POLITECNICO
MILANO 1863

SCUOLA DI INGEGNERIA INDUSTRIALE
E DELL'INFORMAZIONE

Energy price prediction

Report - Numerical Analysis for Machine Learning project

Mirko Magistrado Dacara, 11006751

Advisor:
Prof. Edie Miglio

Academic year:
2023-2024

Abstract:
This report explores the application of various machine learning algorithms in forecasting natural gas spot prices. The accurate prediction of natural gas prices is crucial for market participants, including suppliers, distributors, consumers, investors, and regulatory agencies, as it helps in risk management, reducing the demand-supply gap, and optimizing resource utilization. The study employs a comprehensive dataset comprising 2423 daily observations from December 3, 2010, to September 18, 2020, incorporating 21 explanatory variables selected based on economic theory and relevant literature.
I evaluated the performance of several models, including linear regression, robust linear regression, support vector machines (SVM) with different kernels, Gaussian Process Regression (GPR), decision trees, bagged trees, boosted trees, and a naïve random walk model. The models were trained to forecast natural gas prices for one, three, five, and ten days ahead ($t+1$, $t+3$, $t+5$, and $t+10$).
The results indicate that linear regression and robust linear regression models consistently provided the best performance across different forecasting horizons, particularly for out-of-sample predictions. The random walk model also performed well for the $t+1$ horizon. Additionally, the bagged trees and boosted trees models demonstrated strong predictive capabilities, especially in capturing complex patterns within the data.
This study highlights the effectiveness of linear and ensemble-based machine learning models in short-term natural gas price forecasting and underscores their potential utility for various stakeholders in the natural gas market.

Key-words: energy price, natural gas, machine learning, prediction, linear regression

1. Introduction

Natural gas has emerged as a pivotal energy resource aimed at enhancing energy security and mitigating environmental pollution globally. It stands as the second most utilized energy commodity, following oil. The shift from coal to natural gas has intensified the importance of accurate gas spot price forecasting across various sectors. Accurate predictions of natural gas spot prices are crucial for market operations, power system planning, and regulatory decision-making, impacting both supply and demand dynamics.
Given the substantial economic benefits derived from precise forecasting, numerous methodologies have been explored, particularly in electric load forecasting, including artificial neural networks (ANN) and support vector machines (SVM). However, while energy market forecasting studies predominantly focus on crude oil prices, research on natural gas price forecasting remains relatively scarce.

Noteworthy among the limited studies is the work of Buchanan et al., which attempted to predict natural gas price movements in the U.S. market by analyzing trader positions. Nguyen and Nabney forecasted gas prices one day ahead using monthly forward products and futures, employing wavelet transform (WT) combined with adaptive machine learning and time series models. Their findings highlighted the adaptive MLP/GARCH models as superior for electricity demand and gas price forecasting.
Salehnia et al. utilized several nonlinear models, including local linear regression (LLR), dynamic local linear regression (DLLR), and ANN, to forecast Henry Hub spot prices, concluding that ANN models outperform LLR and DLLR in accuracy. Mishra and Smyth assessed whether futures prices could predict spot prices, finding that futures prices did not outperform a random walk (RW) model.
Further studies, such as those by Herrera et al., compared traditional econometric models with machine learning methods, demonstrating the latter's superior performance and ability to predict market turning points. Zhang and Hamori combined machine learning techniques with dynamic and expanded windows to forecast crises in the U.S. natural gas market, achieving notable accuracy with the XGboost model.
A comprehensive literature survey by Tamba et al. underscored the difficulty of precisely predicting natural gas price evolution. Nevertheless, it is evident that machine learning approaches yield higher prediction accuracy compared to traditional econometric methods.
In this report, I aim to enhance the predictive capability of gas prices by training models including SVM, regression trees, linear regression, Gaussian process regression (GPR), and ensemble of trees. Our focus is on short-term forecasting of natural gas spot prices 1, 3, 5, and 10 days ahead. I evaluate the effectiveness of these machine learning models against a random walk model.
For model training, I utilized lags of natural gas spot prices and 21 explanatory variables identified from relevant literature. These variables were integrated into the forecasting models through a rigorous training and testing process to determine the most efficient and least error-prone models for natural gas price forecasting.
The report is structured as follows: Section 2 details the methodologies and data utilized in the study, Section 3 presents our empirical results, and Section 4 concludes the report.

2. Methodology

2.1. Support Vector Machines

Support vector machines (SVM) are a set of methods for data classification and regression that maximize the interclass distance. The primary goal of SVM is to define the optimal linear separator that generalizes well to unknown data, effectively separating the data points into two classes. To achieve this, the algorithm employs the "kernel trick," which projects the initial data space into a higher-dimensional space (feature space) where the dataset may become linearly separable. In this study, I utilize four kernels: linear, quadratic, cubic, and three Gaussian kernels (fine, medium, and coarse), each following a different structure in the data.

2.2. Gaussian Process Regression

Gaussian processes are flexible non-parametric machine learning models primarily used for modeling spatial and time series data. They are particularly attractive due to their flexibility and computational simplicity. Gaussian process regression (GPR) determines an appropriate kernel function or a measure of similarity between data points. Compared to other machine learning methods, GPR integrates multiple tasks, such as parameter estimation, and performs well with relatively small training datasets. However, due to computational complexity, GPR becomes infeasible for large datasets. In this report, I train four GPR models with the following kernel functions:

1. Rational Quadratic GPR: uses the rational quadratic kernel,
2. Squared Exponential GPR: uses the squared exponential kernel,
3. Matern 5/2 GPR: uses the Matern 5/2 kernel,
4. Exponential GPR: uses the exponential kernel.

2.3. Decision Trees

Decision trees are a forecasting modeling technique employed in statistics, data mining, and machine learning. They transition from observations of an item (branches) to inferences about the object's target value (leaves). Regression trees are used when the target variable takes continuous values. In this study, I use three different tree models:

1. Fine Tree: minimum leaf size is 4,

2. Medium Tree: minimum leaf size is 12,
3. Coarse Tree: minimum leaf size is 36.

2.4. Ensemble of Trees

An ensemble of trees combines several individual trees to improve prediction performance. While decision trees are efficient and interpretable, they suffer from low generalization ability, providing low bias in-sample but high variance out-of-sample. Ensemble techniques address this by combining weak learners into a strong learner. The main techniques are bagging and boosting.

