Multi-Time Series Averaging of Ensemble Machine Learning Models Towards Crude Oil Price Forecasting

Farid Javadnejad, Ph.D.[1]

1. Introduction

Crude oil and other refined liquid products from fossil fuels are critical contributors to the world economy. Petroleum has been the largest energy source for all countries. Its products run vehicles, heat buildings, and produce electricity. Moreover, various industries use petroleum as a raw material to produce intermediate or end-user products that we use daily (EIA 2022b; Lu et al. 2021; Deng, Ma, and Zeng 2021; Kilian and Murphy 2014). In 2019, global petroleum consumption neared 100 million barrels per day (Table 1).

Table 1. The largest oil consumers and their share of total world consumption (EIA 2022b)

Ranking	Country	Million barrels per day	Share of world total
1	United States	20.54	20%
2	China	14.01	14%
3	India	4.92	5%
4	Japan	3.74	4%
5	Russia	3.70	4%
6	Saudi Arabia	3.18	3%
7	Brazil	3.14	3%
8	Canada	2.63	3%
9	South Korea	2.60	3%
10	Germany	2.35	2%
	World total	100.23	

[1] Geospatial Consultant, Aerogeospatial LLC, 1401 21st Street Suite R, Sacramento, CA 95811, e-mail: nejad.fj@gmail.com.

1.1. Problem statement

Crude oil prices are difficult to predict accurately due to the number of influencing factors and the highly complex behavior of such influences. Global economic and social activities can be substantially impacted by fluctuations in crude oil prices. Threfore, despite challenges for prediction of oil price, accurate oil price forecasting is crucial for decision-making support for the manufacturing, logistics, and government sectors to guide industrial and social policies and practices (Kilian and Murphy 2014; Deng, Ma, and Zeng 2021; Lu et al. 2021).

1.2. Background

Brent, West Texas Intermediate (WTI), Dubai/Oman, and Shanghai crude oil prices are the major benchmarks of the crude oil market and are reported is USD per barrel unit. Factors such as supply and demand, financial markets and economics, politics, global events, renewable energy and alternative resources, new resources and development of new oil extraction technologies, social & environmental policies, and consumption patterns may influence the crude oil market dynamics. Such impacts and resultant price fluctuations might be very complex and may occur at different frequencies.

Classical econometric models such as random walk, autoregressive integrated moving average (ARIMA), error correction model (ECM), generalized autoregressive conditional heteroscedasticity (GARCH) model are used for crude oil price prediction. Recently, machine learning (ML) methods such as artificial neural network (ANN) and support vector machine (SVM) are used for the crude oil price prediction, which provide powerful tools to model nonlinear behavior or crude oil market dynamics (Jammazi and Aloui 2012; Lanza, Manera, and Giovannini 2005; Hou and Suardi 2012; Basiri 2015; Yu, Zhao, and Tang 2017; Murat and Tokat 2009; Kilian and Murphy 2014; Javadnejad 2012).

1.3. Objectives

In this work, an ML model is proposed to predict crude oil price using multiple infulencing factors. The predictions are casted on multiple time-series to consider for complex factors that imact the market dynamics in differenct frequencies.

This report is structured as follows. Section describes the metholdogy, where Sub-section 2.1 summarizes the datasets that are used in this study. Sub-section 2.2 describes the data preparation and data wrangling procedures. In Sub-section 2.3, the exploratory data analysis and feature engineering approcahes for ML training are described. Sub-section 2.4 covers the ML pre-processing, training, models selection, model metrics. In Sub-section 2.5, the final results for model training and validation are presented, as well as the predictions for 6-month time frequencies. We present the discussion of our results in Section 3. Finally, in Section 4, we summarise our recoomendaiton for future work.

2. Methodology

2.1. Datatets

The factors that influence the crude oil market dynamics include supply and demand, financial markets, politics, global events, alternative resources, development technologies, policies, and consumption patterns (Hamilton 2008; Hamilton 2009; Kilian and Murphy 2014; Zhao, Li, and Yu 2017; Lu et al. 2021; Wang, Wu, and Yang 2015). We use the crude oil prices of West Texas Intermediate (WTI) benchmark as the target feature. To take into account the aforementioned influencing factors a total of 32 feature variables were selected from publicly accessible data sources (EIA 2022a; FRED 2022; Investing 2022; WSJ 2022). Table 2 provides a list of the selected features, a description about each feature, and the sources of data.

Table 2. Selected dataset of feature variables for crude oil price

Category	Symbol	Variable	Unit	Source
Crude Oil Price	WTIPUUS	West Texas Intermediate Crude Oil Price	dollars per barrel	EIA
Supply	COPR_OPEC	Crude Oil Production, Total OPEC	million barrels per day	EIA
	PAPR_NONOPEC	Crude Oil Production, Total non-OPEC	million barrels per day	EIA
	INTL.55-1-WORL-TBPD	Crude Oil Production, NGPL, and other liquids production, World	thousand barrels per day	EIA
	COPRPUS	Crude Oil Production, U.S.	million barrels per day	EIA
Replacement Cost	RNGWHHD	Henry Hub Natural Gas Spot Price	dollars per million btu	EIA
Demand	PATC_OECD	Liquid Fuels Consumption, Total OECD	million barrels per day	EIA
	PATC_NON_OECD	Liquid Fuels Consumption, Total non-OECD,	million barrels per day	EIA
	FEDFUNDS	Federal Funds Effective Rate	percent, not seasonally adjusted	FRED
	IGREA	Index of Global Real Economic Activity	index, not seasonally adjusted	FRED
	CICPIUS	US Consumer Price Index (CPI): All Commodities	index, 1982-1984=1.00	EIA
	USACPIENGMINMEI	US Consumer Price Index (CPI): Energy for the United States	index 2015=100, not seasonally adjusted	FRED
	WPCPIUS	US Producer Price Index (PPI): All Commodities	index, 1982=1.00	EIA
	WP57IUS	US Producer Price Index (PPI): Petroleum	index, 1982=1.00	EIA
	EA19PIEAMI01GPM	roducer Price Index (PPI) of Euro Area (19 Countries)	index 2015=100, not seasonally adjusted	FRED
	ZOMNIUS	US Manufacturing Production Index (PMI)	index, 2017=100 (seasonally adjusted)	EIA
Inventory	PASC_OECD_T3	Petroleum Inventory, Total OECD	million barrels, end-of-period	EIA
	PASXPUS	Petroleum Inventory, US Total	million barrels, end-of-period	EIA
	COSQPUS	US Crude Oil Inventory: Strategic Petroleum Reserve (SPR)	million barrels, end-of-period	EIA
	COSXPUS	US Crude Oil Inventory: Non-SPR	million barrels, end-of-period	EIA
Monetary Market	RTWEXBG	Real Broad Dollar Index	index Jan 2006=100, not seasonally adjusted	FRED
	DXY	US Dollar Index (DXY)	index	Investing
	DEXUSEU	U.S. Dollars to Euro Spot Exchange Rate (DEXUSEU)	US dollars to one euro, not seasonally adjusted	FRED
Stock Market	SPX	S&P 500 Index	index	WSJ
	DJI	Dow Jones Industrial Index	index	WSJ
	COMP	NASDAQ index	index	WSJ
Commodity Market	Gold_Future	Gold Futures Historical Data	dollar per ounce	Investing

	Copper_Future	Copper Futures Historical Data	dollar per pound	Investing
Policy Uncertainty	GEPUCURRENT	Global Economic Policy Uncertainty Index: Current Price Adjusted GDP	index, not seasonally adjusted	FRED
Technology	MGWHUUS	Refiner Wholesale Gasoline Price	cents per gallon	EIA
	DSWHUUS	Diesel Fuel Refiner Wholesale Price	cents per gallon	EIA
	BREPUUS	Brent Crude Oil Spot Price	dollars per barrel	EIA

2.2. Data Cleaning and Data Wrangling

We used Jupyter Notebook 6.5.2 (Kluyver et al. 2016) and Python 3.9.15 (Python Software Foundation 2022) to process the data. The features in Table 2 were read through APIs (if available) or were downloaded directly from the data source. The data were initially set to be imported with monthly intervals or averaged to monthly values, then were limited to the target time frame of January 2000 and December 2022. The features were index based on their date values and then all merged together on the date values to create corresponding feature values for each month. Figure 1 shows the monthly West Texas Intermediate (WTI) crude oil price in the target time frame.

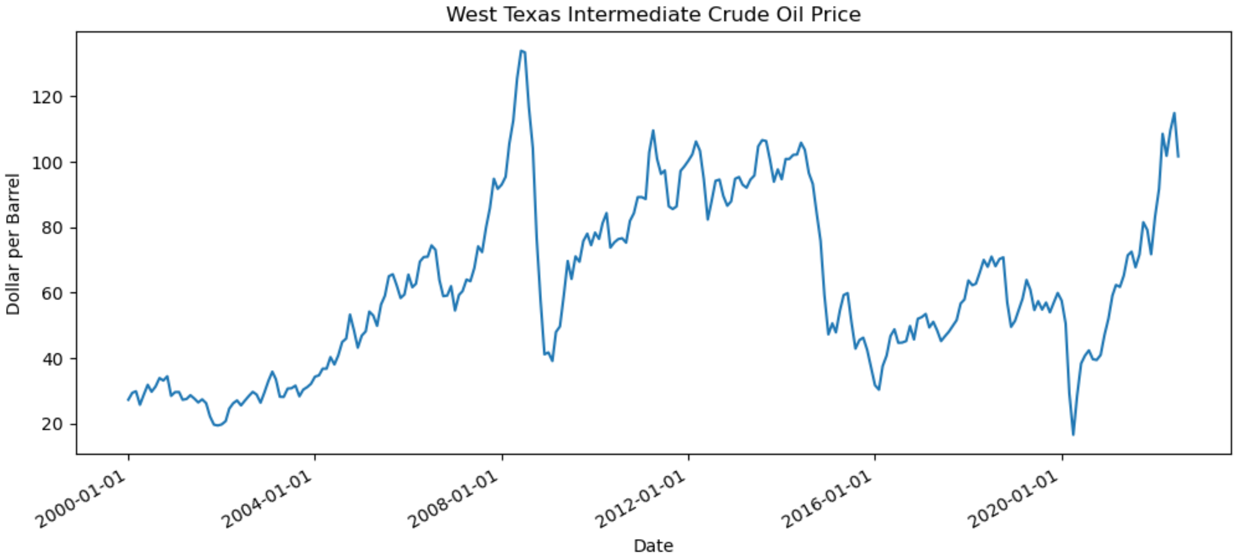


Figure 1. West Texas Intermediate (WTI) crude oil price

We used Pandas 1.5.2 (McKinney 2011) that is an open-source, simple, powerful, and flexible library for data analysis and data manipulation, Pandas is for Python programming language. The final dataset sized (274, 32) that represent 274-month records (rows) for 32 feature variables (columns).

