ORIE 5741 - Project Report

Natural Gas Price Forecasting

Anouar Brahimi (ab2697), Miao Zhang (mz528), Yahan Xu (yx584)

Cornell University

Contents

Abstract

Natural gas accounts for one-fourth of the global energy demand and roughly one-third of the US energy demand. Accurately forecasting natural gas prices is extremely valuable given the volatile nature of the US natural gas market and the sheer amount of factors that lead to its price formation. This project aims to predict the spot price for natural gas leveraging time series analysis and machine learning techniques. More specifically, we used linear models with regularization, support vector regression, and decision trees for price prediction purposes. We also discussed potential ways that we could perfect our prediction model.

1 Data Sourcing and Pre-processing

1.1 Data Sourcing

The pricing of natural gas is subject to various factors such as supply and demand dynamics, geopolitical events, and economic conditions, among others. Storage and other energy markets play a crucial role in natural gas pricing. Indeed, the availability of storage infrastructure to store excess supply during low-demand periods and release it during high-demand periods helps to balance the natural gas market. Additionally, natural gas can be used as a substitute for other energy sources such as coal and oil making the interaction between natural gas and other energy markets impact the pricing of natural gas and vice versa.

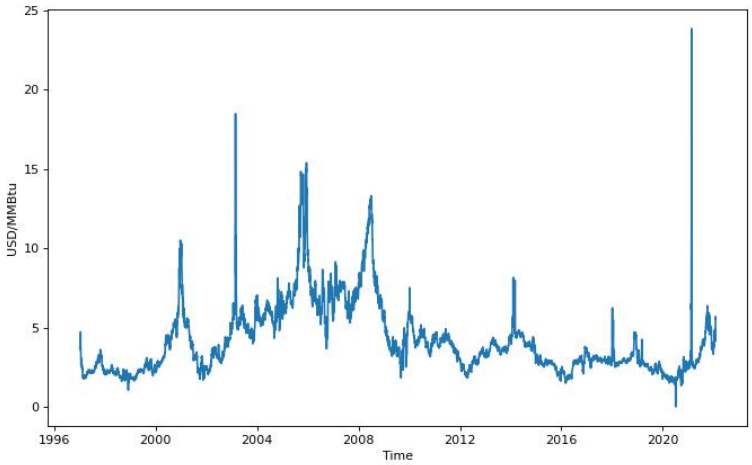

Figure 1: Henry Hub Natural Gas Spot Price (1997-2022)

Regarding the economic indicators, we focused mainly on the US economy with US CPI (Consumer Price Index) as a proxy for inflation and thus the purchasing power of consumers. The Dow Jones Index was used as a market proxy to measure the health of the American economy. Finally, we used the 3-month and 5-year US treasury rates to capture the idk bro??? as a benchmark for consumer interest rate???? idk either?? one-month risk free return??
In this paper, the objective variable is the next-month Henry Hub spot price, a widely used benchmark for natural gas pricing in the US. It is used as a reference point for natural gas contracts and futures traded on other commodity exchanges.
All the data above is sourced from Nasdaq and Yahoo Finance. Below is a table summarizing the features considered in this study:

1.2 Data Pre-processing

Given the abundant data, we need to do some cleanups before we can fit the data to our models.

1.2.1 Missing Data

The very first step is to see whether we have missing data. After importing the data using Nasdaq and Yahoo APIs, we did a run-through to see how much data was missing by counting the number of cells that contain NaN values. Our result was quite good, which shows that for the columns that have data missing, there were at most 0.3333% data missing. Therefore, we simply filled those empty cells up with the last non-null value in the same column. Forward filling in time series data sets is particularly useful here since we only have a small portion of missing values.