2.4.1 Bagging

Bagging (bootstrap aggregation) reduces the variance of a decision tree by generating several subsets of data from the training sample, selected randomly with replacement. Each subset trains a corresponding decision tree model, and the average of all predictions is used, yielding a more accurate model than a single decision tree.

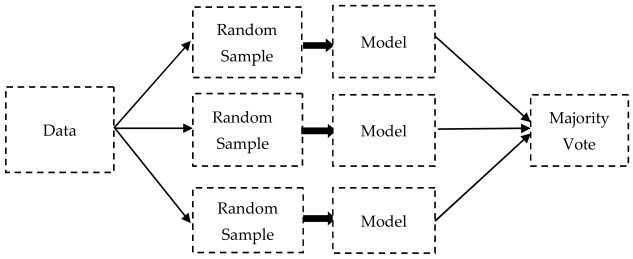


Figure 1: Bagging

2.4.2 Boosting

Boosting improves prediction accuracy by sequentially fitting models to the data. Each successive model focuses on correcting the errors of the previous models by using a weighted data sample. This process continues iteratively, combining all models to form a stronger predictor.

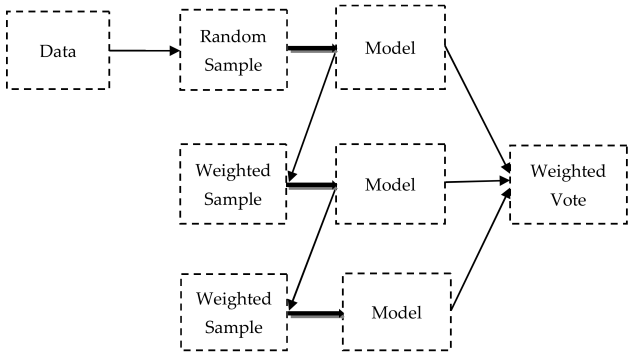


Figure 2: Boosting

2.5. Cross-Validation

To avoid overfitting, I employed a k-fold cross-validation procedure. The in-sample data are divided into k parts (folds) of equal size. In each iteration, one fold is used as the testing set, while the remaining folds are used for training. This process repeats for all k folds. The model's accuracy is evaluated by the average performance over all k folds.

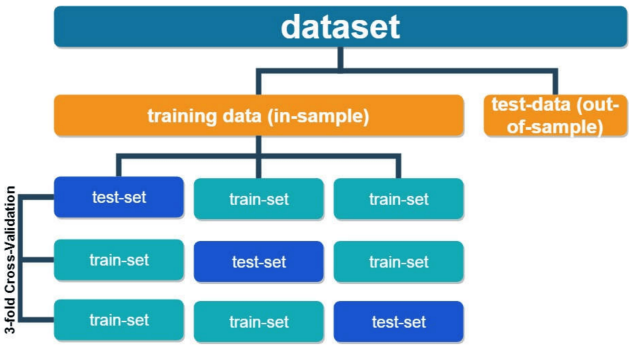


Figure 3: A three-fold cross-validation for a given set of model parameter values. Each fold serves as a test sample, while the remaining folds are used to train the model. The average prediction accuracy over the k-folds assesses the model

2.6. The Dataset

For training and testing our models, I compiled a dataset of 2423 daily natural gas spot price values from the Energy Information Administration (EIA) database and 21 related economic variables from the Federal Reserve Bank of Saint Louis and Yahoo Finance databases. The data span from December 3, 2010, to September 18, 2020. Additionally, I calculated the momentum of the last 5 and 10 days and the 5- and 10-day moving average, adding these to the independent variable set. All variables, except interest rates, were converted to natural logarithms.

#	Name	Mean	Standard Deviation	Skewness	Kurtosis	Variance
Panel A: Stock Indices						
1	NASDAQ Composite Index	5289.44	2132.48	0.61	−0.4	4,549,055
2	S&P 500 Index	2112.97	607.77	0.18	−0.92	369,500
3	Dow Jones Industrial Average Index	18,797.36	5195.66	0.3	−1.04	27,003,372
Panel B: Exchange Rates						
4	USD/EUR	0.19	0.09	0.3	−1.29	0.008
5	JPY/USD	4.62	0.14	−0.82	−0.61	0.019
6	USD/GBP	0.37	0.1	0.19	−1.5	0.010
Panel C: WTI Spot Price						
7	Cushing, OK WTI Spot Price FOB	4.17	0.37	−0.57	0.34	0.1401
Panel D: Interest Rates						
8	Effective Federal Funds Rate	0.638	0.77	1.17	−0.15	0.5974
9	5-Year Breakeven Inflation Rate	1.7	0.32	−0.84	1.57	0.107
10	10-Year Breakeven Inflation Rate	1.94	0.33	−0.44	0.36	0.1142
11	1-Year Treasury Constant Maturity Rate	0.75	0.82	1.1	−0.23	0.6757
12	10-Year Treasury Constant Maturity Rate	2.22	0.61	−0.4	0.51	0.3736
13	Bank Prime Loan Rate	3.76	0.75	1.18	−0.12	0.5732
Panel E: Future Contracts						
14	Natural Gas Futures Contract 1	1.1	0.26	−0.171	−0.51	0.0686
15	Natural Gas Futures Contract 2	1.11	0.25	−0.174	−0.62	0.0626
16	Natural Gas Futures Contract 3	1.13	0.24	−0.163	−0.69	0.0573
17	Natural Gas Futures Contract 4	1.15	0.23	−0.096	−0.75	0.0519
18	OK Crude Oil Future Contract 1	4.181	0.371	−0.53	0.15	0.138
19	OK Crude Oil Future Contract 2	4.189	0.358	−0.37	−0.45	0.1283
20	OK Crude Oil Future Contract 3	4.195	0.348	−0.27	−0.81	0.1213
21	OK Crude Oil Future Contract 4	4.199	0.341	−0.22	−0.95	0.1165

Figure 4: List of explanatory variables along with their mean, standard deviation, skewness, kurtosis, and variance

The dataset was divided into two parts: the first 90% served as the training set (in-sample, 2180 observations), and the remaining 10% was the test set (out-of-sample, 243 observations).

