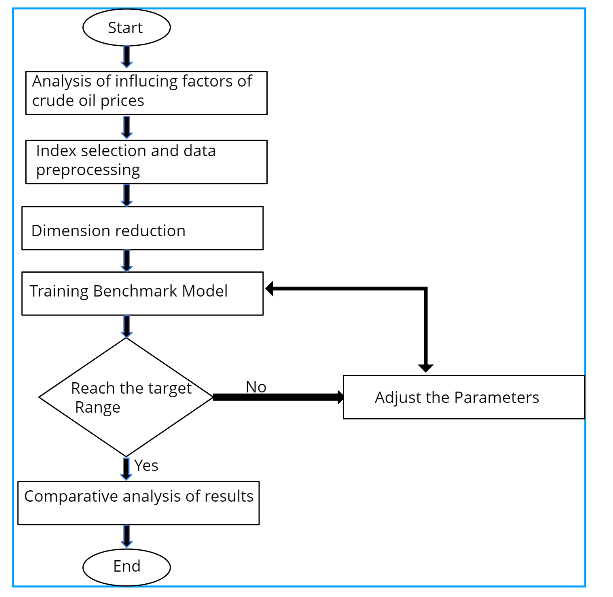
**NON-FUNCTIONAL REQUIREMENTS:**

Followingarethenon-functionalrequirementoftheproposedsolution.

|  |  |  |
| --- | --- | --- |
| **NFR.NO** | **Non-functionalrequirement** | **Description** |
| **NFR-1** | Usability | User interfaces are easy to use |
| **NFR-2** | Security | Sensitive data is protected. |
| **NFR-3** | Reliability | Because there is very little variance from the prediction . the testing is highly dependable. |
| **NFR-4** | Performance | Using LSTM networks gives high performance. |
| **NFR-5** | Availability | The system tested with 4 datasets and the system operating properly. |
| **NFR-6** | Scalability | LSTM model works efficiently for large number of users. |

**5.PROJECT DESIGN**

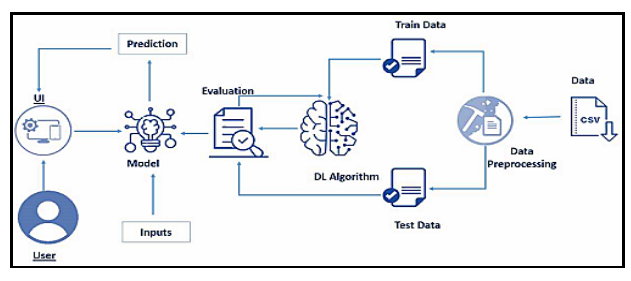
**5.1 DATA FLOW DIAGRAMS:**

****

**5.2 SOLUTION & TECHNICAL ARCHITECTURE:**

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. There has been a renewed interest in accurately forecasting the price of crude oil and its fluctuations. Buying crude oil at a proper time is crucial to avoid risk of losses. Time series analyses 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 prices. Time-series data will be collected and pre-processed as needed, and two architectures of computational neural networks will be tested: Recurrent Neural Network (RNN) and long-short term memory (LSTM) neural networks. The findings suggest that LSTM networks are the best architectures to predict the crude oil price. The outcomes of this project could potentially help in making the oil price prediction mechanism a more tractable task and in assisting decision-makers to improve macroeconomic policies, generate enhanced macroeconomic projections, and better assess macroeconomic risks.