#	Feature	Mean	St Dev
1	WTI Crude Oil Spot	55.868	27.777
2	US Gas Supply	68.489	15.781
3	US Gas Consumption	68.470	15.157
4	US Gas Exports	6.832e+06	1.407e+04
5	US Gas Imports	2.874e+05	5.507e+04
6	US Gas Storage Volume	6.832e+06	8.025e+05
7	US Gas Marketed Production	67.328	15.995
8	US Gas Rig Count	652.66	434.80
9	US CPI	213.084	32.266
10	Dow Jones Index	14740.199	6905.968
11	3-month treasury rate	1.881	1.966
12	5-year treasury rate	2.927	1.745

Table 1: List of features with mean and standard deviation

1.2.2 Feature Selection

We plotted a heat map to represent the correlation matrix to get an idea of the correlation between each pair of features. Then we eliminated the ones that have high correlations to lower the complexity of our model without hurting its performance. One example of such a pair would be the heating oil spot price and WTI spot price. The latter one being the benchmark used for US Crude oil price (the two are considered substitutes for natural gas). After comparing the performance before and after removing certain features, we ended up with the below heat map that contains all the features we considered in our prediction model.

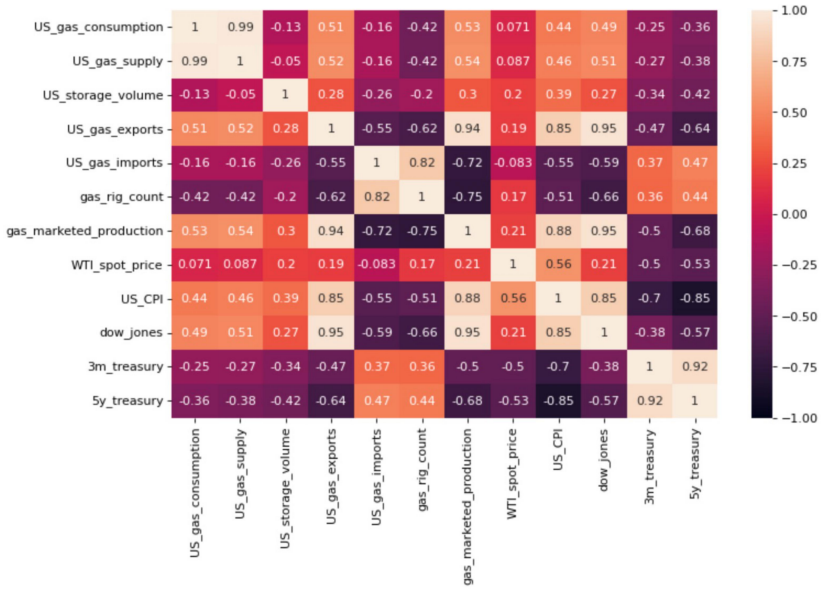

Figure 2: Correlation heat map of the features

1.2.3 Feature Scaling

Certain loss functions are more sensitive to the scaling of input features. For features with larger scales, we may see such features dominate the loss function, leading to an increase in the bias of our model parameters and predictions. Making sure that our features are on the same scale can ensure