3. Empirical Results

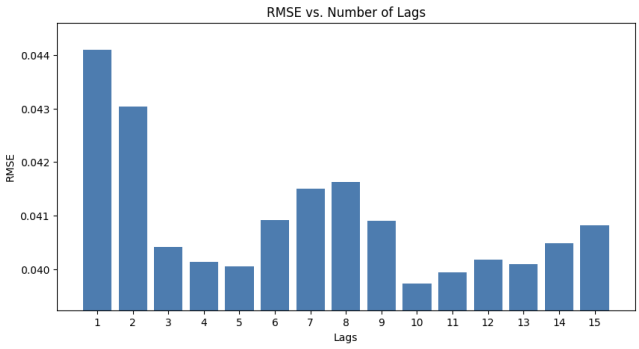
The accuracy of each model's predictions, both for in-sample and out-of-sample data, was assessed using the Root Mean Square Error (RMSE) metric. The model with the lowest RMSE was deemed the most effective:

$$\text{RMSE} = \sqrt{\frac{\sum_{t=1}^{T}(\hat{y}_t - y_t)^2}{T}}$$

where $\hat{y}$ represents the predicted value, y is the actual value, and T denotes the number of observations. Forecasts were generated for various time horizons: 1 day, 3 days, 5 days, and 10 days ahead. To benchmark the performance of our machine learning models, I also implemented a random walk model for comparison. Prior to developing structural models (those incorporating the independent variables from our dataset), I focused on identifying the optimal autoregressive representation, specifically the best AR(q) model. This model leverages past values of natural gas spot prices to predict future prices:

$$X_t = c + \sum_{i=1}^{q} \phi_i X_{t-i} + \epsilon_t$$

where X is the natural gas spot price, q represents the maximum number of lags, and ϕ_i is the parameter vector for the lags.
To determine the optimal number of lags, I trained multiple linear SVM models, each with a different number of lags, starting from AR(1) up to AR(15).
By using the first 10 lags, the RMSE is minimized (0.0397).



To evaluate the effectiveness of different machine learning models, I trained various alternative models and compared them to the random walk model. The RMSE values for both in-sample and out-of-sample predictions are summarized in Table 2. One critical challenge in developing forecasting models is avoiding overfitting, which occurs when a model performs well on training data but poorly on new, unseen data. This issue is often referred to as the bias–variance trade-off in the literature. A robust forecasting model should strike a balance between bias and variance, ensuring consistent performance across both in-sample and out-of-sample datasets.

Listing 1: Python code for model training and evaluation

```
// Standardize the features
scaler = StandardScaler()
X_train_scaled = scaler.fit_transform(X_train)
X_test_scaled = scaler.transform(X_test)

// Define the models to be tested
models = {
    'Random Walk': None,
    'Linear Regression': LinearRegression(),
    'Robust Linear Regression': HuberRegressor(max_iter=1000),
    'SVM Linear': SVR(kernel='linear'),
    'SVM Quadratic': SVR(kernel='poly', degree=2),
    'SVM Cubic': SVR(kernel='poly', degree=3),
    'SVM Gaussian Fine': SVR(kernel='rbf', gamma='scale'),
    'SVM Gaussian Medium': SVR(kernel='rbf', gamma='auto'),
    'SVM Gaussian Coarse': SVR(kernel='rbf', gamma=0.1),
    'GPR Rational Quadratic': GaussianProcessRegressor(kernel=
      ↪ RationalQuadratic(), alpha=1e-5),
    'GPR Squared Exponential': GaussianProcessRegressor(kernel=RBF(), alpha
      ↪ =1e-5),
```

```
    'GPR Matern 5/2': GaussianProcessRegressor(kernel=Matern(nu=2.5), alpha
        ↪ =1e-5),
    'Fine Tree': DecisionTreeRegressor(max_depth=5),
    'Medium Tree': DecisionTreeRegressor(max_depth=10),
    'Coarse Tree': DecisionTreeRegressor(max_depth=15),
    'Bagged Trees': BaggingRegressor(),
    'Boosted Trees': GradientBoostingRegressor()
}

// Define RMSE scoring function
def rmse_score(y_true, y_pred):
    return np.sqrt(mean_squared_error(y_true, y_pred))

rmse_scorer = make_scorer(rmse_score, greater_is_better=False)

// Initialize list to store model results
model_results = []
for name, model in models.items():
    if model is not None:
        try {
            // Perform cross-validation
            scores = cross_val_score(model, X_train_scaled, y_train, cv=
                ↪ TimeSeriesSplit(n_splits=5), scoring=rmse_scorer)
            in_sample_rmse = -scores.mean()

            // Fit model and predict
            model.fit(X_train_scaled, y_train)
            y_pred = model.predict(X_test_scaled)
            out_sample_rmse = rmse_score(y_test, y_pred)

            // Store results
            model_results.append({
                'Model': name,
                'In-Sample RMSE': in_sample_rmse,
                'OOS RMSE': out_sample_rmse,
            })
        } except Exception as e {
            model_results.append({
                'Model': name,
                'In-Sample RMSE': str(e),
                'OOS RMSE': 'Error',
            })
        }
    else:
        if name == 'Random Walk':
            y_pred = np.roll(y_train, 1)
            y_pred[0] = y_train[0]
            in_sample_rmse = rmse_score(y_train, y_pred)
            y_pred_test = np.roll(y_test, 1)
            y_pred_test[0] = y_test[0]
            out_sample_rmse = rmse_score(y_test, y_pred_test)
            model_results.append({
                'Model': name,
                'In-Sample RMSE': in_sample_rmse,
                'OOS RMSE': out_sample_rmse,
            })

// Convert results to DataFrame and print
results_df = pd.DataFrame(model_results)
print(results_df)
```

To address overfitting, I excluded models that showed signs of overfitting from further analysis. Notably, tree-based models, including bagged and boosted trees, did not overfit, whereas all Gaussian Process Regression (GPR) models and most Support Vector Machine (SVM) models, except for the linear SVM, did show overfitting tendencies.

