Forecasting Crude Price

Capstone Project

INTRODUCTION

- There are different types of crude oil in the market– the thick, unprocessed liquid that drillers extract below the earth – and others are more desirable than others.
- There are **two kinds** of Brent, WTI and Dubai/Oman.
- we need benchmarks to value the commodity based on its quality and location, In our case WTI (West Texas Intermediate) would be used.
- **“Brent”** actually refers to oil from four different fields in the North Sea: Brent, Forties, Oseberg and Ekofisk. Crude from this region is light and sweet, making them ideal for the refining of diesel fuel, gasoline.

★ My Goal is to build a **Time Series model** to predict bunker price using Brent Crude. With the aid of **Brent Crude** Datasets. I would explore the commodities prices and determine the best model.

TABLE OF CONTENTS

01

Data Exploration

Datasets:

- Brent Crude Price

  Source: Energy Information Administration

Data Cleaning

Overview

dataframe shape
(8815, 2)

dataframe types

Date	object
Price	float64
dtype: object	

Missing Values

missing values

Date	0
Price	0
dtype: int64	

duplicate values
0

As shown Data has no Missing values and Duplicates , but note there is no data on weekends.

Describe Data

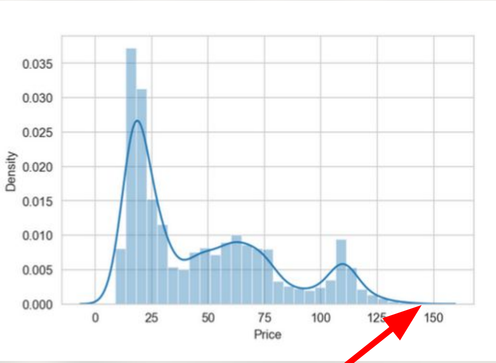
dataframe describe

	Price
count	8815.000000
mean	47.157345
std	32.052617
min	9.100000
25%	18.950000
50%	35.720000
75%	68.415000
max	143.950000

Max value being priced at 144

Data Transformation

Distribution plot



Our Outlier , 144 is very small in term of Density. This would not affect our analysis

Convert date to datetime

	Date	Price
0	05/20/1987	18.63
1	05/21/1987	18.45
2	05/22/1987	18.55
3	05/25/1987	18.60
4	05/26/1987	18.63

Before

	Date	Price
0	1987-05-20	18.63
1	1987-05-21	18.45
2	1987-05-22	18.55
3	1987-05-25	18.60
4	1987-05-26	18.63

After

Convert to date column to index

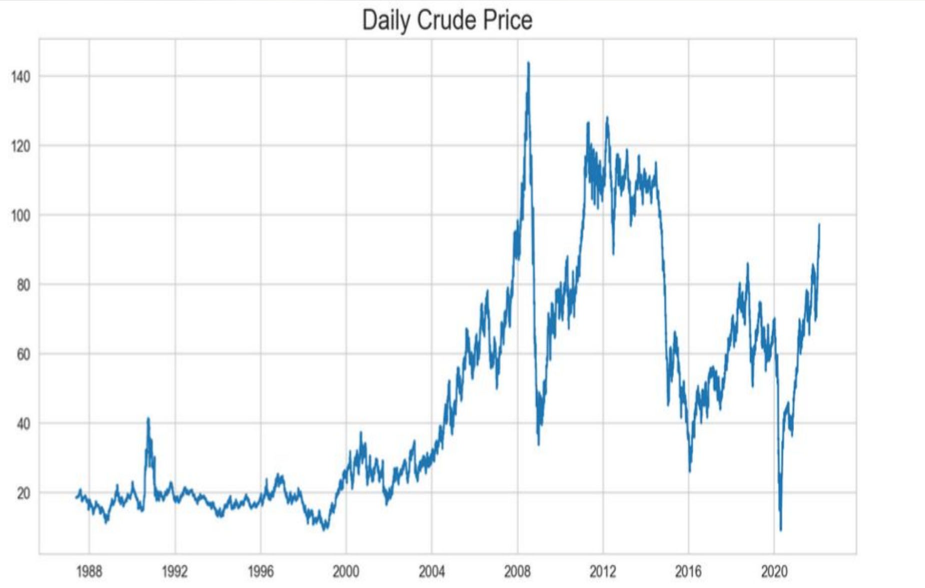
Date	Price
1987-05-20	18.63
1987-05-21	18.45
1987-05-22	18.55