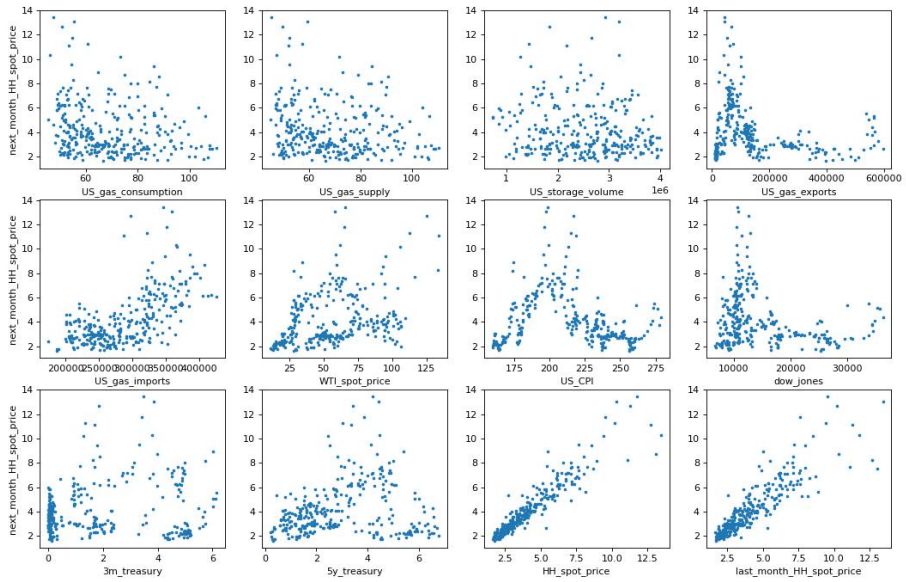

Figure 3: scatter plot of the relationship of features and spot price

that the model is able to learn the underlying patterns in the data regardless of the difference in scaling. This can improve the performance of the model by reducing over-fitting, making it easier to generalize. Feature scaling can also speed up the learning rate of the models during the gradient descent optimization thus our model could converge faster.

1.2.4 Auto-regression

Including auto-regressive features like lagged values of the spot price can capture temporal dependencies in time series data, which can show us if there exist repeating patterns or trends. Doing so helps to detect seasonality and changes in trends over time, therefore, boosting the model's accuracy.

2 Machine Learning model selection

In this section, we will give a brief introduction to the principles of several supervised regression machine learning models and explain the applicability of these models to our data.
To enhance the predictive power of our models, hyperparameter fine-tuning techniques are to be leveraged. However, it is worth mentioning that the traditional k-fold cross-validation is not applicable to the time-series parameter tuning. This is due to the risk of disruption of time-dependent patterns and capturing future information during model training. To overcome this problem, a time-series k-fold cross-validation would be utilized to find the optimal parameters. The illustration using sklearn's TimeSeriesSplit is in Appendix A. This approach allows us to maintain the time-dependent patterns of the data and prevent future information leakage.

2.1 Linear Models

Linear models assume a linear relationship between the features and the objective. Figure 3 illustrates that several features (e.g., spot price, lagged spot price) exhibit a linear relationship with the objective variable. Consequently, we will initially explore linear models.

2.1.1 Least Squares

The Ordinary Least Squares (OLS) method is a fundamental linear model. It aims to minimize the sum of squared differences between the features and the objective variable.

2.1.2 Ridge Regularization

Figure 2 highlights the presence of some degree of correlation (e.g. CPI & WTI) among the features, albeit not particularly strong. Also, OLS has the potential risk of over-fitting. By introducing an L2 penalization term to the regression coefficient, Ridge regression promotes regularization and helps mitigate these issues.

2.1.3 Lasso Regularization

In addition to Ridge regression, we also investigated Lasso regression. In contrast to Ridge regression, Lasso employs an L1 penalization technique, which leads to sparse solutions. This property makes Lasso more robust in addressing multi-collinearity issues by effectively selecting relevant features while discarding less important ones.

2.1.4 Huber Regularization

From Figure 1, it is evident that the time series contains several extreme prices. To account for these outliers, we explored Huber regularization, which utilizes a piecewise loss function that handles different residuals. Compared to Lasso and Ridge regression, Huber regularization is less sensitive to outliers, making it effective in some scenarios.

2.2 Support Vector Regression

As mentioned earlier, only a few features adhere to the assumptions of linear regression. Therefore, our study will now shift toward non-linear models. The support vector machine operates on the principle of finding an optimal hyperplane that maximizes the margin. By utilizing the kernel function, it is capable of generating both linear and non-linear models. The kernel function maps the features to a high-dimensional space using specific functions. In addition to the capability of handling non-linear models, one notable advantage of the support vector machine is its robustness to outliers. It achieves this by prioritizing the correct classification of support vectors located closest to the decision boundary while being less influenced by outliers situated far from the boundary. Additionally, the support vector machine includes a regularization parameter that balances the trade-off between variance and bias.

2.3 Decision Trees

As we saw in Figure 3, not all the features present a linear relation with the target price, that is why we used decision trees that are non-linear models to capture those dependencies.