An important step of data wrangling is dealing with missing data. Table 3 shows the summary statistics of missing data in feature variables. Missingno (Bilogur 2018) is also useful tool that provides a series of visualisations for presence and distribution of the missing data within a pandas dataframe. Figure 2 visually shows the distribution of the missing. To handle the missing data, features with more than the 10% of missing data were dropped from the dataset. The columns that had less than 1% missing features were imputed by using back and forward fill methods. For the remaining missing data between 1% and 10%, the rows for all features were dropped to create a dataset with no missing data. After treating missing data, the final dataframe sized (271, 30).

Table 3. Summary statistics of missing data in feature variables

Variable	Count	Percentage
oil_production_world	3	1.1%
petroleum_inventory_oecd	36	13.1%
global_real_econ_activity_index	1	0.4%
us_cpi_energy	1	0.4%
eu19_ppi	2	0.7%
real_dollar_index	72	26.3%
global_econ_policy_uncert_index	1	0.4%

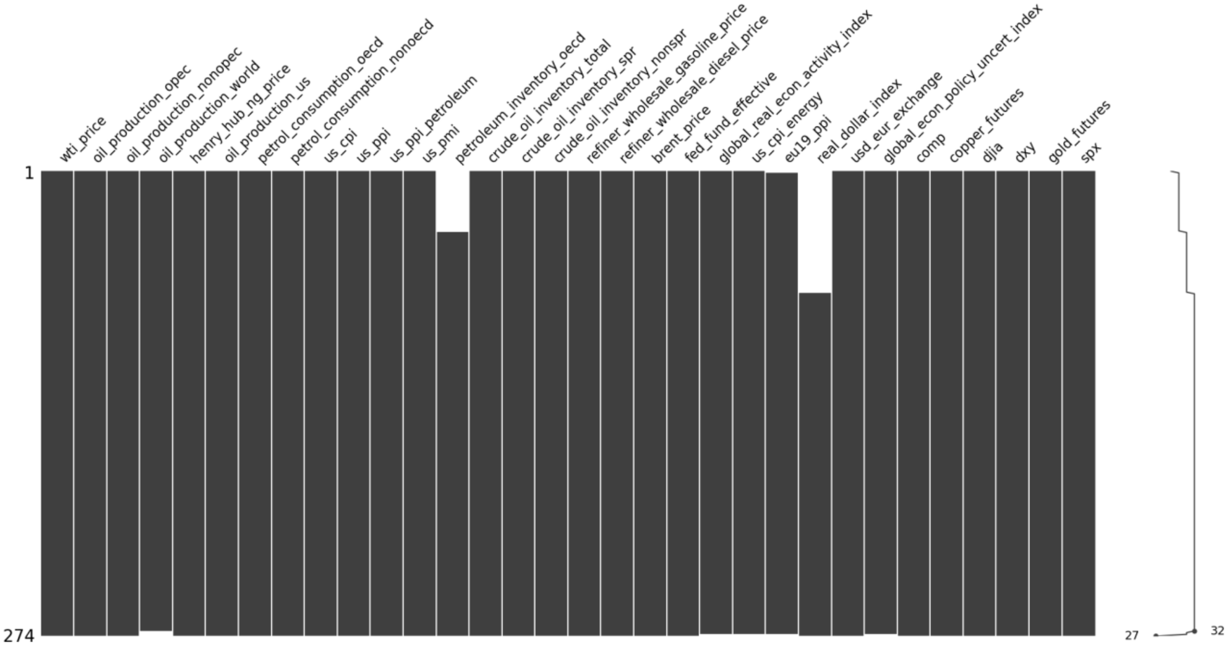


Figure 2. Missing data plot

2.1. Exploratory Data Analysis and Feature Engineering

WTI-Brent Spread is defined as difference between Brent and the WTI Crude Oil spot prices (Eq. 1). WTI is produced from the Gulf of Mexico basin and is the benchmark for oil price in American market. On the other hand, Brent Oil is produced from the North Sea and Atlantic basin and is the pricing benchmark for European market. While, the prices for these two benchmarks are highly correlated; difference between the prices often reflect technical, supply/demand or geopolitical issues.

$$\text{WTI-Brent spread} = \text{WTI spot price} - \text{Brent spot price} \tag{1}$$

The Crack spread is defined as the price difference between crude oil and its refined oil (Eq. 2 and 3), reflecting the supply and demand relationship between the crude oil market and its refined product market (Wang, Wu, and Yang 2015).

$$\text{WTI crack spread} = 3 \times \text{WTI spot price} - 2 \times \text{Gasoline Price} - 1 \times \text{Diesel Fuel Price} \tag{2}$$

$$\text{Brent crack spread} = 3 \times \text{Brent spot price} - 2 \times \text{Gasoline Price} - 1 \times \text{Diesel Fuel Price} \tag{3}$$

After calculating the spreads, Brent price, refiner wholesale gasoline price, and refiner wholesale diesel price were dropped from the dataset.

Figure 3 shows the histograms of initial feature variables and feature engineered variables. Figure 4 includes scatterplots of feature variable against the target variable of WTI oil price. Figure 5 and Figure 6 show the Pearson correlation coefficients matrix, and predictive power score (PPS) matrix for feature variables, respectively.