EDA
Exploratory Data analysis

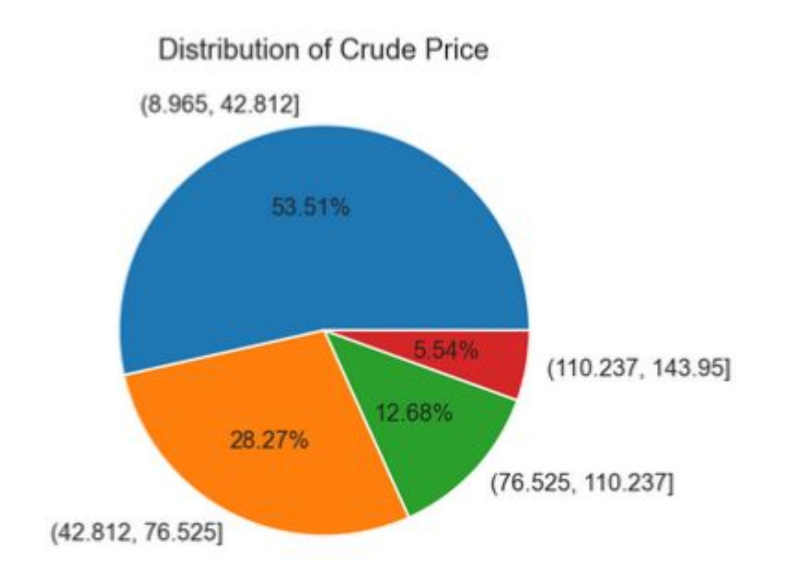
- Pie Chart
- Pairplots

Visualize Data

Plotting Data



Crude Price over 20 years. As you can see the data have some fluctuation

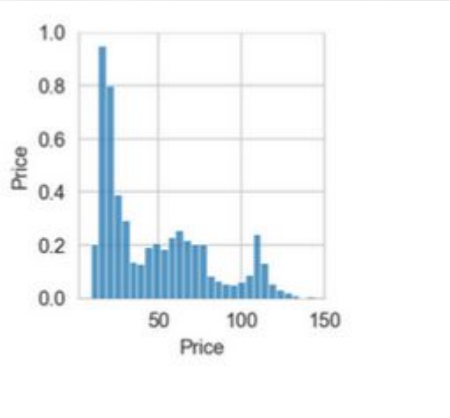


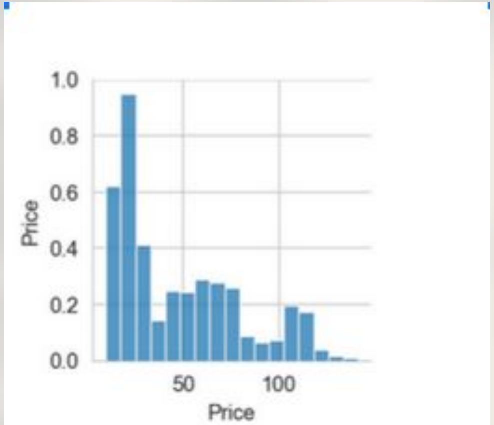
Pie chart

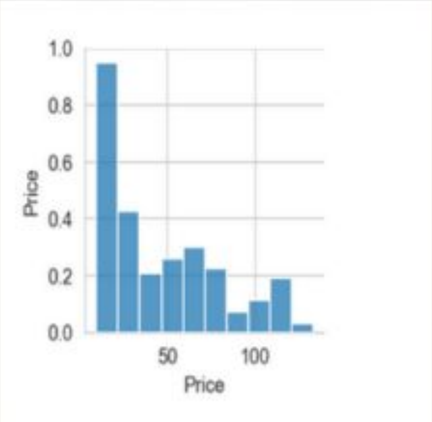
- 53.51% - 8.9 to 43
- 5.54% 110 to 144

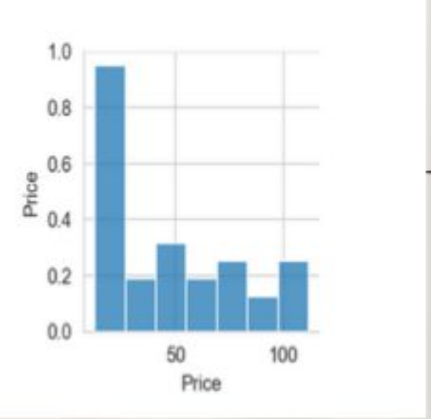
Pairplots of Prices

Degree of Granularity









Daily avg

Date	Price
1987-05-20	18.63
1987-05-21	18.45
1987-05-22	18.55
1987-05-25	18.60
1987-05-26	18.63

Weekly avg

Date	Price
1987-05-24	18.543333
1987-05-31	18.602000
1987-06-07	18.702000
1987-06-14	18.754000
1987-06-21	19.007500

Monthly avg

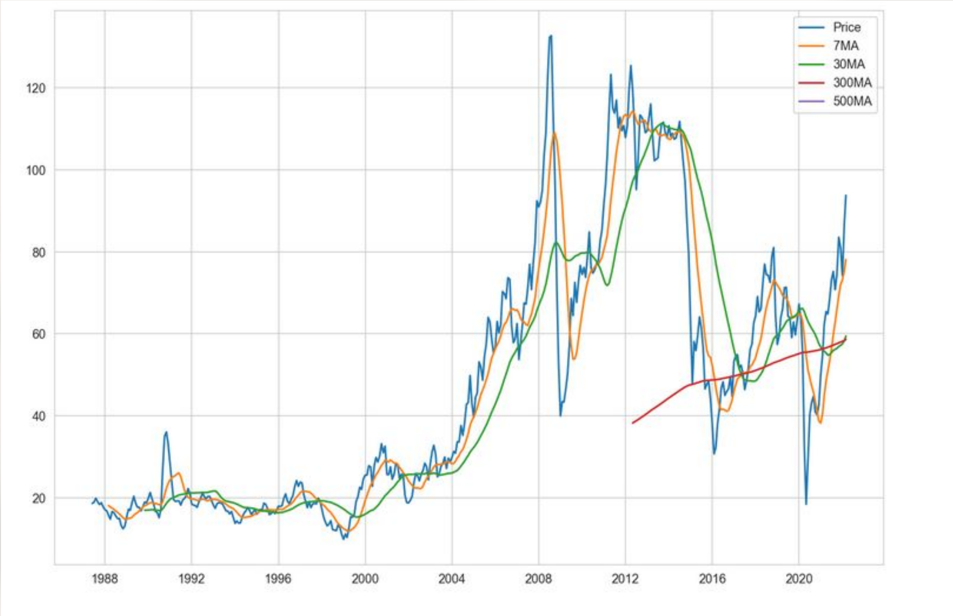
Date	Price
1987-05-31	18.580000
1987-06-30	18.860476
1987-07-31	19.856522
1987-08-31	18.979524
1987-09-30	18.313182