2.3.1 Regression Tree

The regression Tree is the simplest tree model. It works by iteratively partitioning the features to find the segmentation that minimizes the loss function, to create a tree-like structure.

2.3.2 Bagged Tree (Random Forest)

An ensemble model aggregates many simple models (weak learners) together to create a more powerful model with lower variance. Bagging (Bootstrap AGgregation) generates weak learners by training on different bootstrap samples (samples with replacement from the dataset) of the training data. The regression is the average of the outputs. Here, we used the Random Forrest algorithm that consists of a bagged ensemble of trees, each restricted to a random subset of features.

2.3.3 Boosted Tree (XGBoost)

In order to help with bias and under-fitting, we can use boosting which sequentially adds new models to correct the errors. In this case, XGBoost uses gradient boosting involving a forecast of the residuals of previous models to form the final prediction.

3 Results

3.1 Linear Models

Figure 4 illustrates the predictions of the four linear models alongside the actual spot price. It is evident that only lasso regression demonstrates a close alignment with the actual price. Conversely, the remaining three linear models tend to overestimate the gas spot price. Additionally, Table 2 provides further insights. Lasso regression demonstrates a satisfactory result with a test RMSE of 0.58. In contrast, the other three models exhibit significantly higher test RMSE values, approximately twice as large as their respective train RMSE values. This observation suggests that, apart from lasso regression, the remaining three models suffer from the problem of over-fitting.

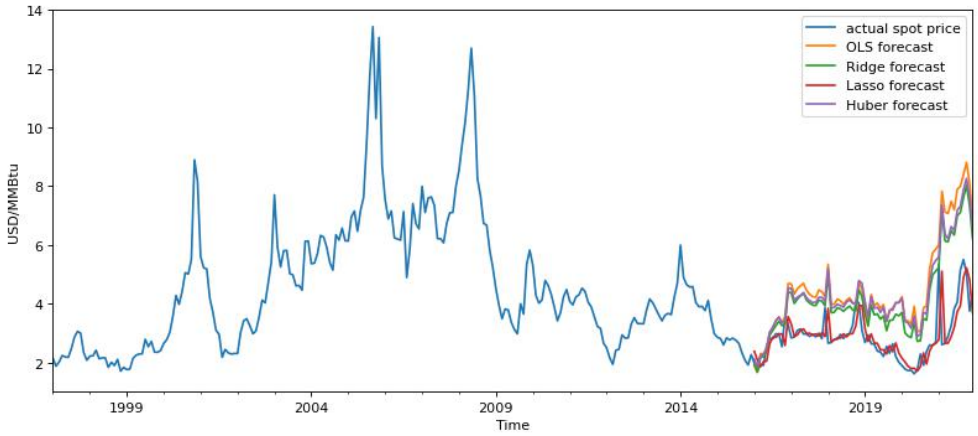

Figure 4: Forecast of Henry Hub spot price using linear models

3.2 Support Vector Regression

We proceeded to explore the SVM algorithm and its application to our dataset. From Figure 5, it is evident that the predictions align well with the actual data before 2018. However, the algorithm loses its predictive power thereafter, failing to accurately capture the fluctuations and peaks in the data. Furthermore, Table 5 reveals that the algorithm exhibits a notably small training error of 0.18, but the test root mean squared error (RMSE) is 0.78. Although the SVM is better than some of the linear models discussed in the previous section, it still encounters the problem of over-fitting.