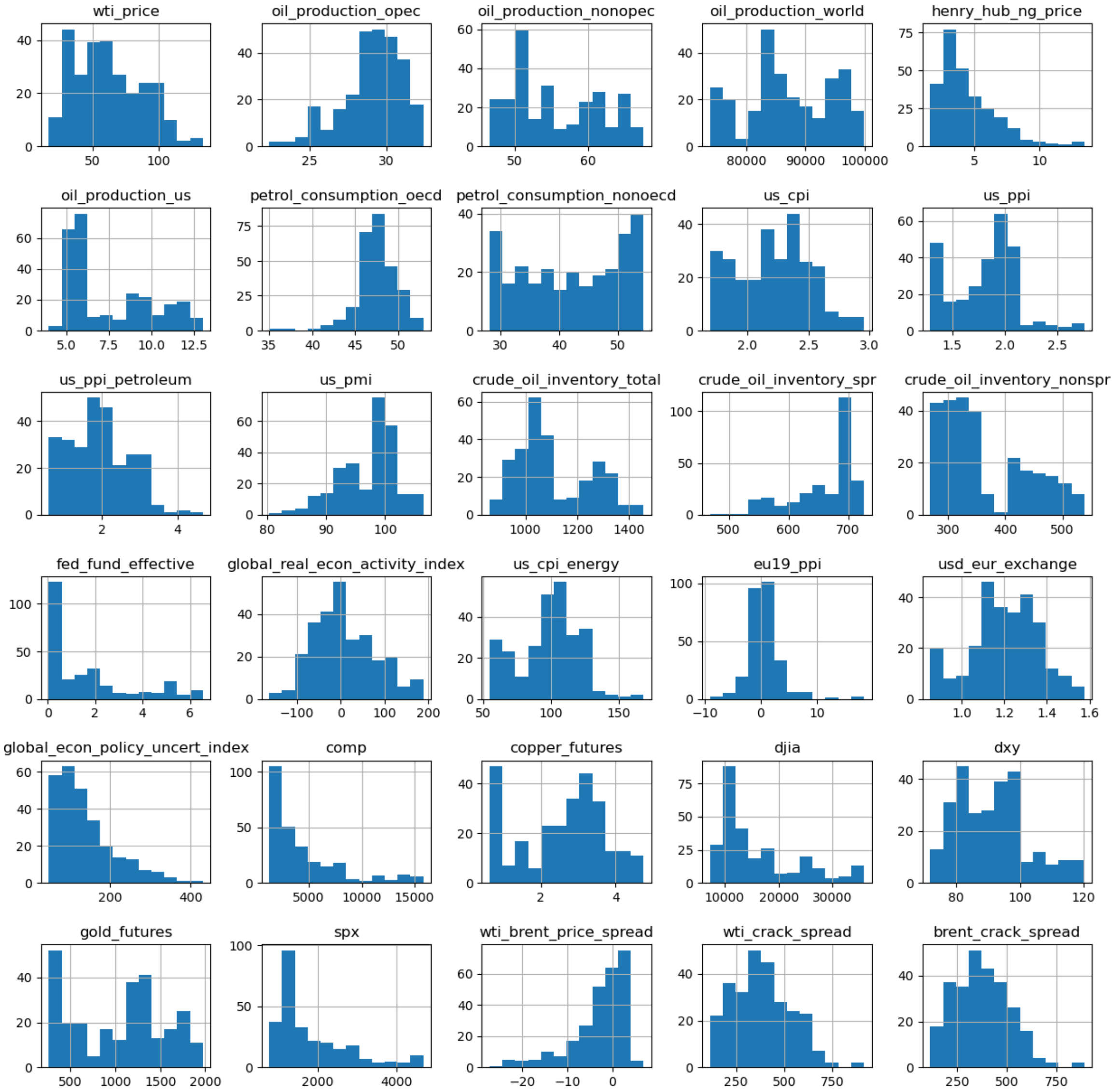


Figure 3. Histograms of feature variables

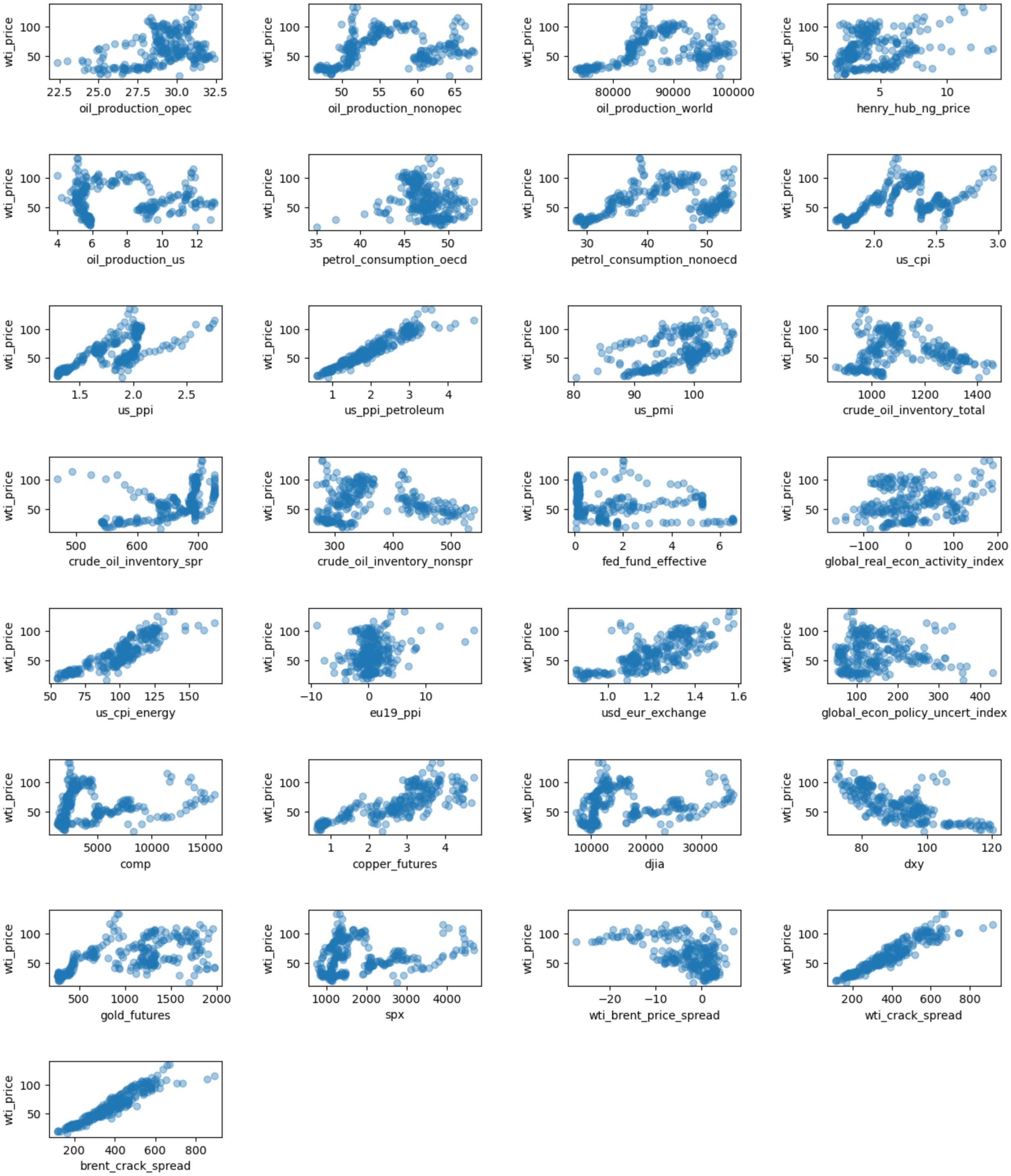


Figure 4. Scatter plots of feature variable against the target variable

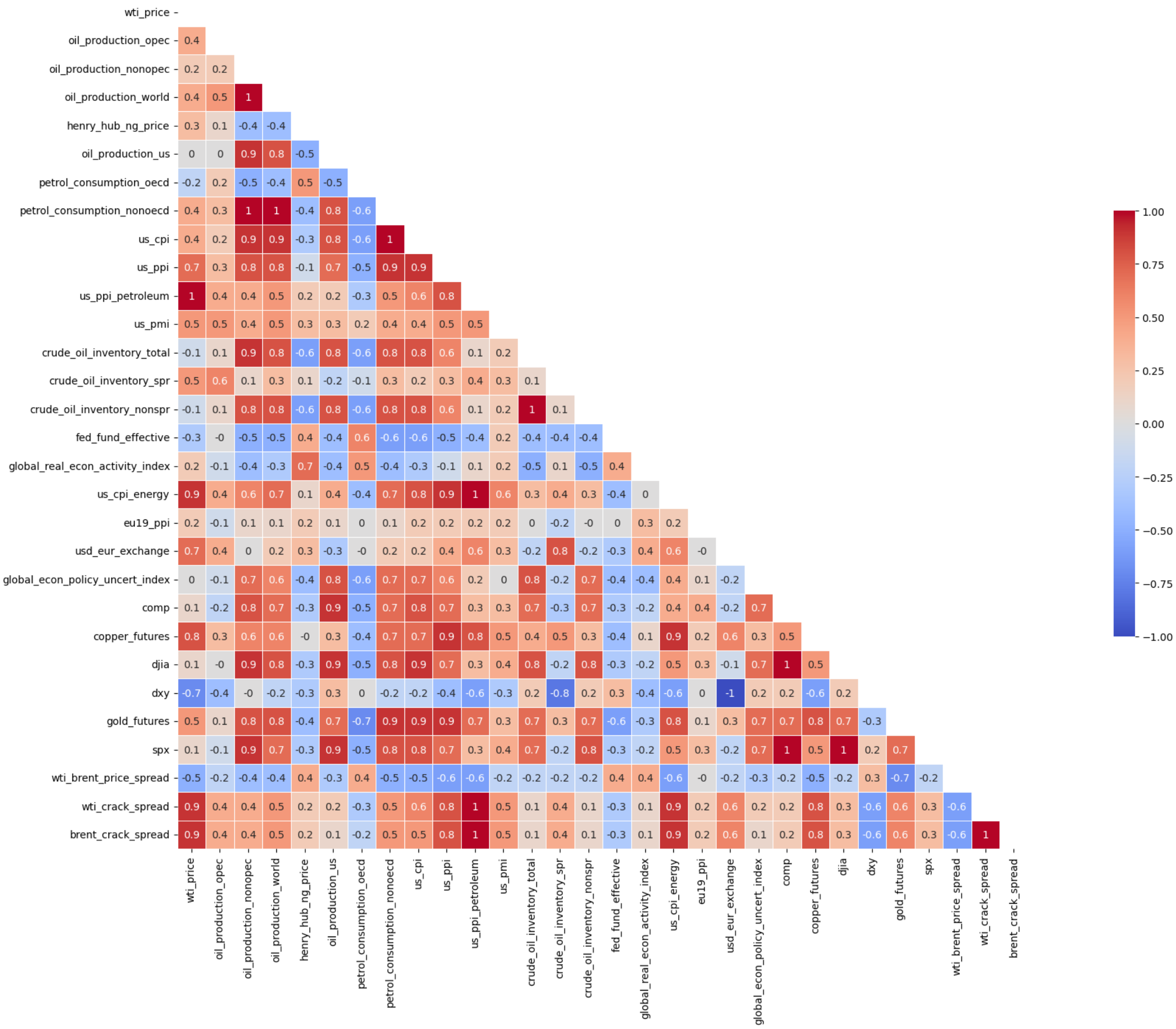


Figure 5. Pearson correlation coefficients matrix for feature variables

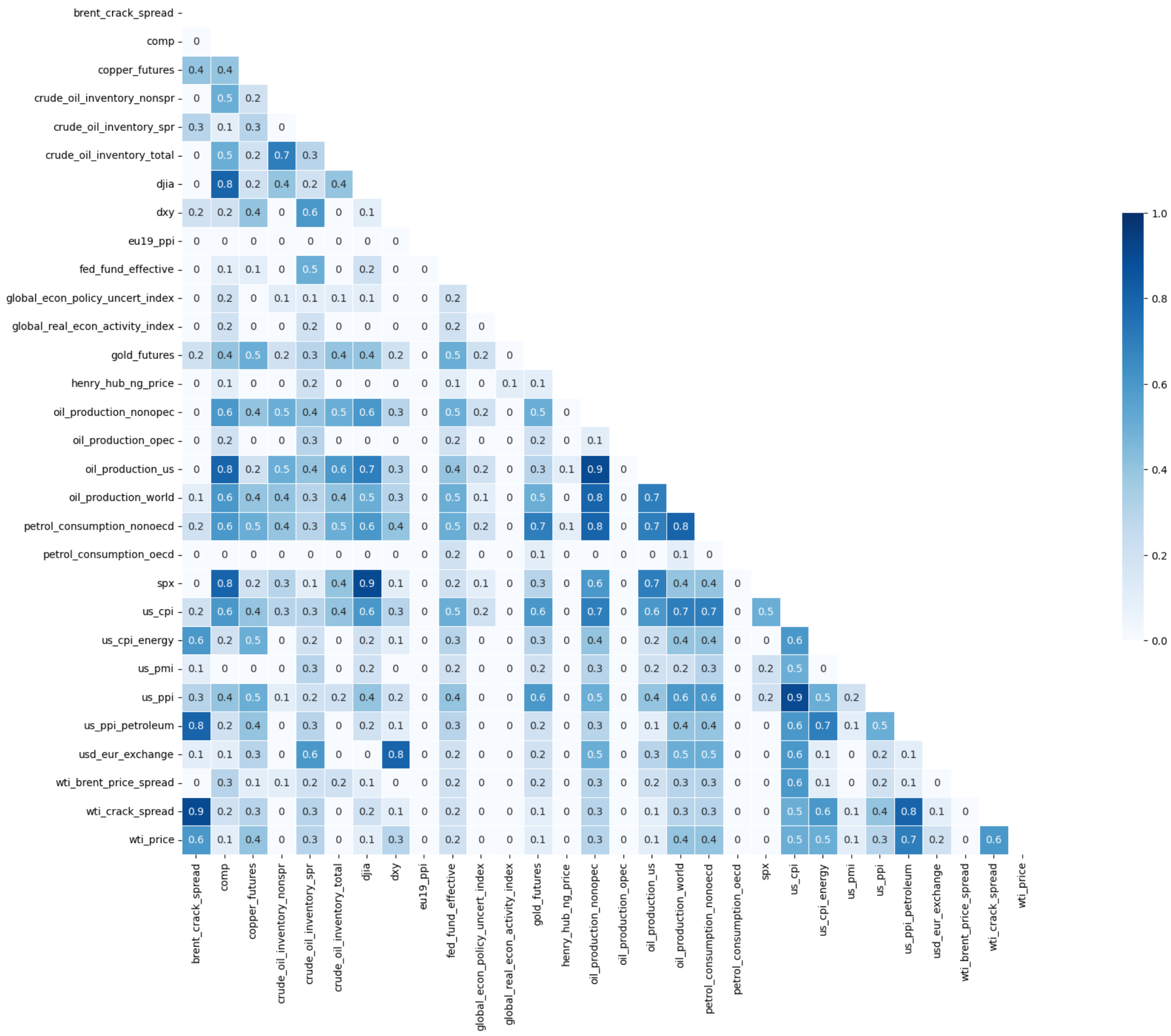


Figure 6. Predictive Power Score (PPS) matrix for feature variables

2.2. Pre-processing and Training

- *Pre-processing*

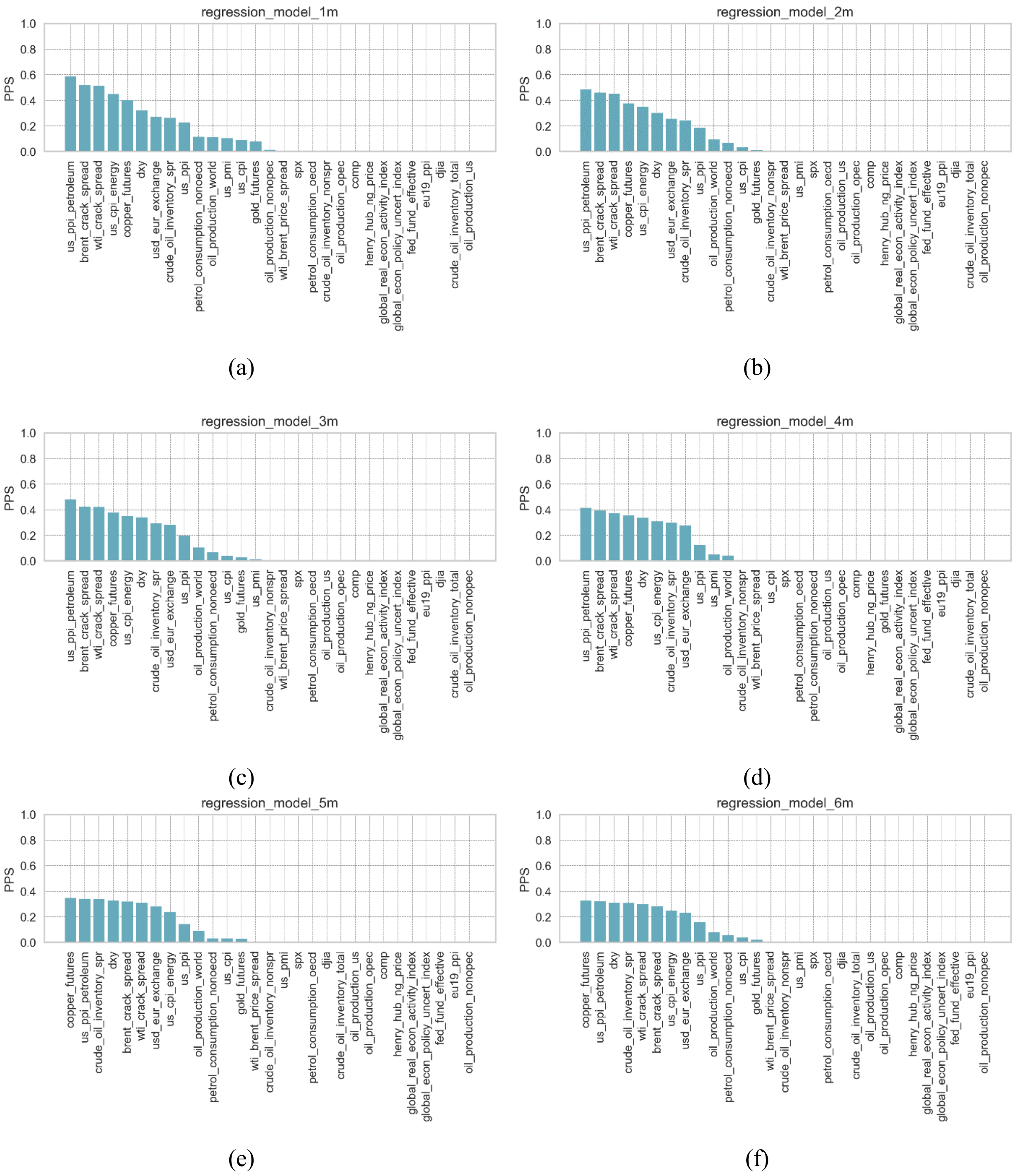
The objective of this time-series analysis is to predict the WTI Price (dependent variable) using the independent variables for a time period in the future, where the independent variables are also unknow. To handle this, a data preparation step is needed before modelling so that the dependent value is shifted in time to be associated with the independent variables of previous time period. The time shift operation is formulated in Eq. 4:

$$Y_t = f(X_{t-n}, \beta_t) + e_{t-n} \quad (4)$$

where Y_t is the dependent variable at time t , X is the independent variable, n is the shift operator, β is the regression coefficient, and e is the residual term.

We performed time shifting for frequency of $n = \{1, 2, 3, 4, 5, 6\}$ months. When we shift the dependable variable in time, we end up having n rows with missing values. So, it is necessary to handle the missing values by dropping the rows (as described is Sub-section 2.2).

After performing time shifting, the dependent variable will have a new set of independent variables that are specific to that frequency, and in each frequency a different variable may have stronger correlation in the WTI Price. In real world scenario, this can be described and the time-lag for an effect to make an impact on the WTI prices. For example, changes in some features may impact the WTI prices in short term, while some other features may impact the dependent variable in a longer term. With the proposed methodology is expected to capture features with impact frequencies of 1 to 6 months. Figure 7 shows the bar chart of sorted PPS coefficients of independent variables against WTI oil price for 1-to-6-month time shift scenarios.



(a)

(b)

(c)

(d)

(e)

(f)