Yearly avg

Date	Price
1987-12-31	18.525813
1988-12-31	14.905412
1989-12-31	18.228228
1990-12-31	23.761445
1991-12-31	20.041128

Yearly Price
100
80
60
40
20
1988
1992
1996
2000
2004
2008
2012
2016
2020
2024

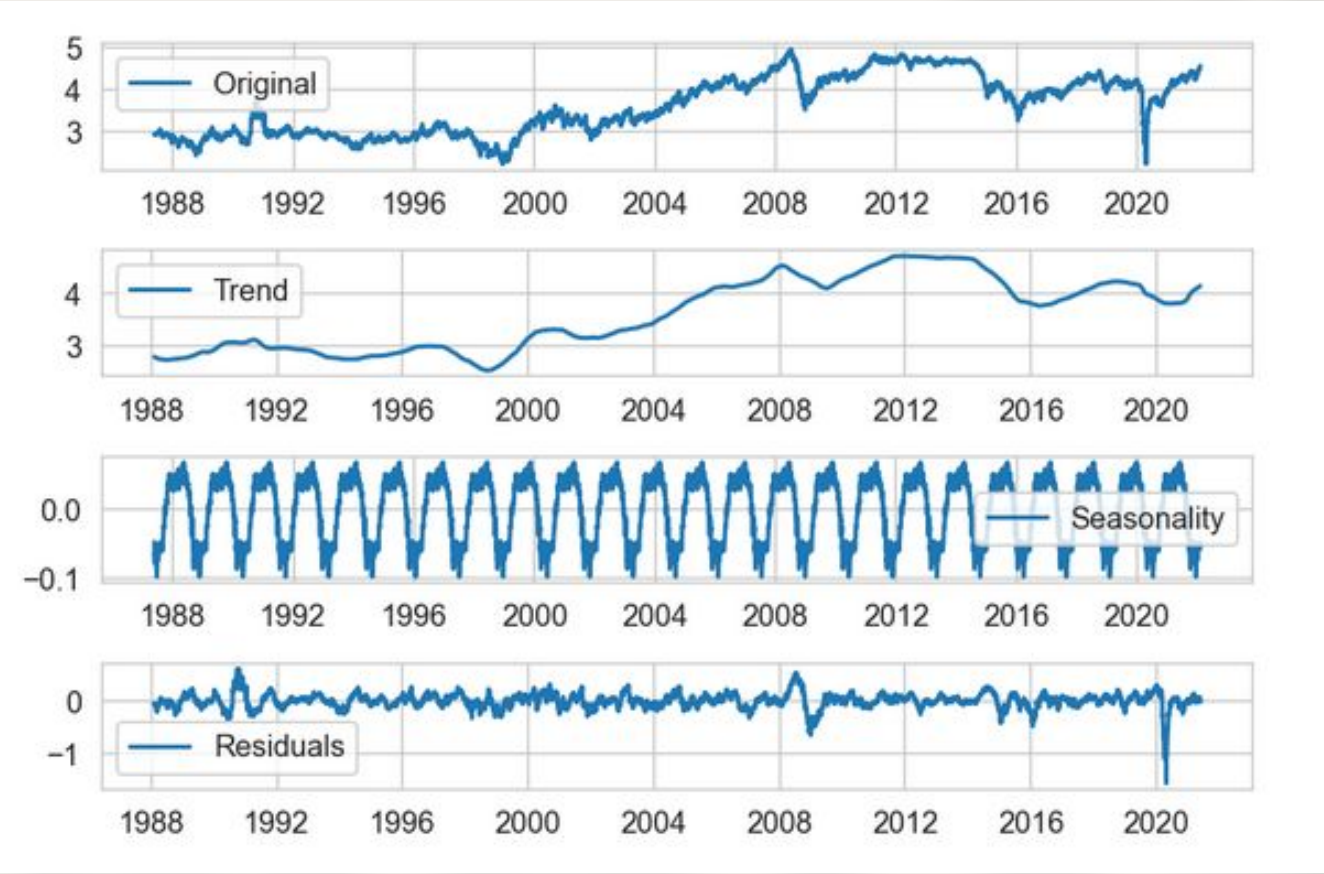
Moving Average



Daily

Monthly

Seasonal Decompose

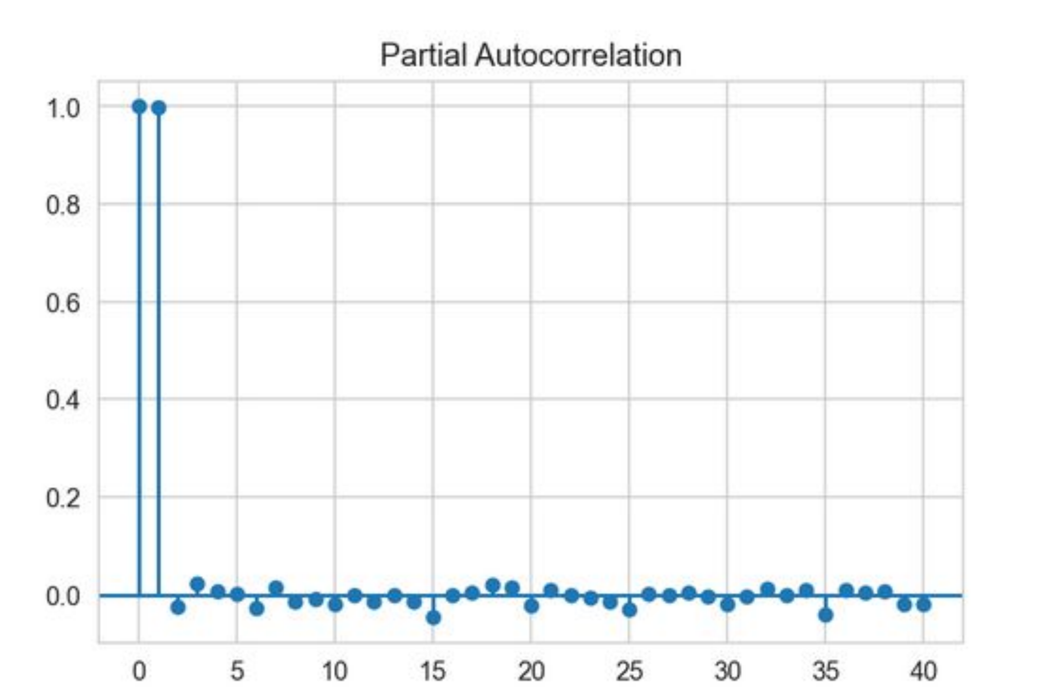


Data increasing Trend

Data pattern

Residuals is the Noise
which will affect our model

Partial AutoCorrelation function



- PACF is apply to determine what order of model
- We know our model is strongly correlated between 0 and 1 lags as diagram shows a 1.0