**TECHNICAL ARCHITECTURE**

****

**6.PROJECT PLANNING & SCHEDULING**

**6.1 SPRINT PLANNING & ESTIMATION:**

|  |  |  |
| --- | --- | --- |
| TITLE | DESCRIPTION | DATE |
| Literature Survey &InformationGathering | Literaturesurveyon theselectedproject &  gathering information by referring the,technicalpapers,researchpublicationsetc. | 20September2022 |
| PrepareEmpathyMap | PrepareEmpathyMap Canvastocapturethe  userPains&Gains,Preparelistofproblemstatements | 24September2022 |
| Ideation | List the by organizing the brainstormingsessionandprioritizethetop3ideasbased  onthefeasibility&importance. | 26September2022 |
| ProposedSolution | Prepare the proposed solution document,which includes the novelty, feasibility ofidea,businessmodel,socialimpact,  scalabilityofsolution,etc. | 28September2022 |
| ProblemSolutionFit | Prepareproblem -solutionfitdocument. | 30September2022 |
| SolutionArchitecture | Preparesolutionarchitecturedocument. | 02October2022 |
| CustomerJourney | Prepare the customer journey maps tounderstand the user interactions &experienceswiththeapplication(entryto  exit). | 05October2022 |
| FunctionalRequirement | Preparethefunctionalrequirement  document. | 07October2022 |
| DataFlowDiagrams | Drawthedataflowdiagramsandsubmitfor  review. | 09October2022 |
| TechnologyArchitecture | Preparethetechnologyarchitecturediagram. | 13October2022 |

|  |  |  |
| --- | --- | --- |
| PrepareMilestone&Activity  List | Preparethemilestones& activitylistof the  project. | 24October2022 |
| SprintSchedule | Preparespringplan | 24October2022 |
| DeliveryofSprint-1 | Develop&submit thedevelopedcode. | 30October2022 |
| DeliveryofSprint-2 | Develop&submit thedevelopedcode. | 06November2022 |
| DeliveryofSprint-3 | Develop&submit thedevelopedcode. | 13November2022 |
| DeliveryofSprint-4 | Develop&submit thedevelopedcode. | 17November2022 |

**6.2 SPRINT DELIVERY SCHEDULE :**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Sprint** | **FunctionalRequirement(Epic)** | **UserStoryNumber** | **UserStory/Task** | **StoryPoints** | **Priority** | **TeamMembers** |
| Sprint-1 | DataCollection | USN-1 | DownloadCrudeOilPrice Dataset | 2 | Medium | Dharmendhira G G  Gokul K |
| Sprint-1 | DataPreprocessing | USN-2 | ImportingTheDatasetinto Workspace | 1 | Low | Dharmendhira G G |
| Sprint-1 |  | USN-3 | HandlingMissingData | 3 | Medium | Jebastin J  Gokul K |
| Sprint-1 |  | USN-4 | FeatureScaling | 3 | Low | Dharmendhira G G  Meenakshi S |
| Sprint-1 |  | USN-5 | DataVisualization | 3 | Medium | Gokul K  Meenakshi S |
| Sprint-1 |  | USN-6 | SplittingDataintoTrainandTest | 4 | High | Dharmendhira G G  Jebastin J |
| Sprint-1 |  | USN-7 | CreatingADatasetwithSlidingWindows | 4 | High | Dharmendhira G G  Meenkshi S  Gokul K |
| Sprint-2 | ModelBuilding | USN-8 | ImportingTheModelBuildingLibraries | 1 | Medium | Jebastin J Meenakshi S |
| Sprint-2 |  | USN-9 | InitializingTheModel | 1 | Medium | Dharmendhira G G |
| Sprint-2 |  | USN-10 | AddingLSTMLayers | 2 | High | Gokul K  Jebastin J |
| Sprint-2 |  | USN-11 | AddingOutputLayers | 3 | Medium | Dharmendhira G G |
| Sprint-2 |  | USN-12 | ConfigureTheLearningProcess | 4 | High | Dharmendhira G G |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Sprint-2 |  | USN-13 | TrainTheModel | 2 | Medium | DharmendhiraG G |
| Sprint-2 |  | USN-14 | ModelEvaluation | 1 | Medium | Gokul K  Meenakshi S |
| Sprint-2 |  | USN-15 | SaveTheModel | 2 | Medium | Dharmendhira G G  Gokul K |
| Sprint-2 |  | USN-16 | TestTheModel | 3 | High | Dharmendhira G G  Jebastin J  Meenakshi S |
| Sprint-3 | ApplicationBuilding | USN-17 | CreateAnHTMLFile | 4 | Medium | Jebastin J  Meenakshi S |
| Sprint-3 |  | USN-18 | BuildPythonCode | 4 | High | Dharmendhira G G  Gokul K  Meenakshi S |
| Sprint-3 |  | USN-19 | RunTheApp in LocalBrowser | 4 | Medium | Gokul K  Jebastin J |
| Sprint-3 |  | USN-20 | ShowcasingPredictionOnUI | 4 | High | Dharmendhira G G  Gokul K  Jebastin j  Meenakshi S |
| Sprint-4 | TrainTheModelOnIBM | USN-21 | RegisterForIBMCloud | 4 | Medium | Dharmendhira G G  Gokul K  Jebastin J  Meenakshi S |
| Sprint-4 |  | USN-22 | TrainTheMLModelOnIBM | 8 | High | Jebastin J Gokul K |
| Sprint-4 |  | USN-23 | IntegrateFlaskwithScoringEndPoint | 8 | High | Dharmendhira G G  Jebastin J  Meenakshi S |