3.3 Decision Trees

Figure 4 illustrates the predictions of the three tree models alongside the actual spot price. It can be observed that the predictions of the random forest and XGBoost align well with the actual spot price. The decision tree, on the other hand, has some prediction errors in 2017 and 2018, but overall it performs well during the testing period. Table 2 provides further insights. We can observe that all of the tree models have a test RMSE of approximately 0.65, indicating their good performance during the testing period. Moreover, the training RMSE of the decision trees and random forests suggests that they are not overfitting. However, the extremely small training RMSE of XGBoost

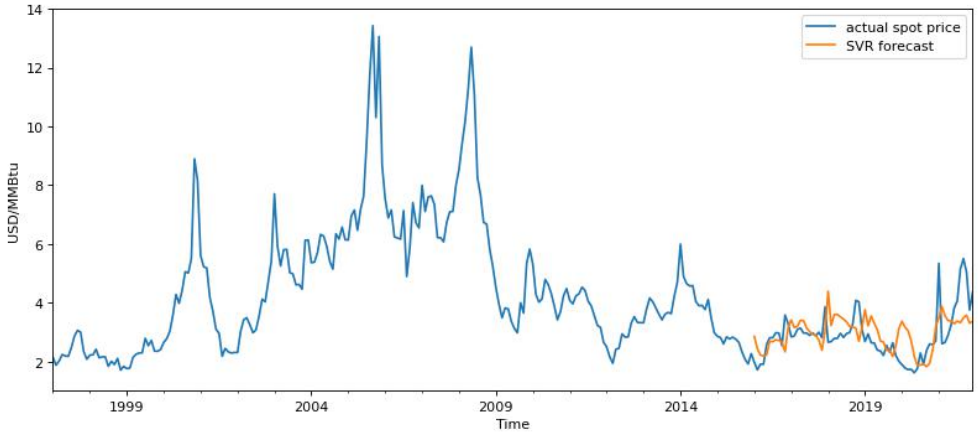

Figure 5: Forecast of Henry Hub spot price using Support Vector Regression

suggests potential overfitting. Although XGBoost performed well on the test set, the significant gap between the training and test performance raises concerns about its ability to maintain predictive validity in the future. In contrast, the random forest stands out as the most stable model among the three tree models, demonstrating similar performance (RMSE) in both the training and test sets. Therefore, we select it as the best tree model.

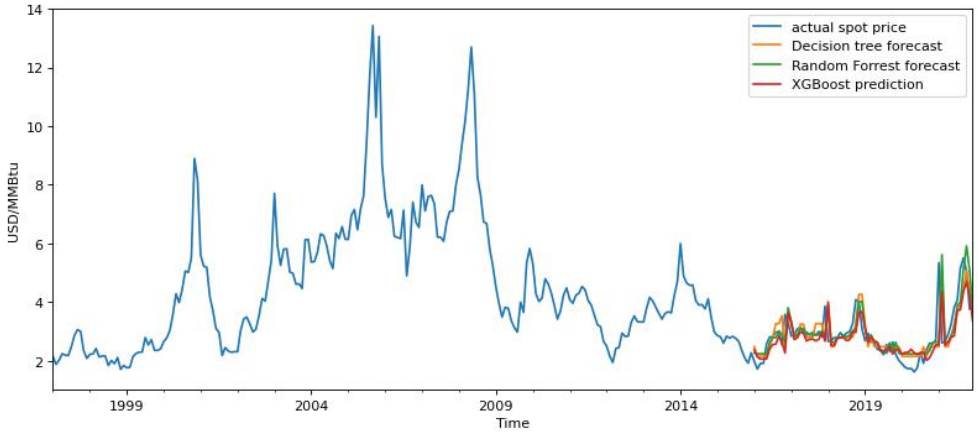

Figure 6: Forecast of Henry Hub spot price using regression trees

4 Potential ways of improvement

4.1 Time Series Analysis

As shown below when plotting the PACF (Partial Auto-Correlation Function) of the price time series, there is an auto-correlation component for some lags. Another approach would then be to apply models, traditionally used in econometrics, such as ARIMA+GARCH. The GARCH component is usually used for financial time series as it captures volatility persistence. However, these techniques require the stationary of the time series, so pre-processing may be needed before getting any valuable results.