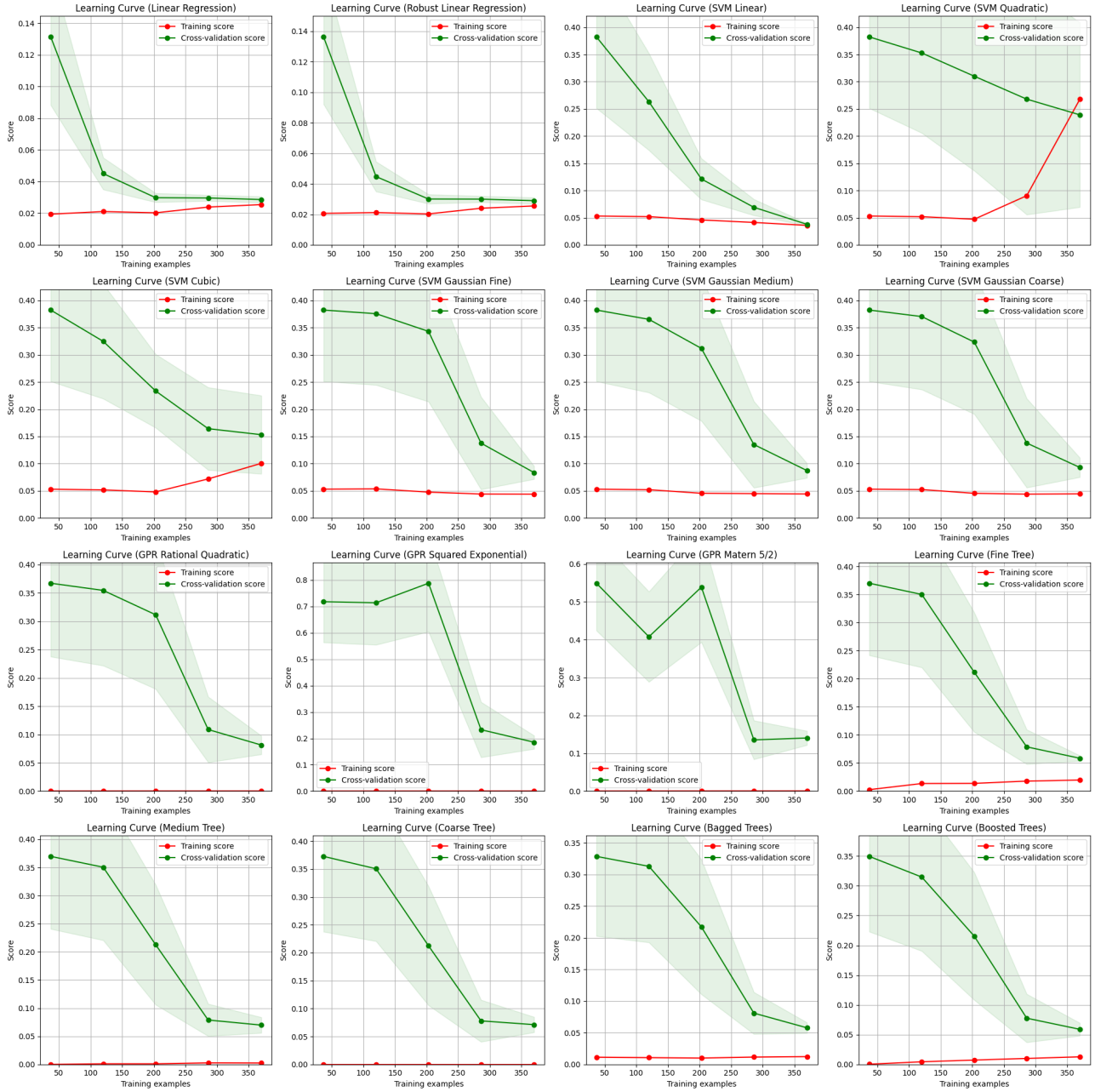


Figure 5: Learning curves for the models used in natural gas spot price forecasting.

3.1. Time Frame t + 1

For the one-day-ahead forecast (t + 1), the random walk model demonstrated surprisingly competitive performance, achieving an in-sample RMSE of 0.027760 and an out-of-sample RMSE of 0.037453. This performance was comparable to the best machine learning models, such as linear regression and robust linear regression, which had slightly higher RMSEs both in-sample and out-of-sample. The random walk model's simplicity makes it a strong baseline, especially for very short-term forecasts.

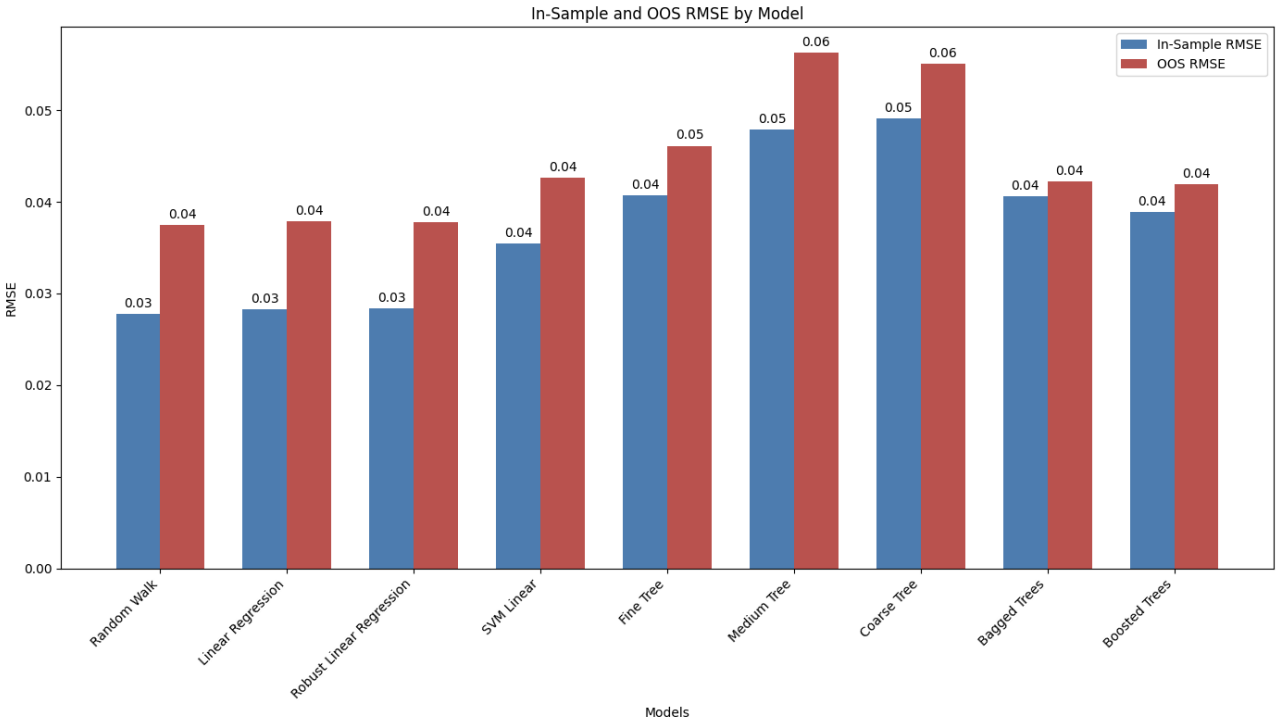


Figure 6: RMSE for t + 1 forecasting.