**7. CODING & SOLUTIONING**

**# Data Preprocessing**

**## Import the libraries**

**"""**

**!pip install ibm-cos-sdk**

**import ibm\_boto3**

**from ibm\_botocore.client import Config**

**import pandas as pd**

**import numpy as np**

**import io, datetime**

**import matplotlib.pyplot as plt**

**import seaborn as sns**

**import statsmodels.api as sm**

**from pylab import rcParams**

**from sklearn.preprocessing import MinMaxScaler**

**"""## Importing the dataset"""**

**cos\_credentials={**

**"apikey": "5lDfM8QcqpTFlKVVjKmm06zGbSspFR6gqGpmbFDWLlRc",**

**"endpoints": "https://control.cloud-object-storage.cloud.ibm.com/v2/endpoints",**

**"iam\_apikey\_description": "Auto-generated for key crn:v1:bluemix:public:cloud-object-storage:global:a/d2c796b84a794b58a1cff48368133ea1:e1e617d9-39a8-465d-bd9a-f3ca9bbb5297:resource-key:cd3a6762-cdaf-4808-b931-198e378e86d5",**

**"iam\_apikey\_name": "pnt2022tmid13214",**

**"iam\_role\_crn": "crn:v1:bluemix:public:iam::::serviceRole:Writer",**

**"iam\_serviceid\_crn": "crn:v1:bluemix:public:iam-identity::a/d2c796b84a794b58a1cff48368133ea1::serviceid:ServiceId-469be452-375b-4d58-9c23-d742c9a3256e",**

**"resource\_instance\_id": "crn:v1:bluemix:public:cloud-object-storage:global:a/d2c796b84a794b58a1cff48368133ea1:e1e617d9-39a8-465d-bd9a-f3ca9bbb5297::"**