#	Model	RMSE train	RMSE test	overfitting?
1	OLS	0.7267	2.0015	yes
2	Ridge	0.7293	1.4784	yes
3	Lasso	0.7832	0.5890	no
4	Huber	0.7415	1,7602	yes
5	SVR	0.1832	0.7809	yes
6	Regression Tree	0.3000	0.6588	no
7	Random Forest	0.5951	0.6459	no
8	XGBoost	0.0009	0.6345	yes

Table 2: RMSE scores of the models tested

4.2 Natural Language Processing

So far, we based the forecast of the price only on market data, however, another aspect that should not be undermined is how the short-term volatility can be event-driven. Thus, using large language models to process text data such as tweets and news headlines could be of great value. An approach would be to define a sentiment score that can then be used to enhance the short-term variations of the forecast.

4.3 Neural Networks

Long Short-Term Memory (LSTM) can remember values over arbitrary time intervals. These models have the advantage over simple neural network architecture by its capacity to encounter current and previous sequences in producing forecasts. Moreover, LSTM modeling does not need the input dataset to be stationary, as required by the ARIMA-type models. Such makes it a potential candidate in time-series forecasts with the possibility of encountering historical shocks across units and layers through the memory cell at each time step.

5 Conclusion

In this paper, we investigated machine learning techniques for the problem of natural gas price forecasting. Specifically, we collected 11 years of gas spot prices and various features that could potentially impact the gas price based on intuition. The data was processed by handling missing values, selecting features with low correlation, normalizing the features, and incorporating lagged values as new features.

We then proceed to the core part of model selection and implementation. Upon examining our features, we discovered both linear and non-linear relationships with the objective. Consequently, we initially explored four linear models and found that lasso achieved the best performance, delivering a satisfactory test RMSE of 0.58. Subsequently, we sought to capture the non-linear relationships by employing non-linear models, including SVM and three tree models. The non-linear models generally outperformed the linear models (except for lasso), yielding testing RMSE values ranging from approximately 0.6 to 0.8. However, both SVM and XGBoost exhibited signs of overfitting, potentially hindering their generalization to future data. In contrast, Random Forest demonstrated consistent performance on both the training and testing sets, making it the preferred non-linear model. In conclusion, among the models utilized, lasso stood out as the best linear model, while Random Forest emerged as the best non-linear model with good generalization properties.

For further improvement, we suggest exploring three additional fields: time series models, Natural Language Processing (NLP), and Neural Networks, which have the potential to yield deeper insights and provide more effective forecasts than the existing models.