Model	In-Sample RMSE	OOS RMSE
Random Walk	0.0278	0.0375
Linear Regression	0.0283	0.0378
Robust Linear Regression	0.0284	0.0377
SVM Linear	0.0354	0.0426
Fine Tree	0.0407	0.0461
Medium Tree	0.0479	0.0563
Coarse Tree	0.0491	0.0551
Bagged Trees	0.0406	0.0422
Boosted Trees	0.0389	0.0419

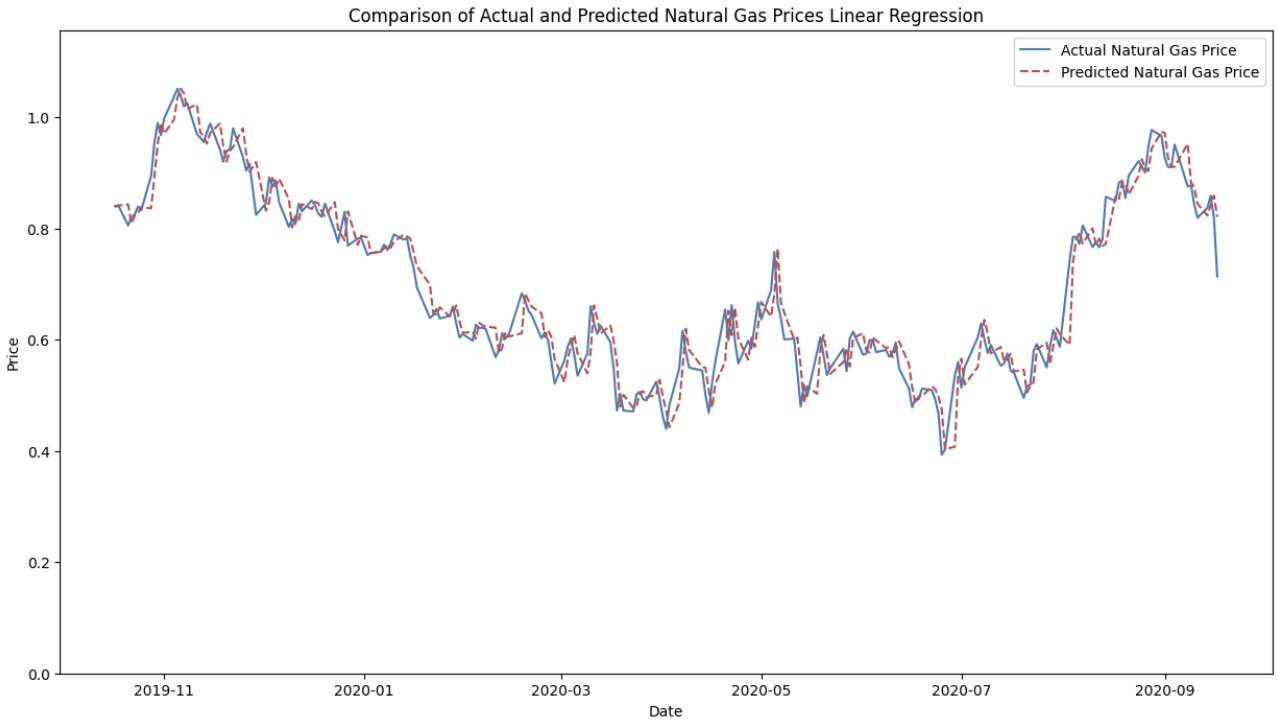


Figure 7: Comparison of the actual natural gas spot prices and the predicted prices with the robust linear regression model for t + 1 in the out-of-sample part of the dataset.