**}**

**auth\_endpoint = 'https://iam.cloud.ibm.com/oidc/token'**

**service\_endpoint = 'https://s3.us-east.cloud-object-storage.appdomain.cloud'**

**cos = ibm\_boto3.client('s3',**

**ibm\_api\_key\_id=cos\_credentials['apikey'],**

**ibm\_service\_instance\_id=cos\_credentials['resource\_instance\_id'],**

**ibm\_auth\_endpoint=auth\_endpoint,**

**config=Config(signature\_version='oauth'),**

**endpoint\_url=service\_endpoint)**

**obj =cos.get\_object(Bucket='pnt2022tmid13214', Key='Crude Oil Prices Daily.xlsx')**

**df = pd.read\_excel(io.BytesIO(obj['Body'].read()), header=None, names=['date', 'price'] ,skiprows=1)**

**df.head()**

**"""## Handling missing data"""**

**df.isnull().any()**

**df.dropna(axis=0,inplace=True)**

**df.isnull().any()**

**df.shape**

**"""## Data visualization"""**

**plot = plt.figure(figsize=(15, 6))**

**time = pd.to\_datetime(df['date'])**

**price = list(df['price'])**

**data = pd.Series(price, time)**

**plt.plot(data)**

**#Decompose the plot**

**df.set\_index('date', inplace=True)**

**y = df['price'].resample('MS').mean()**

**y.plot(figsize=(15, 6))**

**plt.show()**

**rcParams['figure.figsize'] = 18, 8**

**decomposition = sm.tsa.seasonal\_decompose(y, model='additive')**

**fig = decomposition.plot()**

**plt.show()**

**"""## Feature Scaling**

**"""**

**df1 = df.reset\_index()['price']**

**sc = MinMaxScaler(feature\_range = (0, 1))**

**df1 = sc.fit\_transform(np.array(df1).reshape(-1,1))**

**df1.shape**

**"""## Train Test Split"""**

**train\_size = int(len(df1) \* 0.80)**

**test\_size = len(df1) - train\_size**

**train, test = df1[0:train\_size, :], df1[train\_size:len(df1), :]**

**len(test)**

**"""## Creating Window"""**

**def dataset(df, lookback=1):**

**data\_x, data\_y = [], []**

**for i in range(len(df) - lookback - 1):**

**a = df[i:(i + lookback), 0]**

**data\_x.append(a)**

**data\_y.append(df[i + lookback, 0])**

**return np.array(data\_x), np.array(data\_y)**

**time\_step = 10**

**# Reshape into X=t and Y=t+1**

**X\_train ,Y\_train = dataset(train,time\_step)**

**X\_test ,Y\_test = dataset(test,time\_step)**

**# Reshape input to be [samples, time steps, features]**

**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)**

**X\_train.shap**

**8. TESTING**

**8.1 TEST CASES**

A test case has components that describe input, action and an expected response, in order to determine if a feature of an application is working correctly. A test case is a set of instructions on “HOW” to validate a particular test objective/target, which when followed will tell us if the expected behavior of the system is satisfied or not.

8.2 Characteristics of a good test case:

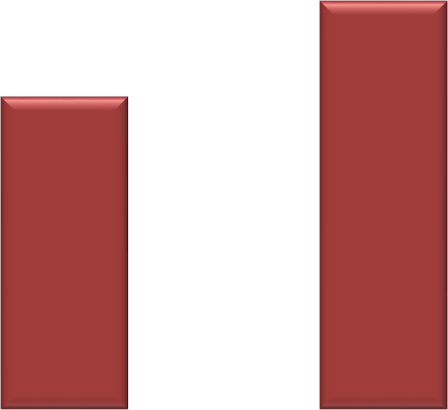
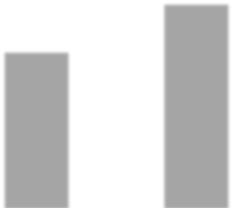
* + - Accurate: Exacts the purpose.
    - Economical: No unnecessary steps or words.
    - Traceable: Capable of being traced to requirements.
    - Repeatable: Can be used to perform the test over and over.
    - Reusable: Can be reused if necessary.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **S.NO** | **Scenario** | **Input** | **Output** | **Status** |
| 1 | User login | User name and password | Login | Login success. |
| 2 | Price Prediction | Give input in the user defined parameters | Predicting the crude  oil prices by getting the previous prices | Details are stored in a database. |

**8.3USER ACCEPTANCE TESTING**

This sort of testing is carried out by users, clients, or other authorised bodies to identify the requirements and operational procedures of an application or piece of software. The most crucial stage of testing is acceptance testing since it determines whether or not the customer will accept the application or program. It could entail the application's U.I., performance, usability, and usefulness. It is also referred to as end-user testing, operational acceptance testing, and user acceptance testing (UAT).

**9.RESULT**



120

100

80

60

40

20

0

Algorithm Accuracy Level

GBM

XGBOOST

|  |
| --- |
|  |
|  |
|  |
|  |
|  |
|  |

**10.ADVANTAGES & DISADVANTAGES**

**10.1 ADVANTAGES**

* The principal advantage of RNN over ANN is that RNN can model a collection of records (i.e. time collection) so that each pattern can be assumed to be dependent on previous ones.
* Recurrent neural networks are even used with convolutional layers to extend the powerful pixel neighbourhood.
* Give accurate result
* Easy to access and get the price
* Effective with large datasets
* **10.2 DISADVANTAGES**
* The computation of this neural network is slow.
* Training can be difficult.
* If you are using the activation functions, then it becomes very tedious to process long sequences.
* Hard to find oil price
* Inefficient in accuracy
* Poor Customer support

**11. CONCLUSION**

Predicting Crude Oil prices is a very challenging problem due to the high volatility of oil prices. In this paper, we developed a new oil price prediction approach using ideas and tools from stream learning, a machine learning paradigm for analysis and inference of continuous flow of non-stationary data. Our stream learning model will be updated whenever new oil price data are available, and provided to model, so the model continuously evolves over time, and can capture the changing pattern of oil prices. In addition, updating the model requires only a small constant time per new data example, the experiment results show that our stream learning model outperformed four other popular oil price prediction models over a variety of forecast time horizons. This process is used to Predict the oil Prices. The prediction model predicts continuous valued functions