Figure 7. Bar chart of sorted PPS coefficients of independent variables against WTI oil price multi-frequency time scenarios; a) 1-month, b) 2-month, c) 3-month, d) 4-month, e) 5-month, f) 6-month

Training a model on one dataset and testing it on the same data is a methodological mistake that causes overfitting. The model will learn from the provided sample and adopt to the training data, but it may fail if a different dataset is feed to the model. By partitioning the data into train/test splits, without letting the model to learn about the test split, it is possible to avoid overfitting. Also, the performance of the model can be independency assessed by developing quality metrics based on prediction of the test data (Pedregosa et al. 2011). The train/test split is also critical when optimizing the model estimators by evaluating different settings ("hyper-parameters") via cross-validation (CV). This approach is also called k-fold CV, where the training set is split into k smaller sets. Then the model is trained using $k - 1$ folds as training data and the resulting model is validated on the remaining part of the data to optimize the model. Finally, the final evaluation is made based on the performance of the unseen test data (Pedregosa et al. 2011; Refaeilzadeh, Tang, and Liu 2009). The train/test split and hyper-parameterization is discussed in training sub-section.

Normalization or scaling of datasets is a commonly required for modelling. As previously presented, the data has various units; however, the variables should be scaled into a common unit. This is usually achieved by removing the mean value of each feature, and dividing the by their standard deviation. It is recommended to find scaling parameters from the train then apply to the test dataset to practically keep them unseen. Figure 8 shows boxplot of scaled train data.

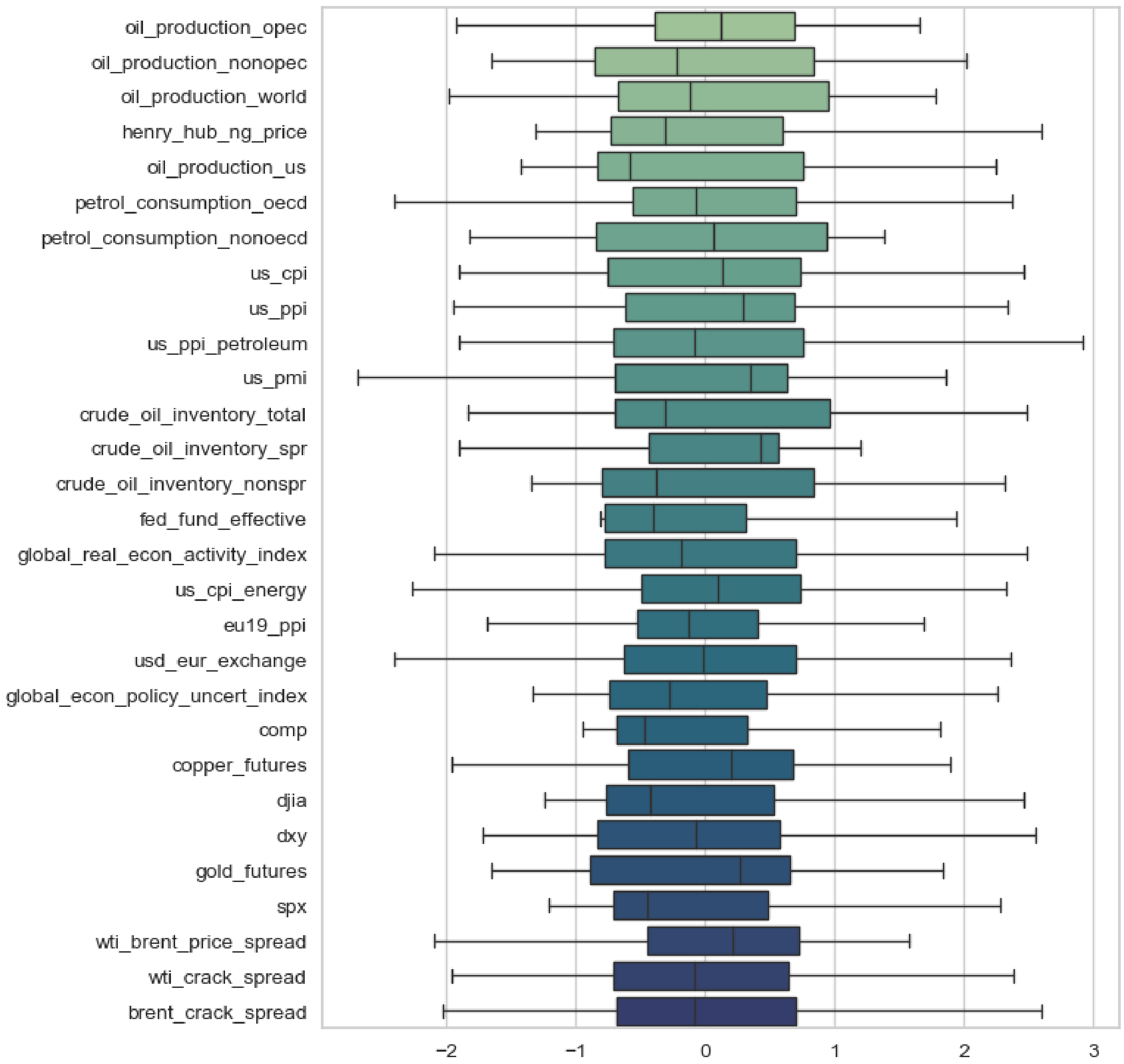


Figure 8. Boxplots of scaled train datasets

- *Training*

We use Pycaret, an open-source low code Python library that automates machine learning (AutoML) models to construct and deploy the models (Moez 2022). The library manages twenty-five different algorithms for regression, such as Extra Trees Regressor (ET), Gradient Boosting Regressor (GBR), Extreme Gradient Boosting Regressor (XGB), Random Forest Regressor (RF), Linear Regression (LR), AdaBoost Regressor (ADA), and eighteen other algorithms for classification.