Listing 2: Python code for evaluating and comparing models across different time horizons

```
// Initialize list to store model results and define time horizons
model_results = []
time_horizons = [3, 5, 10]

// Loop through each time horizon
for shift in time_horizons:
    // Prepare the shifted training and testing datasets
    X_train_shifted, y_train_shifted = prepare_data(X_train_scaled, y_train,
        ↪  shift)
    X_test_shifted, y_test_shifted = prepare_data(X_test_scaled, y_test,
        ↪ shift)

    // Initialize list to store results for each model at the current time
        ↪ horizon
    results_by_model = []

    // Loop through each model
    for name, model in models.items():
        if model is not None:
            // Perform cross-validation and fit the model
            scores = cross_val_score(model, X_train_shifted, y_train_shifted
                ↪ , cv=TimeSeriesSplit(n_splits=5), scoring=rmse_scorer)
            model.fit(X_train_shifted, y_train_shifted)
            y_pred = model.predict(X_test_shifted)

            // Calculate in-sample and out-of-sample RMSE
            in_sample_rmse = -scores.mean()
            out_sample_rmse = rmse_score(y_test_shifted, y_pred)

            // Append results to the list
            results_by_model.append({
                'Model': name,
```

```
            'In-Sample RMSE': in_sample_rmse,
            'Out-of-Sample RMSE': out_sample_rmse
        })

// Convert results to a DataFrame
results_df_shift = pd.DataFrame(results_by_model)

// Identify the best models based on RMSE
best_in_sample_model = results_df_shift.loc[results_df_shift['In-Sample
    ↪ RMSE'].idxmin()]
best_out_sample_model = results_df_shift.loc[results_df_shift['Out-of-
    ↪ Sample RMSE'].idxmin()]

// Print RMSE scores in ascending order
print("\nIn-Sample RMSE Scores in Ascending Order:")
print(results_df_shift[['Model', 'In-Sample RMSE']].sort_values(by='In-
    ↪ Sample RMSE'))

print("\nOut-of-Sample RMSE Scores in Ascending Order:")
print(results_df_shift[['Model', 'Out-of-Sample RMSE']].sort_values(by='
    ↪ Out-of-Sample RMSE'))

// Print best models for the current time horizon
print(f"Results for t+{shift}:")
print(f"Best In-Sample Model: {best_in_sample_model['Model']} with RMSE:
    ↪  {best_in_sample_model['In-Sample RMSE']:.4f}")
print(f"Best Out-of-Sample Model: {best_out_sample_model['Model']} with
    ↪ RMSE: {best_out_sample_model['Out-of-Sample RMSE']:.4f}")

// Plot actual vs predicted prices for the best out-of-sample model
if best_out_sample_model['Model']:
    plt.figure(figsize=(10, 5))
    plt.plot(y_test_shifted.index, y_test_shifted, label='Actual Prices'
        ↪ , color='#4d7caf')
    plt.plot(y_test_shifted.index, y_pred, label='Predicted Prices by '
        ↪ + best_out_sample_model['Model'], linestyle='--', color='#
        ↪ b9524e')
    plt.title(f'Forecasting Results for {best_out_sample_model["Model"]}
        ↪  at t+{shift}')
    plt.xlabel('Date')
    plt.ylabel('Price')
    plt.legend()
    plt.show()

// Plot bar chart of RMSE scores
x = np.arange(len(results_df_shift['Model']))
width = 0.35

fig, ax = plt.subplots(figsize=(14, 8))
rects1 = ax.bar(x - width/2, results_df_shift['In-Sample RMSE'], width,
    ↪ label='In-Sample RMSE', color='#4d7caf')
rects2 = ax.bar(x + width/2, results_df_shift['Out-of-Sample RMSE'],
    ↪ width, label='OOS RMSE', color='#b9524e')

ax.set_xlabel('Models')
ax.set_ylabel('RMSE')
ax.set_title(f'In-Sample and OOS RMSE by Model at t+{shift}')
ax.set_xticks(x)
ax.set_xticklabels(results_df_shift['Model'], rotation=45)
ax.legend()
```

```
plt.tight_layout()
plt.show()

// Display results as a table
results_df_shift = results_df_shift.applymap(lambda x: round(x, 4) if
    ↪ isinstance(x, (float, int)) else x)
fig, ax = plt.subplots(figsize=(12, 1))
ax.axis('tight')
ax.axis('off')
the_table = ax.table(cellText=results_df_shift.values, colLabels=
    ↪ results_df_shift.columns, loc='center', cellLoc='center',
    ↪ colColours=['none', '#4d7caf', '#b9524e'])
the_table.auto_set_font_size(False)
the_table.set_fontsize(10)
the_table.scale(1, 1.4)
plt.show()
```

3.2. Time Frame t + 3

The results for the three-day-ahead forecast (t + 3) are illustrated in Figure 7. For this time horizon, I determined that the linear regression model was the most effective for in-sample data, achieving an RMSE of 0.0283. For out-of-sample predictions, the robust linear regression model delivered the best performance with an RMSE of 0.0379. The SVM Linear also behaved well, giving the third-best in-sample score: 0.0354, and fourth out-of-sample score: 0.0428.

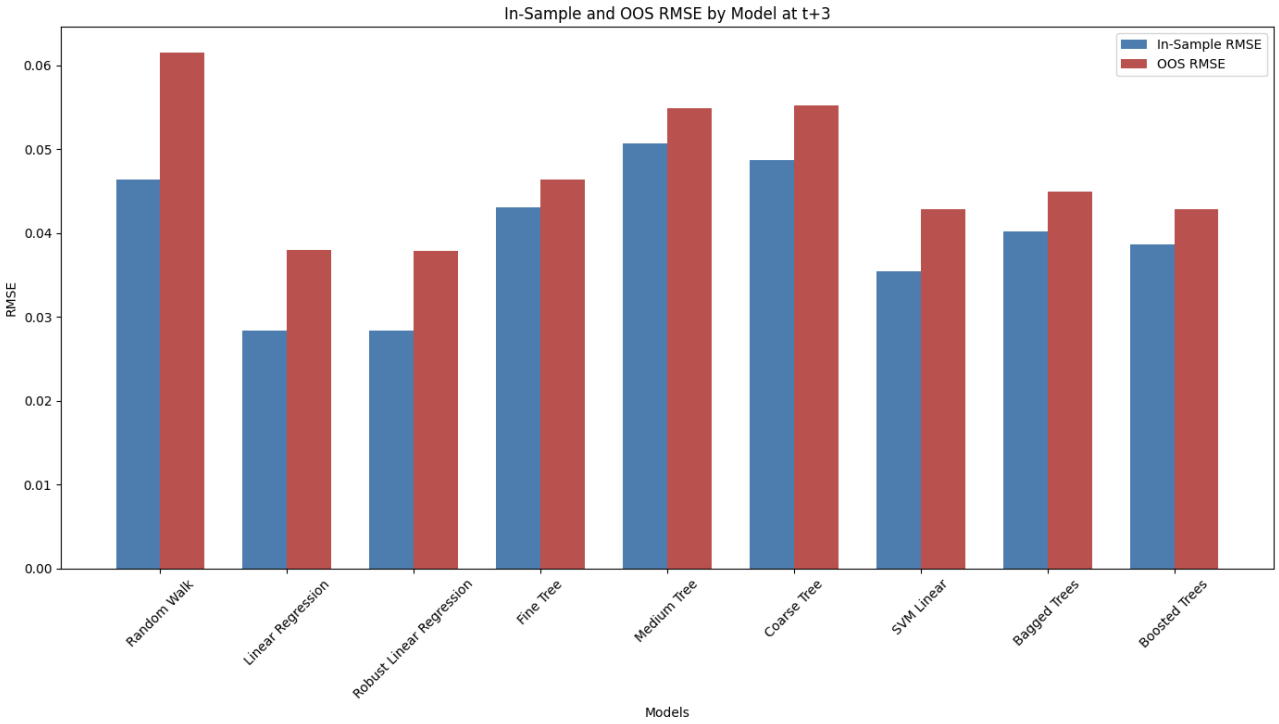


Figure 8: RMSE for t + 3 forecasting.