**12.FUTURE SCOPE**

Future research may extend our work by considering a richer set of market variables, such as political or commercial factors and phases of economic instability, which are often determinants of crude oil price. Moreover, another direction for future research is the application of the proposed model to forecast the price of other commodities. Moreover, it is a worthwhile direction to explore the consideration of one or more computational cost factors when comparing different forecasting models. Therefore, calculations based on operational research methods might be a good direction.

**13.APPENDIX**

**13.1 SOURCE CODE**

**# Data Preprocessing**

**## Import the libraries**

**"""**

**!pip install ibm-cos-sdk**

**import ibm\_boto3**

**from ibm\_botocore.client import Config**

**import pandas as pd**

**import numpy as np**

**import io, datetime**

**import matplotlib.pyplot as plt**

**import seaborn as sns**

**import statsmodels.api as sm**

**from pylab import rcParams**

**from sklearn.preprocessing import MinMaxScaler**

**"""## Importing the dataset"""**

**cos\_credentials={**

**"apikey": "5lDfM8QcqpTFlKVVjKmm06zGbSspFR6gqGpmbFDWLlRc",**

**"endpoints": "https://control.cloud-object-storage.cloud.ibm.com/v2/endpoints",**

**"iam\_apikey\_description": "Auto-generated for key crn:v1:bluemix:public:cloud-object-storage:global:a/d2c796b84a794b58a1cff48368133ea1:e1e617d9-39a8-465d-bd9a-f3ca9bbb5297:resource-key:cd3a6762-cdaf-4808-b931-198e378e86d5",**

**"iam\_apikey\_name": "pnt2022tmid13214",**

**"iam\_role\_crn": "crn:v1:bluemix:public:iam::::serviceRole:Writer",**

**"iam\_serviceid\_crn": "crn:v1:bluemix:public:iam-identity::a/d2c796b84a794b58a1cff48368133ea1::serviceid:ServiceId-469be452-375b-4d58-9c23-d742c9a3256e",**

**"resource\_instance\_id": "crn:v1:bluemix:public:cloud-object-storage:global:a/d2c796b84a794b58a1cff48368133ea1:e1e617d9-39a8-465d-bd9a-f3ca9bbb5297::"**