We compare the performance of twenty-five AutoML models based on coefficient of determination (R^2), Mean Absolute Error (MAE), Mean Squared Error (MSE), and Root Mean Squared Error (RMSE). Then, we select the top five models for each time-series and build ensemble models. Ensemble methods benefit different training algorithms for increasing the training accuracy for reaching a higher testing accuracy to substantially improve the accuracy of the integerated model (Ardabili, Mosavi, and Várkonyi-Kóczy 2020).

Pycaret offer settings for train/test splits, missing value imputation methods, scaling, normalziation, etc. Table 4 shows the Pycaret regression session settings, where there data is configured to train/test split of 70/30, and transformation and normalization to be applied.

Table 4. Pycaret regression session settings

Description	Value
Session id	786
Target	wti_price
Target type	Regression
Data shape	(270, 30)
Train data shape	(188, 30)
Test data shape	(82, 30)
Numeric features	29
Preprocess	True
Imputation type	simple
Numeric imputation	mean
Categorical imputation	constant
Low variance threshold	0
Transformation	True
Transformation method	yeo-johnson
Normalize	True
Normalize method	zscore
Fold Generator	KFold
Fold Number	10

After setting the session we compared the ML models and their performance on the data with 5-fold of cross-validation generator.

Table 5. Performance of ML models on the predicting WTI price for 1-month

	Model	MAE	MSE	RMSE	R^2	RMSLE	MAPE
et	Extra Trees Regressor	4.51	40.47	6.32	0.95	0.12	0.08
gbr	Gradient Boosting Regressor	5.10	48.24	6.85	0.94	0.12	0.09
lightgbm	Light Gradient Boosting Machine	5.22	53.37	7.25	0.93	0.12	0.09
rf	Random Forest Regressor	5.30	55.06	7.38	0.93	0.13	0.10
lr	Linear Regression	5.90	56.68	7.49	0.93	0.16	0.12
ada	AdaBoost Regressor	5.80	59.57	7.63	0.92	0.13	0.11
huber	Huber Regressor	5.81	58.89	7.64	0.92	0.16	0.12
ridge	Ridge Regression	5.94	59.68	7.69	0.92	0.16	0.12
br	Bayesian Ridge	6.02	62.60	7.88	0.92	0.16	0.12
lasso	Lasso Regression	6.26	68.05	8.22	0.91	0.16	0.12
knn	K Neighbors Regressor	5.89	73.74	8.42	0.90	0.14	0.11
omp	Orthogonal Matching Pursuit	6.31	75.90	8.70	0.90	0.17	0.12
en	Elastic Net	6.86	79.07	8.86	0.89	0.17	0.13
dt	Decision Tree Regressor	6.64	84.37	9.07	0.88	0.15	0.12
par	Passive Aggressive Regressor	8.66	132.63	10.90	0.82	0.31	0.19
llar	Lasso Least Angle Regression	12.88	249.70	15.76	0.67	0.28	0.26
dummy	Dummy Regressor	23.21	756.93	27.45	-0.01	0.48	0.50

```
# Store the top 5 models in top5.
# We plan to use this for Stacking and Blending purposes.
```

Hyper-parameterization

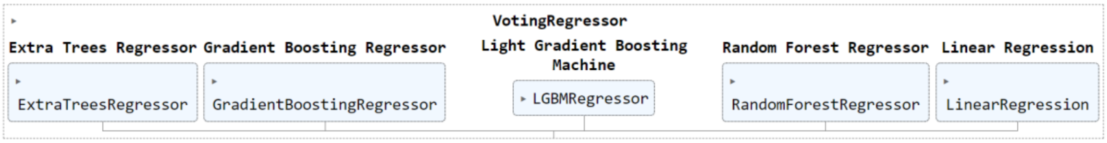
Tune the top 5 models dynamically with a higher iteration rate (n_iter).

This is to find more optimal hyper parameters over a larger search space.

```
tuned_top5[i] = tune_model(top, n_iter=120, optimize='RMSE', fold=5)
Fitting 5 folds for each of 120 candidates, totalling 600 fits
Original model was better than the tuned model, hence it will be returned.
NOTE: The display metrics are for the tuned model (not the original one).
```

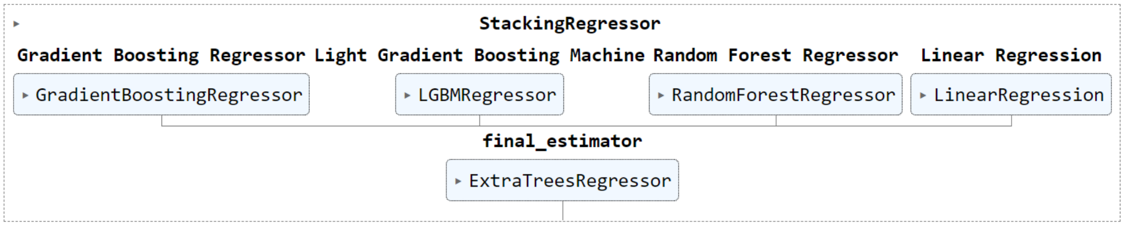
```
### Combine top 5 models
#### Build blend models
# Building a blending model from our top 5 models and evaluate the results.
# Blending models is an ensemble method of combining different machine learning algorithms.
# Use a majority vote to build consensus of final prediction values.

blender_specific = blend_models(estimator_list=tuned_top5[0:], fold=5, optimize='RMSE',
choose_better=False)
```

Fold	MAE	MSE	RMSE	R2	RMSLE	MAPE
0	4.7849	36.0137	6.0011	0.9562	0.0872	0.0736
1	4.4661	47.8932	6.9205	0.9324	0.1016	0.0757
2	4.3926	28.0386	5.2951	0.9577	0.0908	0.0752
3	5.4943	53.3835	7.3064	0.9415	0.1802	0.1332
4	3.8538	29.3091	5.4138	0.9554	0.1047	0.0748
Mean	4.5984	38.9276	6.1874	0.9486	0.1129	0.0865
Std	0.5388	10.0889	0.8024	0.01	0.0343	0.0234

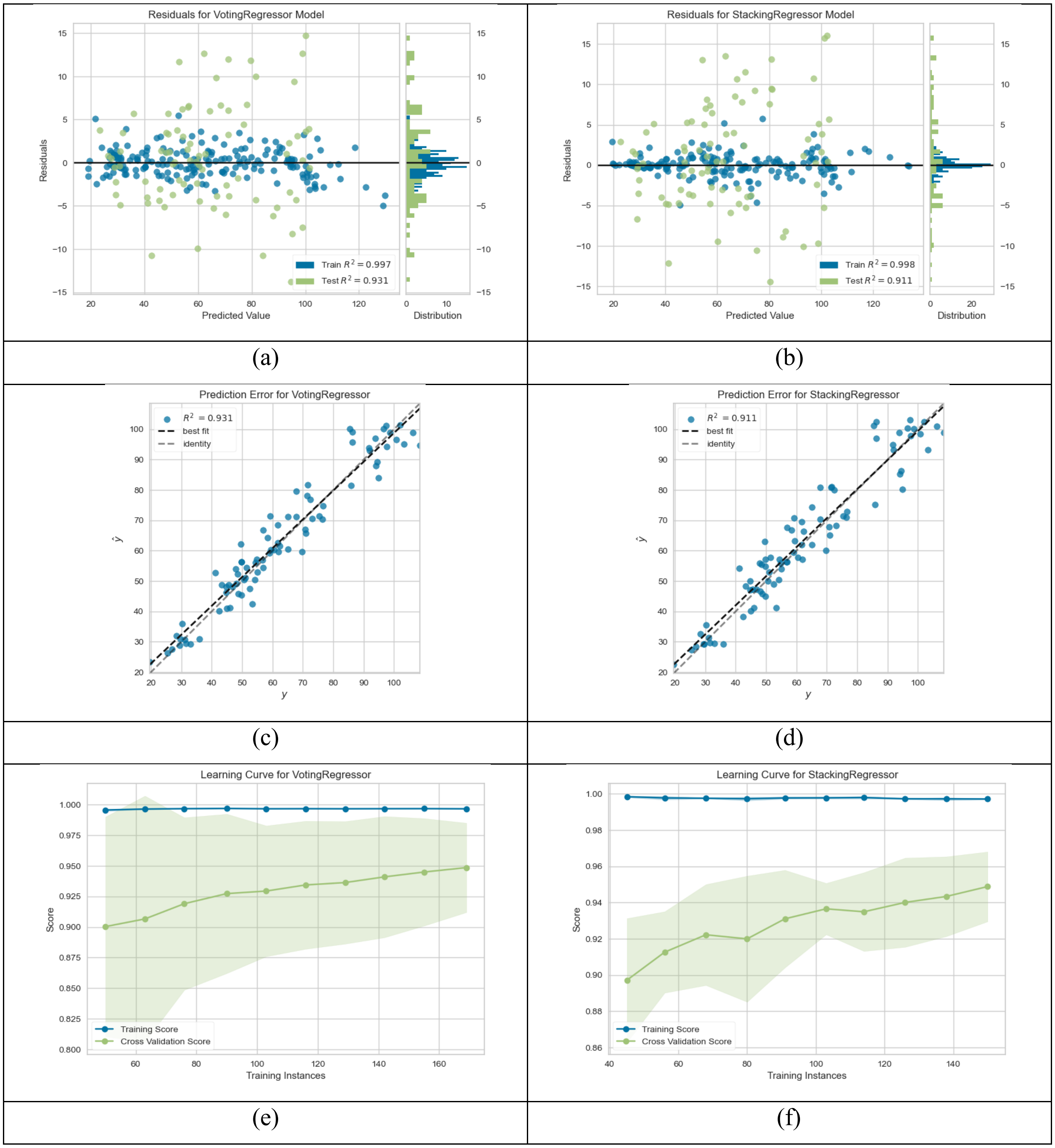


```
#### Build stack models
# Build a stacking model from our top 5 models and evaluate the results.
# Stacking models is an ensemble method of using meta learning.
# A meta model is created using multiple base estimators to generate the final prediction.

stacker_specific = stack_models(estimator_list=tuned_top5[1:],
                    meta_model=tuned_top5[0],
                    fold=5,
                    optimize='RMSE',
                    choose_better=False)
```

Fold	MAE	MSE	RMSE	R2	RMSLE	MAPE
0	3.9086	23.294	4.8264	0.9717	0.0643	0.0563
1	5.3399	61.0785	7.8153	0.9138	0.1114	0.0851
2	4.1576	26.6576	5.1631	0.9597	0.0903	0.0729
3	5.7948	61.4996	7.8422	0.9326	0.1865	0.138
4	3.7863	26.4179	5.1398	0.9598	0.1006	0.0716
Mean	4.5974	39.7895	6.1574	0.9475	0.1106	0.0848
Std	0.8137	17.5949	1.3699	0.0212	0.041	0.0281