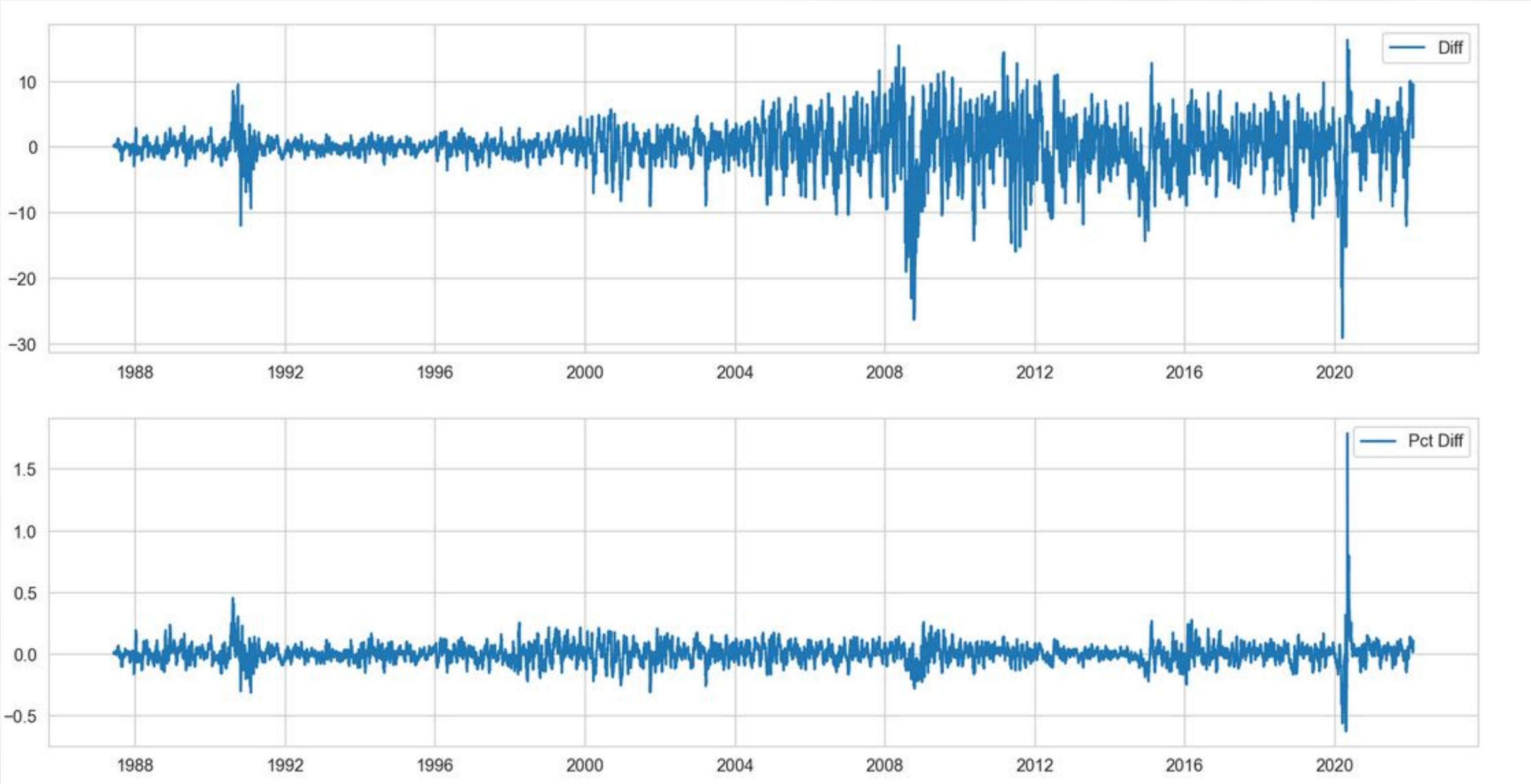
Augmented Dickey–Fuller test

ADF: -1.8997715641726658
P-value: 0.3321685817750109
Number of lags: 35
Number of Observation used for ADF Regression & Critical value Calculation: 8779
Critical value:

1% : -3.43109509774388
5% : -2.8618692838372026
10% : -2.566945272806482

P-value = 0.3 > 0.05, means dataset is not stationary

Differencing Data



Train -Test Split

Train test sets

- y_train : 2000 to 2007
- y_test : 2008

```
(8815, 1)
(2046, 1) (253, 1)
```

- Why 2000 to 2007?

Auto ARIMA

```
Performing stepwise search to minimize aic
 ARIMA(0,1,0)(0,0,0)[12] intercept   : AIC=5546.136, Time=0.30 sec
 ARIMA(1,1,0)(1,0,0)[12] intercept   : AIC=5548.524, Time=0.30 sec
 ARIMA(0,1,1)(0,0,1)[12] intercept   : AIC=5548.614, Time=0.35 sec
 ARIMA(0,1,0)(0,0,0)[12]             : AIC=5546.836, Time=0.04 sec
 ARIMA(0,1,0)(1,0,0)[12] intercept   : AIC=5546.554, Time=0.25 sec
 ARIMA(0,1,0)(0,0,1)[12] intercept   : AIC=5546.644, Time=0.26 sec
 ARIMA(0,1,0)(1,0,1)[12] intercept   : AIC=5548.051, Time=0.87 sec
 ARIMA(1,1,0)(0,0,0)[12] intercept   : AIC=5548.096, Time=0.19 sec
 ARIMA(0,1,1)(0,0,0)[12] intercept   : AIC=5548.098, Time=0.24 sec
 ARIMA(1,1,1)(0,0,0)[12] intercept   : AIC=5542.902, Time=0.86 sec
 ARIMA(1,1,1)(1,0,0)[12] intercept   : AIC=5549.737, Time=1.07 sec
 ARIMA(1,1,1)(0,0,1)[12] intercept   : AIC=5549.818, Time=1.04 sec
 ARIMA(1,1,1)(1,0,1)[12] intercept   : AIC=5551.305, Time=1.62 sec
 ARIMA(2,1,1)(0,0,0)[12] intercept   : AIC=5550.518, Time=0.58 sec
 ARIMA(1,1,2)(0,0,0)[12] intercept   : AIC=5550.523, Time=0.28 sec
 ARIMA(0,1,2)(0,0,0)[12] intercept   : AIC=5548.832, Time=0.22 sec
 ARIMA(2,1,0)(0,0,0)[12] intercept   : AIC=5548.795, Time=0.20 sec
 ARIMA(2,1,2)(0,0,0)[12] intercept   : AIC=5552.509, Time=0.68 sec
 ARIMA(1,1,1)(0,0,0)[12]             : AIC=5543.626, Time=0.42 sec

Best model:  ARIMA(1,1,1)(0,0,0)[12] intercept
```

Auto ARIMA Plot

- Making prediction on Test set

LSTM (Long short-term memory)

	Target Date	Target-3	Target-2	Target-1	Target
0	1987-05-25	18.63	18.45	18.55	18.60
1	1987-05-26	18.45	18.55	18.60	18.63
2	1987-05-27	18.55	18.60	18.63	18.60
3	1987-05-28	18.60	18.63	18.60	18.60
4	1987-05-29	18.63	18.60	18.60	18.58
...	...	...	...	...	...
8807	2022-02-01	90.70	91.47	92.35	90.24
8808	2022-02-02	91.47	92.35	90.24	91.43
8809	2022-02-03	92.35	90.24	91.43	92.99
8810	2022-02-04	90.24	91.43	92.99	96.86
8811	2022-02-07	91.43	92.99	96.86	97.28