**}**

**auth\_endpoint = 'https://iam.cloud.ibm.com/oidc/token'**

**service\_endpoint = 'https://s3.us-east.cloud-object-storage.appdomain.cloud'**

**cos = ibm\_boto3.client('s3',**

**ibm\_api\_key\_id=cos\_credentials['apikey'],**

**ibm\_service\_instance\_id=cos\_credentials['resource\_instance\_id'],**

**ibm\_auth\_endpoint=auth\_endpoint,**

**config=Config(signature\_version='oauth'),**

**endpoint\_url=service\_endpoint)**

**obj =cos.get\_object(Bucket='pnt2022tmid13214', Key='Crude Oil Prices Daily.xlsx')**

**df = pd.read\_excel(io.BytesIO(obj['Body'].read()), header=None, names=['date', 'price'] ,skiprows=1)**

**df.head()**

**"""## Handling missing data"""**

**df.isnull().any()**

**df.dropna(axis=0,inplace=True)**

**df.isnull().any()**

**df.shape**

**"""## Data visualization"""**

**plot = plt.figure(figsize=(15, 6))**

**time = pd.to\_datetime(df['date'])**

**price = list(df['price'])**

**data = pd.Series(price, time)**

**plt.plot(data)**

**#Decompose the plot**

**df.set\_index('date', inplace=True)**

**y = df['price'].resample('MS').mean()**

**y.plot(figsize=(15, 6))**

**plt.show()**

**rcParams['figure.figsize'] = 18, 8**

**decomposition = sm.tsa.seasonal\_decompose(y, model='additive')**

**fig = decomposition.plot()**

**plt.show()**

**"""## Feature Scaling**

**"""**

**df1 = df.reset\_index()['price']**

**sc = MinMaxScaler(feature\_range = (0, 1))**

**df1 = sc.fit\_transform(np.array(df1).reshape(-1,1))**

**df1.shape**

**"""## Train Test Split"""**

**train\_size = int(len(df1) \* 0.80)**

**test\_size = len(df1) - train\_size**

**train, test = df1[0:train\_size, :], df1[train\_size:len(df1), :]**

**len(test)**

**"""## Creating Window"""**

**def dataset(df, lookback=1):**

**data\_x, data\_y = [], []**

**for i in range(len(df) - lookback - 1):**

**a = df[i:(i + lookback), 0]**

**data\_x.append(a)**

**data\_y.append(df[i + lookback, 0])**

**return np.array(data\_x), np.array(data\_y)**

**time\_step = 10**

**# Reshape into X=t and Y=t+1**

**X\_train ,Y\_train = dataset(train,time\_step)**

**X\_test ,Y\_test = dataset(test,time\_step)**

**# Reshape input to be [samples, time steps, features]**

**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)**

**X\_train.shape**

**13.2 PREDICT.HTML**

<!DOCTYPE html>

<head>

    <title>Crude Oil Price Prediction </title>

    <link href='https://fonts.googleapis.com/css?family=Roboto' rel='stylesheet'>

    <link rel="stylesheet" href="{{ url\_for('static', filename='css/predict.css') }}">

</head>

<body style="text-align:center;background-color: lightsteelblue;">

    <h1 style="color: white;font-size: 50px;font-family: roboto;">

    Crude Oil Price Prediction </h1>

    <h1 style="color: white;font-size: 50px;font-family: roboto;">

        Enter the Oil price for 10 days </h1>

        <form action="/predict" method="POST" enctype = "multipart/form-data">

            <div style="color:white;font-size:50px;font-family:roboto;">

                {{prediction}}

            </div>

        <input type="text" name="val" style="border-radius: 18px;padding: 20px;width: 300px;height: 15px;text-align: center; align:center;" >

         <br> <br> <br>

        <input type="submit"/ style="border-radius: 9px;;padding: 10px;width: 150px;

        height: 40px;text-align: center;background: #003d66;color: white;">

        </form>

        <br>

        <form action="/predict" method="GET" enctype = "multipart/form-data">

        <input type="submit"/ value="Reset" style="border-radius: 9px;;padding: 10px;width: 150px;

        height: 40px;text-align: center;background: #003d66;color: white;">

        </form>

</body>

**13.2 INDEX.CSS**

h1 {

    text-align: center;

    color: floralwhite;

    font-size: 50px;

    font-family: roboto;

}

p {

    font-family: roboto;

    color: ghostwhite;

    margin-right: 30px;

    margin-left: 30px;

    text-align: center;

    font-size: 20px;

    font-weight: bold;

}

body {

    background: url(index.png);

    background-repeat: no-repeat;

    background-size: cover;

}

.button {

    display: inline-block;

    border-radius: 4px;

    background-color: black;

    border: none;

    color: #FFFFFF;

    text-align: center;

    font-size: 20px;

    padding: 12px;

    width: 100px;

    transition: all 0.5s;

    cursor: pointer;

    margin: 5px;

}

a {

    font-size: 20px;

    font-family: roboto;

    color: ghostwhite;

    margin-right: 30px;

    margin-left: 30px;

    text-align: center;

    font-size: 20px;

    font-weight: bold;

}

table {

    background: slateblue;

    opacity: 0.8;

    margin-left:auto;

    margin-right:auto;

    margin-bottom: 0px;

}

th,

td {

    text-align: left;

    color: black;

    font-size: 30px;

    font-family: roboto;

}

**13.2 APP.PY**

from flask import Flask,render\_template,request,redirect

import numpy as np

import joblib

from keras.models import load\_model

app = Flask(\_\_name\_\_)

@app.route('/',methods=["GET"])

def index():

    return render\_template('index.html')