A K-fold cross-validation for time series

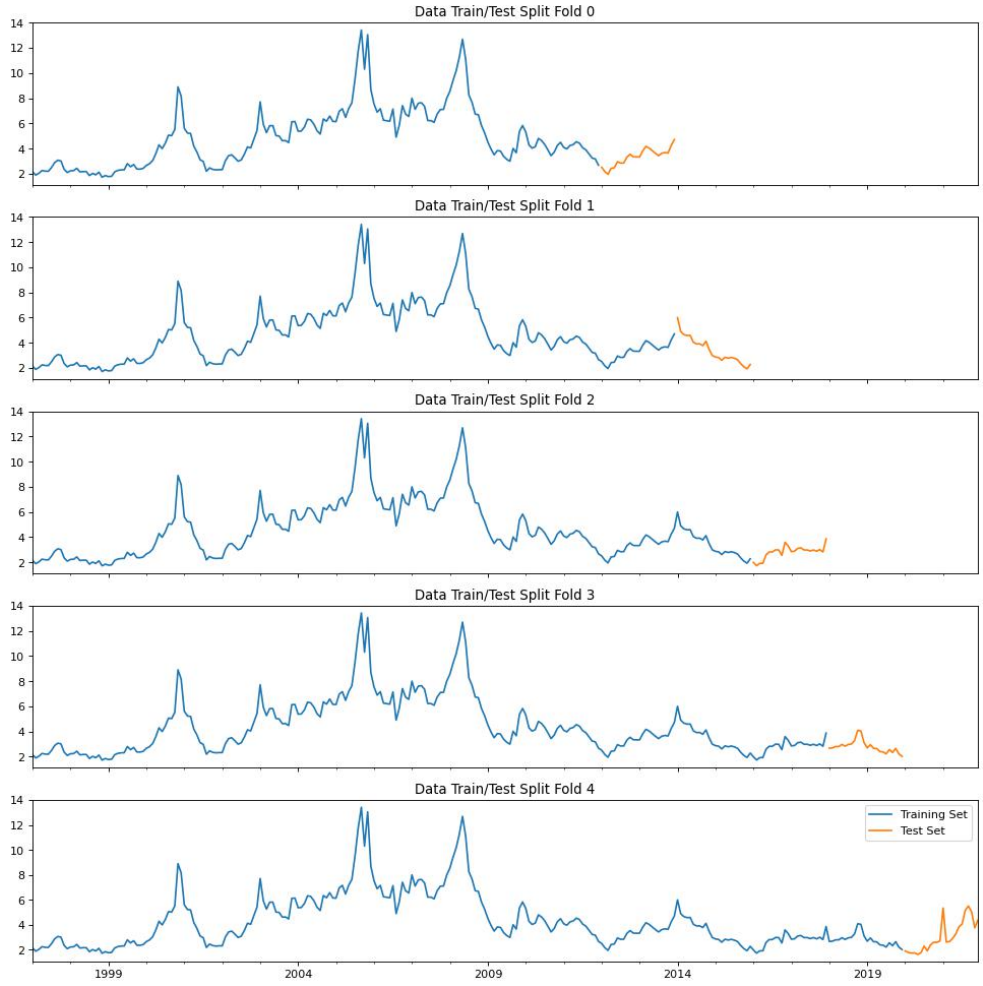

Figure 7: K-fold cross-validation illustration for the spot price time series (k=5 here)

B Scatter plots of predicted values vs. actual values

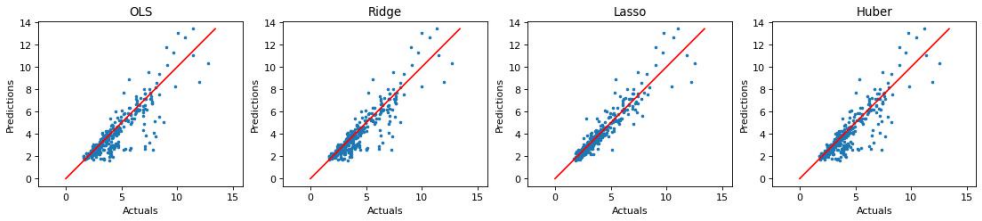

Figure 8: Linear models

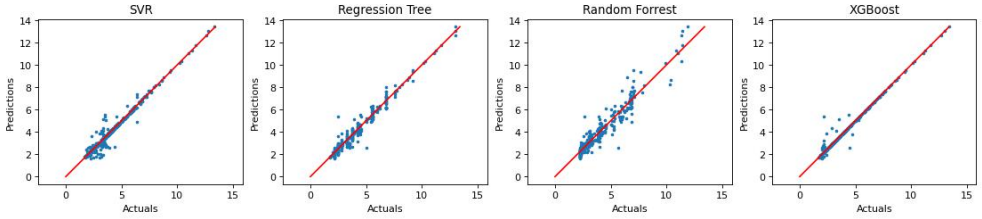

Figure 9: Non-linear models

References

[1] D. Mouchtaris, E. Sofianos, P. Gogas, T. Papadimitriou (2021) *Forecasting Natural Gas Spot Prices with Machine Learning*, Aristotle University of Thessaloniki

[2] M. Su, Z. Zhang, Y. Zhu, D. Zha (2019) *Data-Driven Natural Gas Spot Price Forecasting with Least Squares Regression Boosting Algorithm*

[3] E. Howell, (2023) *How To Correctly Perform Cross-Validation For Time Series*, https://towardsdatascience.com/how-to-correctly-perform-cross-validation-for-time-series-b083b869e42c