8812 rows × 5 columns

Years includes: 1987-05-25 to 2022-02-07

Converting to shifting 3 days back for the price label as Target-3 ,2,1

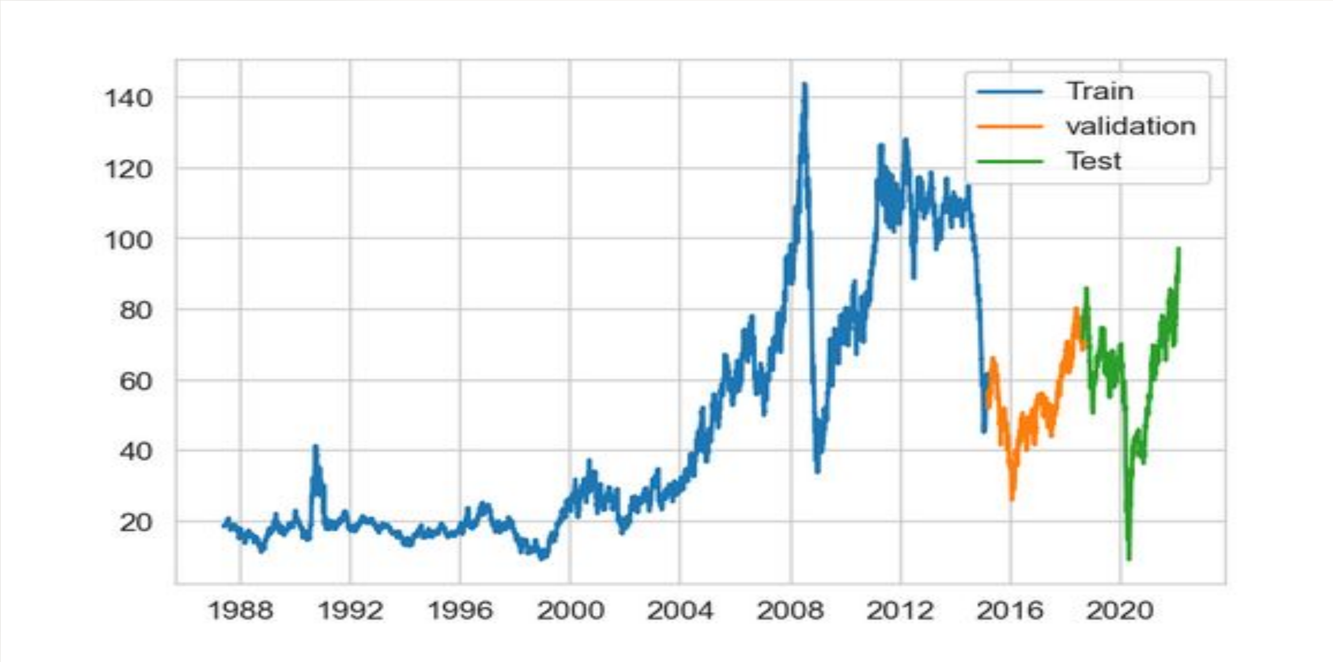
Converting to array

```
def win_df_date_X_y(win_dataframe):
    df_as_np = win_dataframe.to_numpy()
    dates= df_as_np[:,0]
    middle_matrix=df_as_np[:,1:-1]
    X=middle_matrix.reshape((len(dates),middle_matrix.shape[1],1))

    Y=df_as_np[:,-1]
    return dates, X.astype(np.float32),Y.astype(np.float32)
```

```
((8812,), (8812, 3, 1), (8812,))
```

Train-Test sets




```
q_80 = int(len(dates) *.8)
q_90 =int(len(dates) *.9)
```

```
dates_train, X_train, y_train = dates[:q_80],X[:q_80],y[:q_80]
dates_val, X_val, y_val = dates[q_80:q_90],X[q_80:q_90],y[q_80:q_90]
dates_test, X_test, y_test = dates[q_90:],X[q_90:],y[q_90:]
```