Evaluate the performances



(a)

(b)

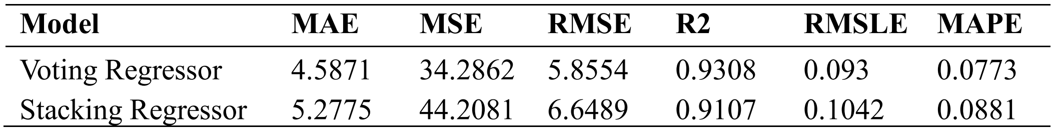
(c)

(d)

(e)

(f)

```
# Execute the predict_model() function to use the blender model to generate the predicted values.
# Execute the predict_model() function to use the stack model to generate the predicted values.
```



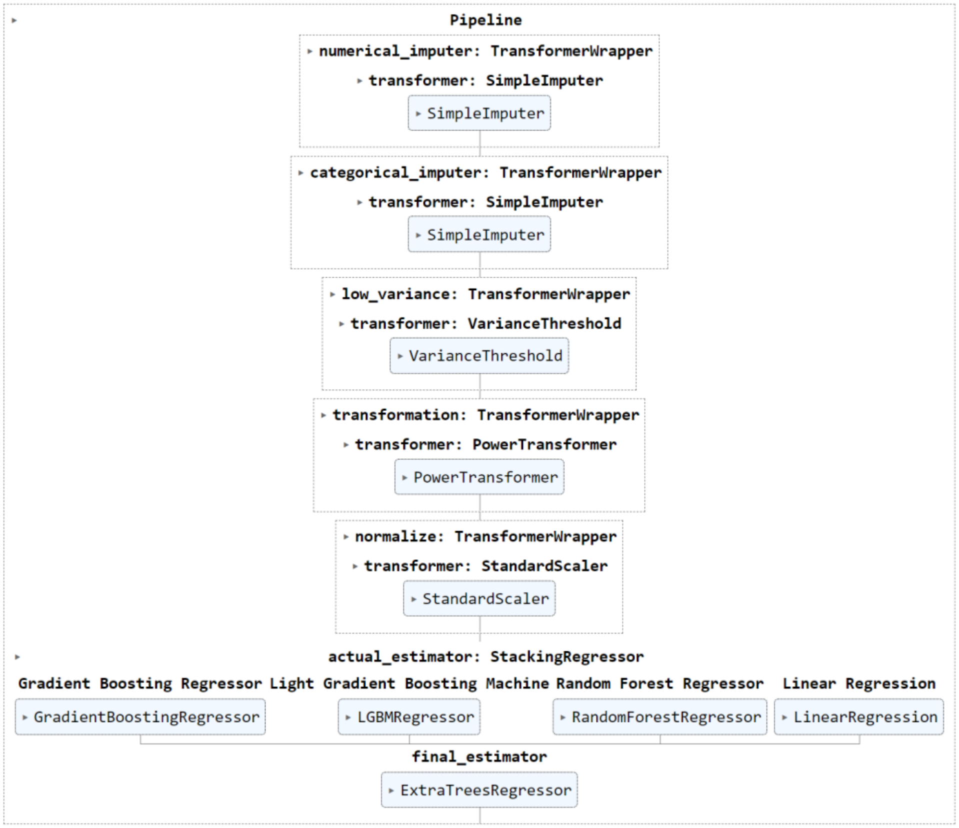
Model	MAE	MSE	RMSE	R2	RMSLE	MAPE
Voting Regressor	4.5871	34.2862	5.8554	0.9308	0.093	0.0773
Stacking Regressor	5.2775	44.2081	6.6489	0.9107	0.1042	0.0881

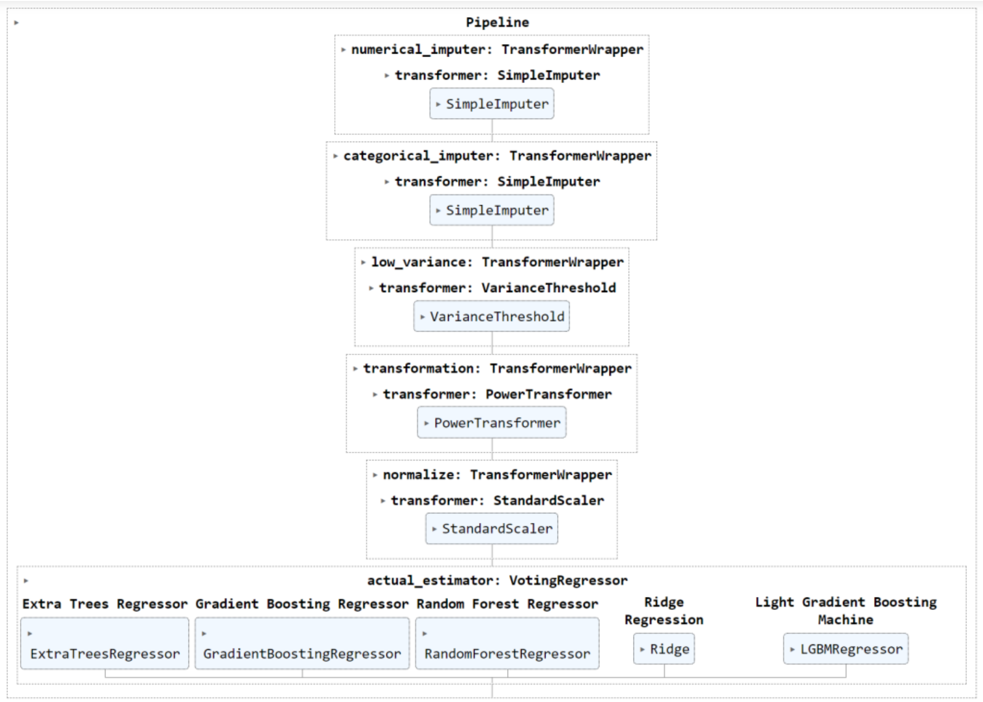
Save model

2.3. Modelling

Finally, the multi-frequnecy prediction time-series are weight-averaged based on the perfromance of the ML model into a single integrated predication series that represent the final oil price predictions.

```
## Analyze the Performance of Final Models on Entire Dataset
```


Pipeline
numerical_imputer: TransformerWrapper
transformer: SimpleImputer
SimpleImputer
categorical_imputer: TransformerWrapper
transformer: SimpleImputer
SimpleImputer
low_variance: TransformerWrapper
transformer: VarianceThreshold
VarianceThreshold
transformation: TransformerWrapper
transformer: PowerTransformer
PowerTransformer
normalize: TransformerWrapper
transformer: StandardScaler
StandardScaler
actual_estimator: StackingRegressor
Gradient Boosting Regressor
Light Gradient Boosting Machine
Random Forest Regressor
Linear Regression
GradientBoostingRegressor
LGBMRegressor
RandomForestRegressor
LinearRegression
final_estimator
ExtraTreesRegressor


Pipeline
numerical_imputer: TransformerWrapper
transformer: SimpleImputer
SimpleImputer
categorical_imputer: TransformerWrapper
transformer: SimpleImputer
SimpleImputer
low_variance: TransformerWrapper
transformer: VarianceThreshold
VarianceThreshold
transformation: TransformerWrapper
transformer: PowerTransformer
PowerTransformer
normalize: TransformerWrapper
transformer: StandardScaler
StandardScaler
actual_estimator: VotingRegressor
Extra Trees Regressor
Gradient Boosting Regressor
Random Forest Regressor
Ridge Regression
Light Gradient Boosting Machine
ExtraTreesRegressor
GradientBoostingRegressor
RandomForestRegressor
Ridge
LGBMRegressor

```
regression_model_1m
```

	Model	MAE	MSE	RMSE	R2	RMSLE	MAPE
0	Stacking Regressor	0.9828	1.9863	1.4094	0.9971	0.0246	0.0164

```
regression_model_2m
```

	Model	MAE	MSE	RMSE	R2	RMSLE	MAPE
0	Voting Regressor	1.4974	4.0136	2.0034	0.9941	0.0385	0.0273

```
regression_model_3m
```

	Model	MAE	MSE	RMSE	R2	RMSLE	MAPE
0	Stacking Regressor	1.3449	4.1664	2.0412	0.9938	0.0344	0.0224

```
regression_model_4m
```

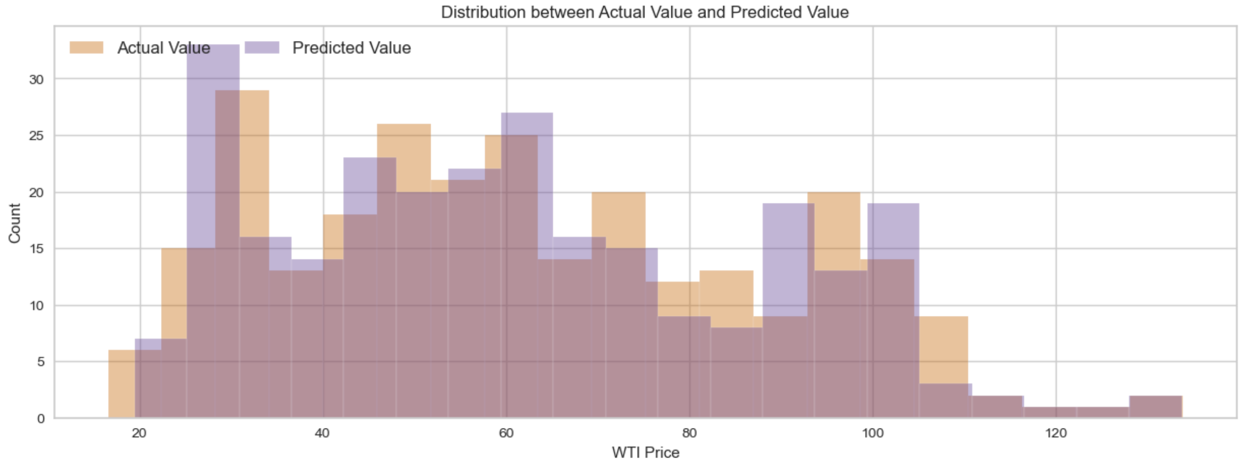
	Model	MAE	MSE	RMSE	R2	RMSLE	MAPE
0	Voting Regressor	1.8882	6.3462	2.5192	0.9905	0.0503	0.0346