@app.route('/predict',methods=["POST","GET"])

def predict():

    if request.method == "POST":

        string = request.form['val']

        if(string ==""):

            return render\_template('predict.html')

        string = string.split(',')

        x\_input = [eval(i) for i in string]

        sc = joblib.load("scaler.save")

        x\_input = sc.fit\_transform(np.array(x\_input).reshape(-1,1))

        x\_input = np.array(x\_input).reshape(1,-1)

        x\_input = x\_input.reshape(1,-1)

        x\_input = x\_input.reshape((1,10,1))

        model = load\_model('model.h5')

        output = model.predict(x\_input)

        val = sc.inverse\_transform(output)

        return render\_template('predict.html', prediction = "The predicted price is {:.2f}".format(val[0][0]))

    if request.method == "GET":

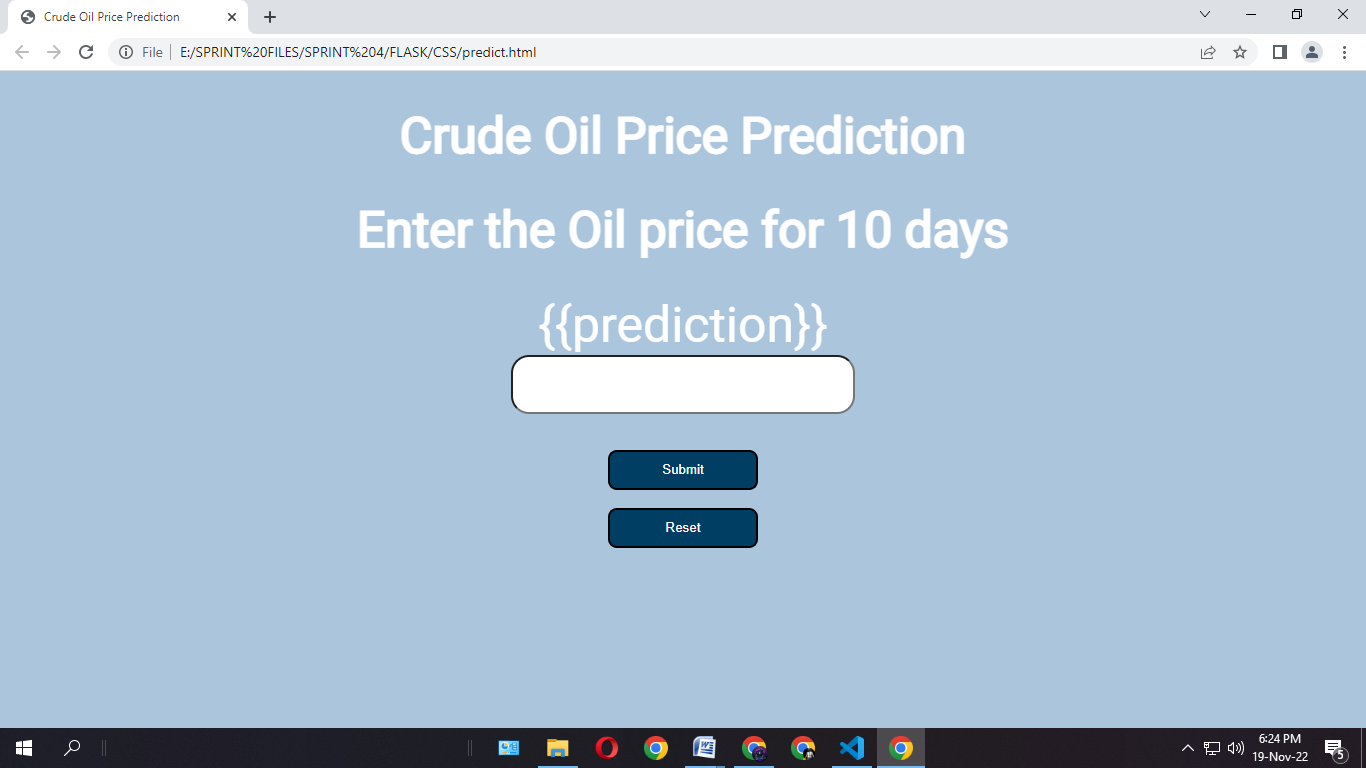
        return render\_template('predict.html')

if \_\_name\_\_=="\_\_main\_\_":

    model = load\_model('model.h5')

    app.run(host='0.0.0.0', port=5000)

**13.3 OUTPUT**

****

**14. GITHUB LINK**

**https://github.com/IBM-EPBL/IBM-Project-42009-1660647700**