Model	In-Sample RMSE	Out-of-Sample RMSE
Random Walk	0.0464	0.0615
Linear Regression	0.0283	0.038
Robust Linear Regression	0.0284	0.0379
Fine Tree	0.043	0.0464
Medium Tree	0.0507	0.0549
Coarse Tree	0.0487	0.0553
SVM Linear	0.0354	0.0428
Bagged Trees	0.0402	0.045
Boosted Trees	0.0386	0.0429

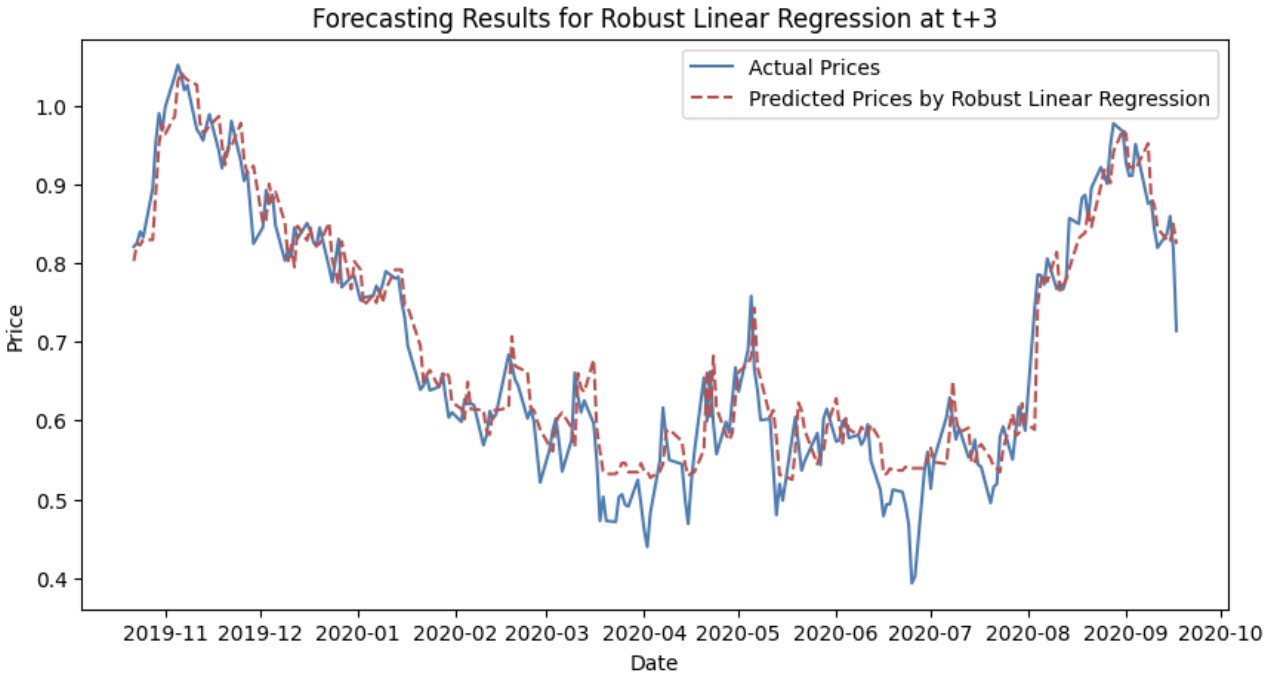


Figure 9: Comparison of the actual natural gas spot prices and the predicted prices with the robust linear regression model for t + 3 in the out-of-sample part of the dataset.

3.3. Time Frame t + 5

For the five-day-ahead forecast ($t + 5$), both the linear regression model and robust linear regression model demonstrated the best performance for in-sample data, achieving an RMSE of 0.0282. Other notable in-sample performers included the SVM linear model (RMSE: 0.035), boosted trees (RMSE: 0.0385), and bagged trees (RMSE: 0.0392). The fine tree, medium tree, and coarse tree models showed higher in-sample RMSEs of 0.0426, 0.0483, and 0.0495, respectively.

When evaluating out-of-sample performance, the robust linear regression model emerged as the most accurate, with an RMSE of 0.0381. The linear regression model also performed well, with an RMSE of 0.0382. Boosted trees and SVM linear models followed, with RMSEs of 0.0427 and 0.0430, respectively. Bagged trees (RMSE: 0.0457), fine tree (RMSE: 0.0465), medium tree (RMSE: 0.0560), and coarse tree (RMSE: 0.0574) models exhibited higher out-of-sample RMSEs.

In summary, for the $t + 5$ forecast horizon, the linear regression model proved to be the best in-sample model, while the robust linear regression model provided the most accurate out-of-sample predictions. Both linear regression and robust linear regression models showcased strong performance, highlighting their suitability for short-term natural gas spot price forecasting.