Modeling LSTM

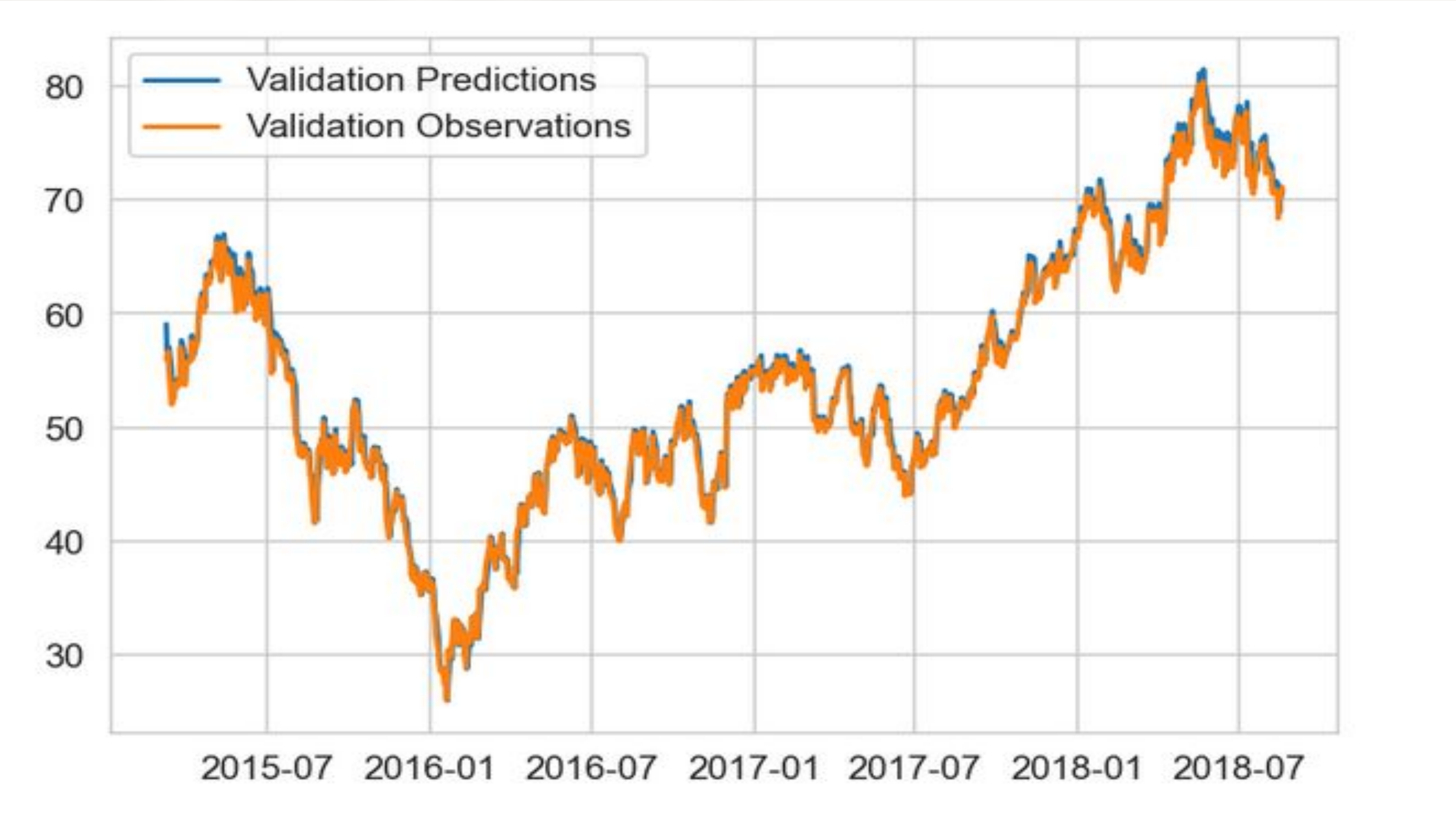
```
from keras.models import Sequential
from tensorflow.keras.optimizers import Adam
from tensorflow.keras import layers
```

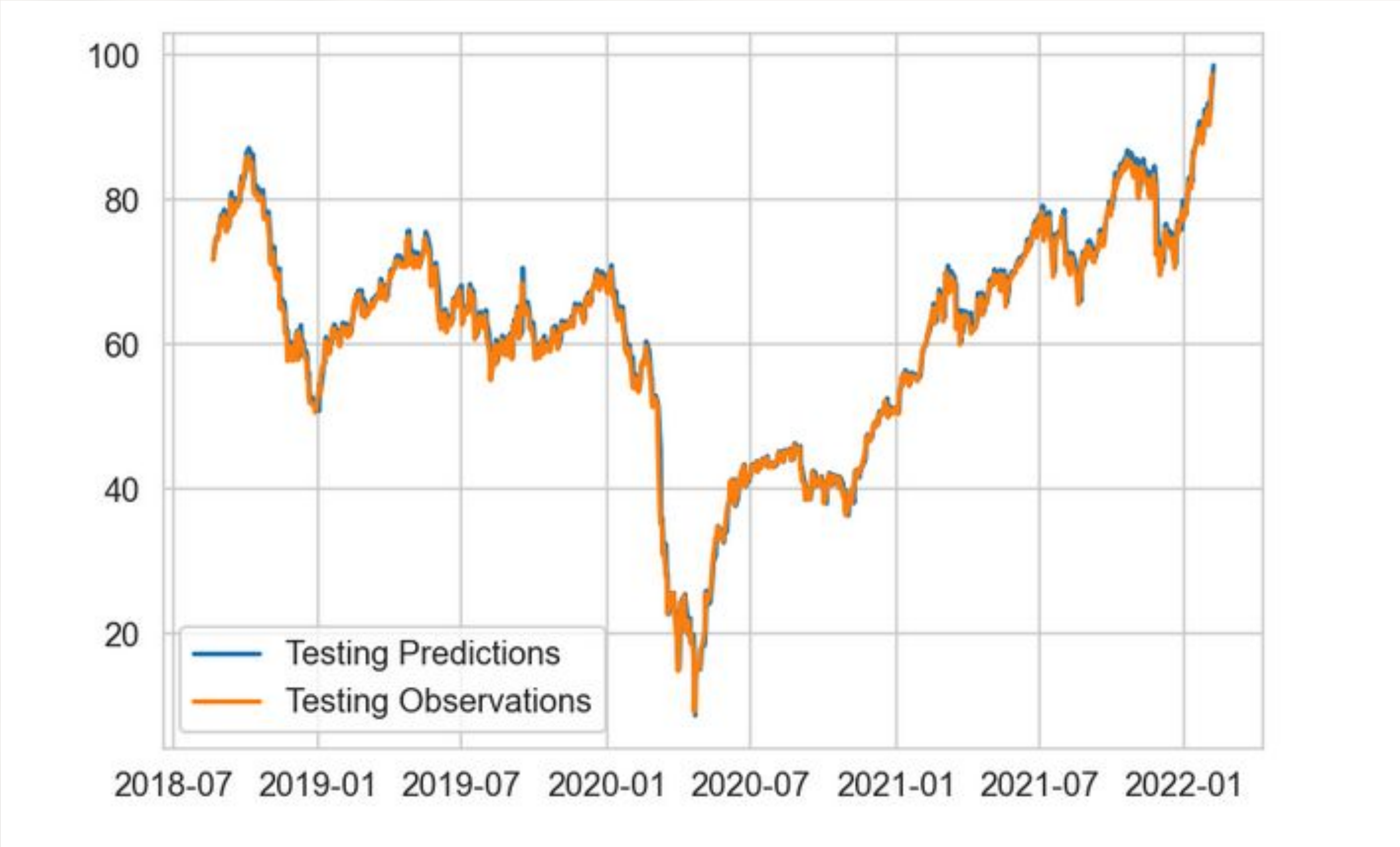
```
model= Sequential([layers.Input((3,1)),
                layers.LSTM(60),
                layers.Dense(32,activation ='relu'),
                layers.Dense(32,activation ='relu'),
                layers.Dense(1)])
model.compile(loss='mse',
                optimizer=Adam(learning_rate=0.001),
                metrics=['mean_absolute_error'])
```

```
model.fit(X_train, y_train, validation_data=(X_val,y_val), epochs=100, )
```

```
Epoch 69/100
221/221 [==============================] -
absolute_error: 0.8969
Epoch 70/100
221/221 [==============================] -
absolute_error: 0.8785
Epoch 71/100
221/221 [==============================] -
absolute_error: 0.8848
Epoch 72/100
221/221 [==============================] -
absolute_error: 0.8849
Epoch 73/100
221/221 [==============================] -
```