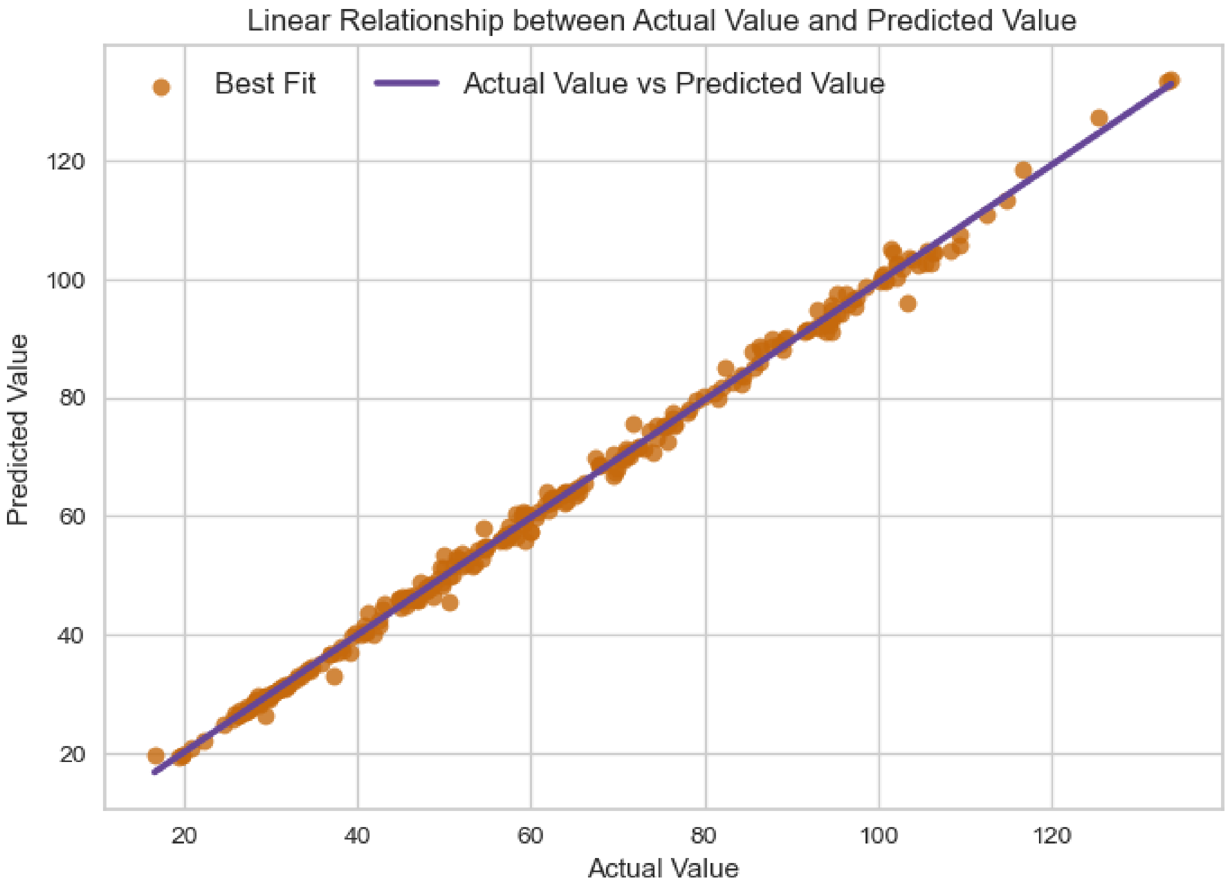
```
regression_model_5m
```

	Model	MAE	MSE	RMSE	R2	RMSLE	MAPE
0	Stacking Regressor	1.2967	5.0315	2.2431	0.9925	0.0553	0.025

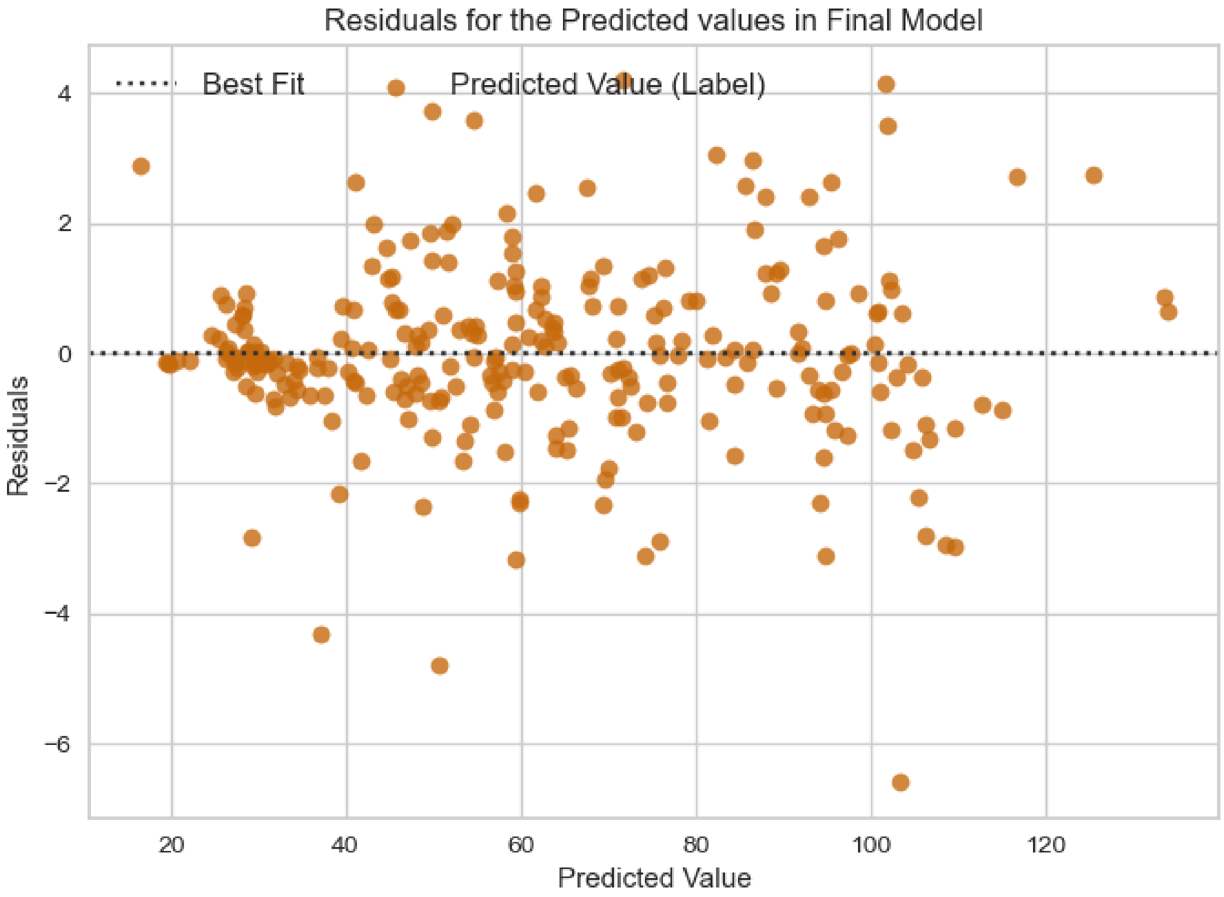
```
regression_model_6m
```

	Model	MAE	MSE	RMSE	R2	RMSLE	MAPE
0	Stacking Regressor	1.2893	5.3536	2.3138	0.992	0.0585	0.0253

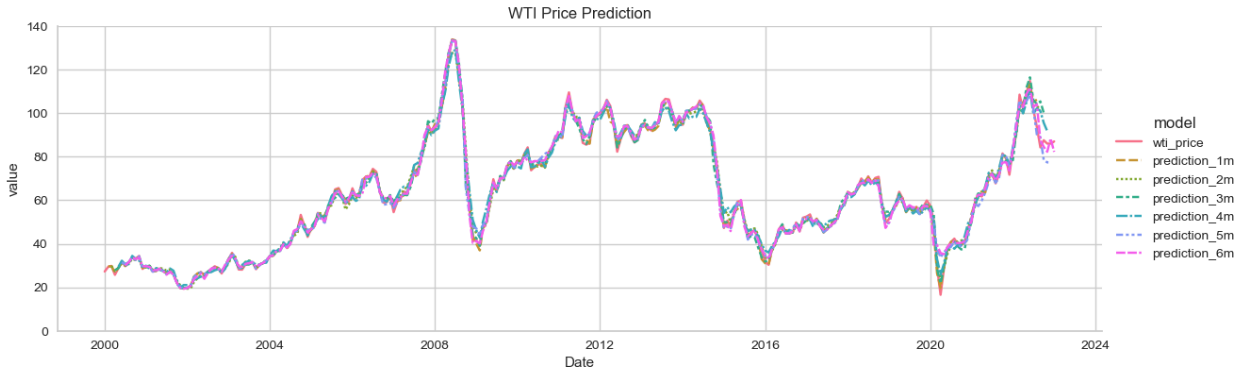


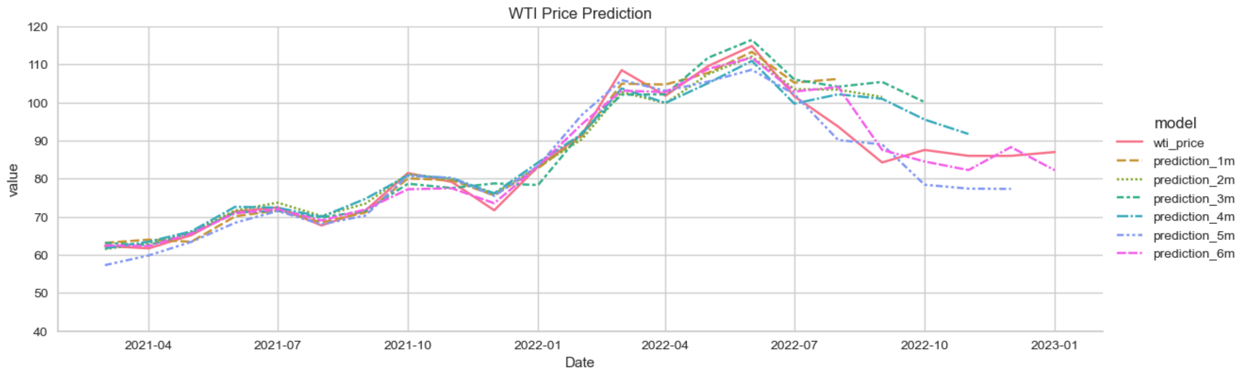

Linear Relationship between Actual Value and Predicted Value
Best Fit
Actual Value vs Predicted Value
Predicted Value
20
40
60
80
100
120
Actual Value
20
40
60
80
100
120

263



```
### Extend Date index for predicted months
### Cast predictions to the dataframe
## Plot predictions
```