Training Predictions
Training Observations
140
120
100
80
60
40
20
1988
1992
1996
2000
2004
2008
2012
2016


Validation Predictions
Validation Observations
80
70
60
50
40
30
2015-07
2016-01
2016-07
2017-01
2017-07
2018-01
2018-07

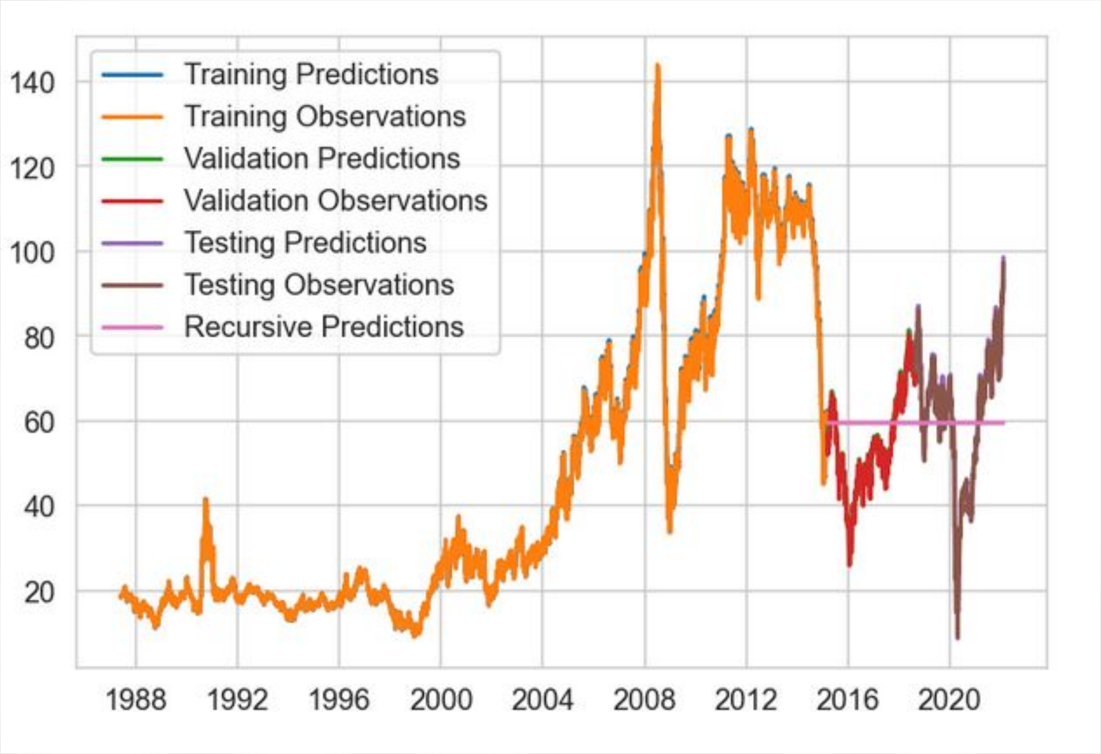

100
80
60
40
20
2018-07
2019-01
2019-07
2020-01
2020-07
2021-01
2021-07
2022-01
Testing Predictions
Testing Observations

Recursive Prediction

```
recursive_predictions = []
recursive_dates =
np.concatenate([dates_val, dates_test])

for target_date in recursive_dates:
  last_window = deepcopy(X_train[-1])
  next_prediction =
model.predict(np.array([last_window])).flatte
n()

recursive_predictions.append(next_predictio
n)
  last_window[-1] = next_prediction
```



Conclusion

- Explore Model used with **Generative adversarial network GAN**
- Explore Project in a Classification point of view
- Include other variables like stock of Oil and Gas companies to do multivariate analysis

THANKS!