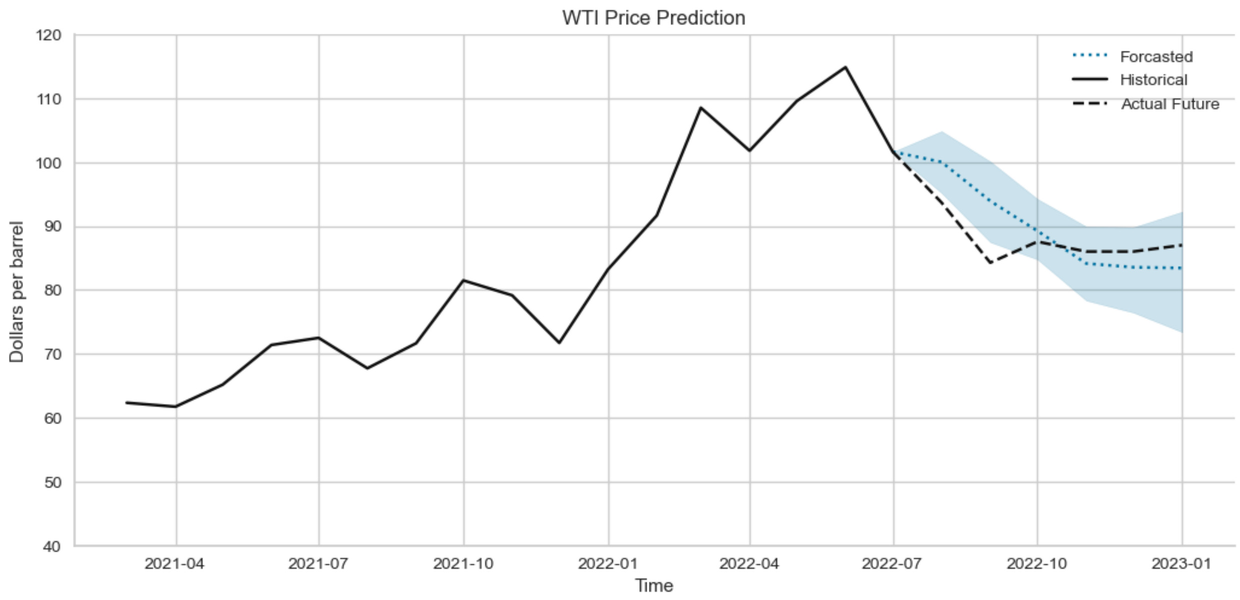




```
## Calculate Expected Average
#Make a copy to store average predictions
df_average = dates_extend_df.copy()


### Calculate weights
#Calcluate weight as 1/RMSE
weights = power(rmse_all, -1)

#Calclate weighted mean of predictions
df_average['avg_prediction'] = (df_average.iloc[:,3:]*weights_df).sum(axis=1)


### Calcalte average RMSE and upper and lower bounds
```

2.5758 is used because the 99% confidence interval has only 1% on each side. The probability for a z score below −2.5758 is 1%, and similarly for a z score above +1.96; added together this is 1%.

```
#Calcluate average RMSE of multiple models
count_rmse = dates_extend_df.notnull().iloc[:,3:].sum(axis=1)
df_average['avg_rmse'] = np.sqrt(sum(power(rmse_all,2))/count_rmse)
```



3. Results and Discussion

4. Recommendations and Future Work

- Fused TIR and RGB 3D models generated from UAS imagery offer great potential for mapping heat loss, supplementing non-destructive testing of structures, aiding in the inspection of electrical parts, and more.
- This study tested a simplified approach for generating 3D TIR point clouds from coacquired TIR and RGB images for remote sensing applications. The constructed TIR point clouds are georeferenced to the same coordinate system as the RGB

clouds. The resultant point cloud preserves the spatial density and resolution of the RGB point cloud while adding TIR attributes.

- The integrated visualization approach tested in this study enables 3D point cloud and 2D raster representation of RGB and TIR data in one model, enhancing the visual interpretation and analysis of the remotely-sensed data.
- The approach does not require additional depth sensors, such as lidar, or GNSS-aided INS for registration purposes.
- In general, the approach is appropriate for cases when….. For evaluation, and as examples of implementation…. While the SfM processing of RGB images was able to generate reliable….
- In future work, the proposed integration and visualization can be integrated into standard …... Radiometric calibration was considered beyond the scope of the present study; however, in-situ radiometric calibration of the thermal camera might improve the spectral content of the data. As an alternative …...
- TIR-RGB image feature matching and auto-registration can handle non-synchronized dual-head camera captures; however, extraction of identical features and co-registration based on the extracted pair is challenging for images of different spectral bands at the scene without well-designed calibration patterns.
- It is recommended that follow-on studies be conducted to address these topics…..

Acknowledgments

We thank … for their valuable comments and suggestions on improving the quality of this paper.

We are thankful to …. for their help with collecting the data,

We would like to acknowledge … for supplying the logistics for

We also appreciate …. for providing surveying equipment and/or software.

We would like to thank …. anonymous reviewers for their constructive suggestions and comments.

References

Ardabili, Sina, Amir Mosavi, and Annamária R. Várkonyi-Kóczy. 2020. “Advances in Machine Learning Modeling Reviewing Hybrid and Ensemble Methods.” In , 215–227. doi:10.1007/978-3-030-36841-8_21.

Basiri, Mohammad Hossein; Javadnejad, Farshid; Saeedi, Azita. 2015. “Forecasting Crude Oil Price with an Artificial Neural Network Model Based on a Regular Pattern for Selecting of the Training and Testing Sets Using Dynamic Command-Line Functions.” In *24th International Mining Congress and Exhibition of Turkey-IMCET’15*.

Bilogur, Aleksey. 2018. “Missingno: A Missing Data Visualization Suite.” *The Journal of Open Source Software* 3 (22). The Open Journal: 547. doi:10.21105/JOSS.00547.

Deng, Chao, Liang Ma, and Taishan Zeng. 2021. “Crude Oil Price Forecast Based on Deep Transfer Learning: Shanghai Crude Oil as an Example.” *Sustainability* 13 (24): 13770. doi:10.3390/su132413770.

EIA. 2022a. “Opendata - U.S. Energy Information Administration (EIA).” https://www.eia.gov/opendata/.

EIA. 2022b. “International Energy Statistics, Total Oil Production.” *U.S. Energy Information Administration*.

FRED. 2022. “St. Louis Fed Web Services: FRED® API.” https://fred.stlouisfed.org/docs/api/fred/.

Hamilton, James. 2008. *Understanding Crude Oil Prices*. Cambridge, MA. doi:10.3386/w14492.

Hamilton, James. 2009. *Causes and Consequences of the Oil Shock of 2007-08*. Cambridge, MA. doi:10.3386/w15002.

Hou, Aijun, and Sandy Suardi. 2012. “A Nonparametric GARCH Model of Crude Oil Price Return Volatility.” *Energy Economics* 34 (2): 618–626. doi:10.1016/j.eneco.2011.08.004.

Investing. 2022. “Investing.Com - Stock Market Quotes & Financial News.” https://www.investing.com/.

Jammazi, Rania, and Chaker Aloui. 2012. “Crude Oil Price Forecasting: Experimental Evidence from Wavelet Decomposition and Neural Network Modeling.” *Energy Economics* 34 (3): 828–841. doi:10.1016/j.eneco.2011.07.018.

Javadnejad, Farshid. 2012. “Presenting a Model for Prediction of Crude Oil Price Based on Artificial Intelligent Hybrid Methods and Time-Series.” Tarbiat Modares University.

Kilian, Lutz, and Daniel P. Murphy. 2014. “THE ROLE OF INVENTORIES AND SPECULATIVE TRADING IN THE GLOBAL MARKET FOR CRUDE OIL.” *Journal of Applied Econometrics* 29 (3): 454–478. doi:10.1002/jae.2322.

Kluyver, Thomas, Benjamin Ragan-Kelley, Fernando Pérez, Brian E Granger, Matthias Bussonnier, Jonathan Frederic, Kyle Kelley, et al. 2016. *Jupyter Notebooks: A Publishing Format for Reproducible Computational Workflows*. Vol. 2016. https://jupyter.org/.

Lanza, Alessandro, Matteo Manera, and Massimo Giovannini. 2005. “Modeling and Forecasting Cointegrated Relationships among Heavy Oil and Product Prices.” *Energy Economics* 27 (6): 831–848. doi:10.1016/j.eneco.2005.07.001.

Lu, Quanying, Shaolong Sun, Hongbo Duan, and Shouyang Wang. 2021. “Analysis and Forecasting of Crude Oil Price Based on the Variable Selection-LSTM Integrated Model.” *Energy Informatics* 4 (S2): 47. doi:10.1186/s42162-021-00166-4.

McKinney, Wes. 2011. “Pandas: A Foundational Python Library for Data Analysis and Statistics.” *Python for High Performance and Scientific Computing* 14 (9). Seattle: 1–9. https://pandas.pydata.org/.

Moez, Ali. 2022. “PyCaret: An Open Source, Low-Code Machine Learning Library in Python.”

Murat, Atilim, and Ekin Tokat. 2009. “Forecasting Oil Price Movements with Crack Spread Futures.” *Energy Economics* 31 (1): 85–90. doi:10.1016/j.eneco.2008.07.008.

Pedregosa, F, G Varoquaux, A Gramfort, B Michel V. and Thirion, O Grisel, M Blondel, R Prettenhofer P. and Weiss, et al. 2011. “Scikit-Learn: Machine Learning in Python.” *Journal of Machine Learning Research* 12: 2825–2830.

Python Software Foundation. 2022. “Python.” Python Software Foundation (PSF). https://www.python.org/.

Refaeilzadeh, Payam, Lei Tang, and Huan Liu. 2009. “Cross-Validation.” *Encyclopedia of Database Systems* 5. Springer: 532–538.

Wang, Yudong, Chongfeng Wu, and Li Yang. 2015. “Forecasting the Real Prices of Crude Oil: A Dynamic Model Averaging Approach.” *SSRN Electronic Journal*. doi:10.2139/ssrn.2590195.

WSJ. 2022. “The Wall Street Journal - Breaking News, Business, Financial & Economic News, World News and Video.” https://www.wsj.com/.

Yu, Lean, Yang Zhao, and Ling Tang. 2017. “Ensemble Forecasting for Complex Time Series Using Sparse Representation and Neural Networks.” *Journal of Forecasting* 36 (2): 122–138. doi:10.1002/for.2418.

Zhao, Yang, Jianping Li, and Lean Yu. 2017. “A Deep Learning Ensemble Approach for Crude Oil Price Forecasting.” *Energy Economics* 66 (August): 9–16. doi:10.1016/j.eneco.2017.05.023.