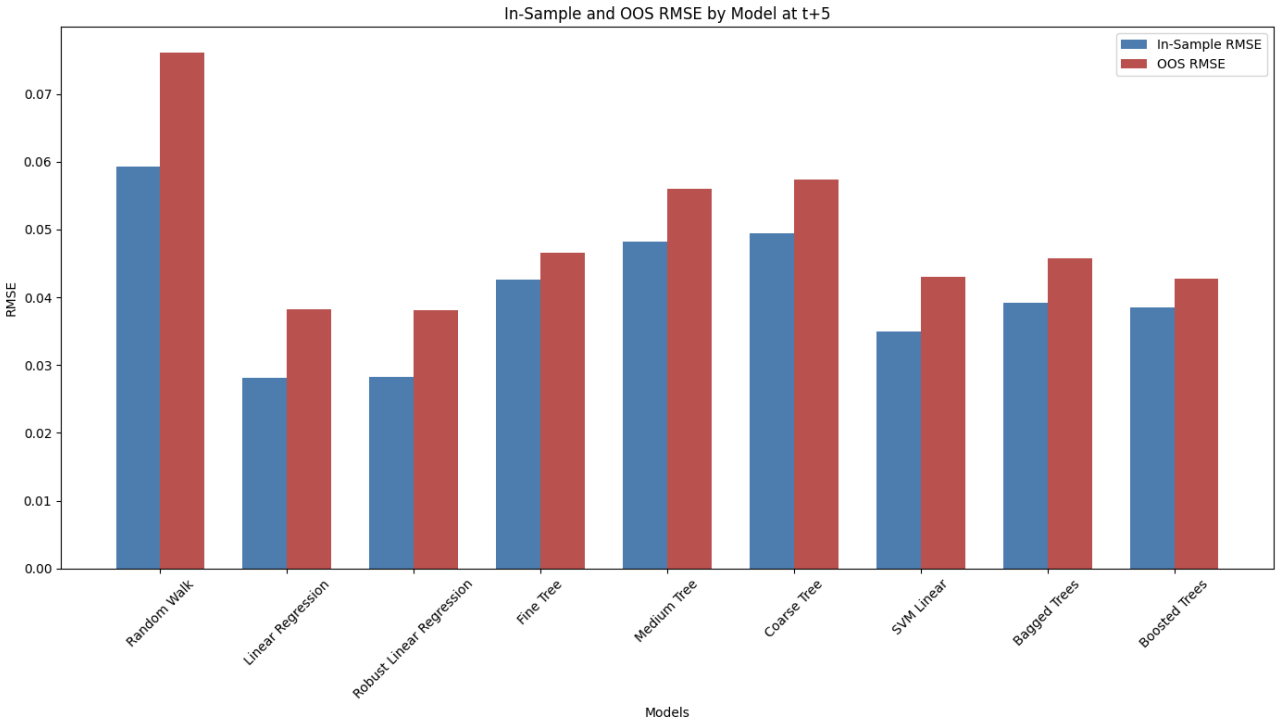


Figure 10: RMSE for t + 5 forecasting.

Model	In-Sample RMSE	Out-of-Sample RMSE
Random Walk	0.0593	0.0761
Linear Regression	0.0282	0.0382
Robust Linear Regression	0.0282	0.0381
Fine Tree	0.0426	0.0465
Medium Tree	0.0483	0.056
Coarse Tree	0.0495	0.0574
SVM Linear	0.035	0.043
Bagged Trees	0.0392	0.0457
Boosted Trees	0.0385	0.0427

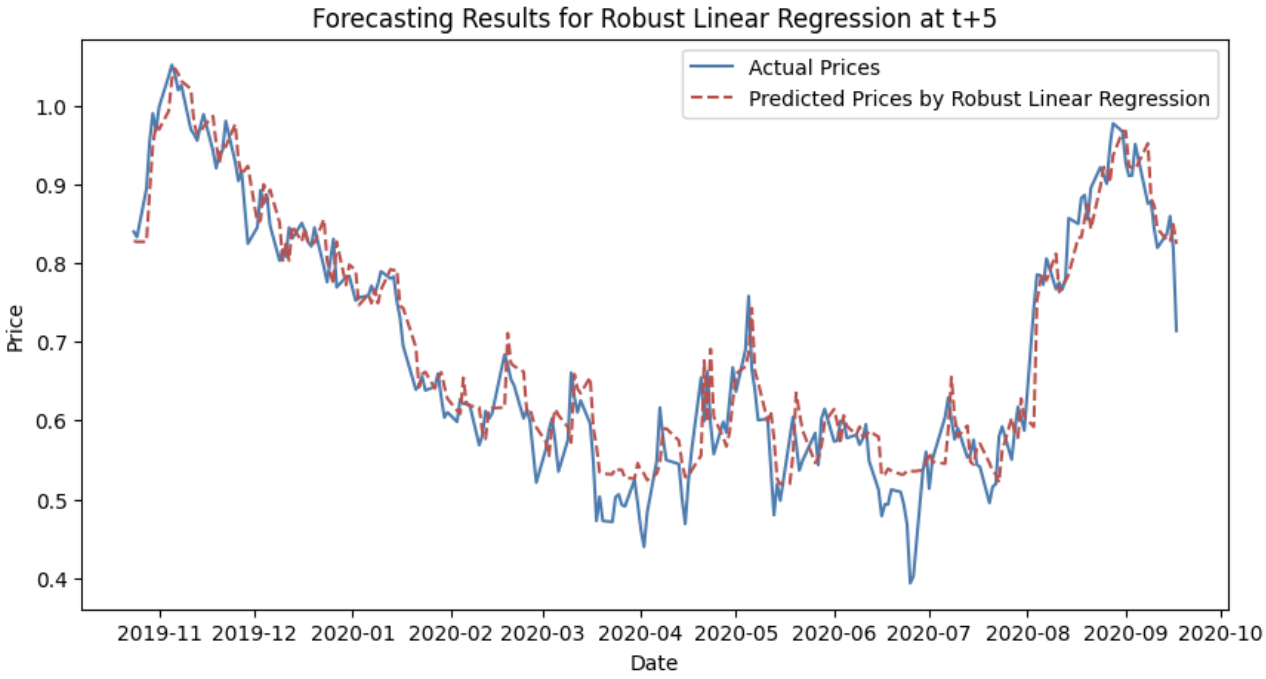


Figure 11: Comparison of the actual natural gas spot prices and the predicted prices with the robust linear regression model for t + 5 in the out-of-sample part of the dataset.

3.4. Time Frame t + 10

For the ten-day-ahead forecast $(t + 10)$, the linear regression model demonstrated the best performance for in-sample data, achieving an RMSE of 0.0281. This was closely followed by the robust linear regression model with an RMSE of 0.0282. Other notable in-sample performers included the SVM linear model (RMSE: 0.0355), boosted trees (RMSE: 0.0388), and bagged trees (RMSE: 0.0392). The fine tree, coarse tree, and medium tree models showed higher in-sample RMSEs of 0.0406, 0.0483, and 0.0494, respectively.
When evaluating out-of-sample performance, the robust linear regression model emerged as the most accurate, with an RMSE of 0.0380. The linear regression model also performed well, with an RMSE of 0.0381. Boosted trees and the SVM linear model followed, with RMSEs of 0.0415 and 0.0432, respectively. Bagged trees also performed similarly, with an RMSE of 0.0451. The fine tree (RMSE: 0.0467), coarse tree (RMSE: 0.0534), and medium tree (RMSE: 0.0567) models exhibited higher out-of-sample RMSEs.
In summary, for the $t + 10$ forecast horizon, the linear regression model proved to be the best in-sample model, while the robust linear regression model provided the most accurate out-of-sample predictions. Both linear regression and robust linear regression models showcased strong performance, highlighting their suitability for short-term natural gas spot price forecasting over a ten-day period.

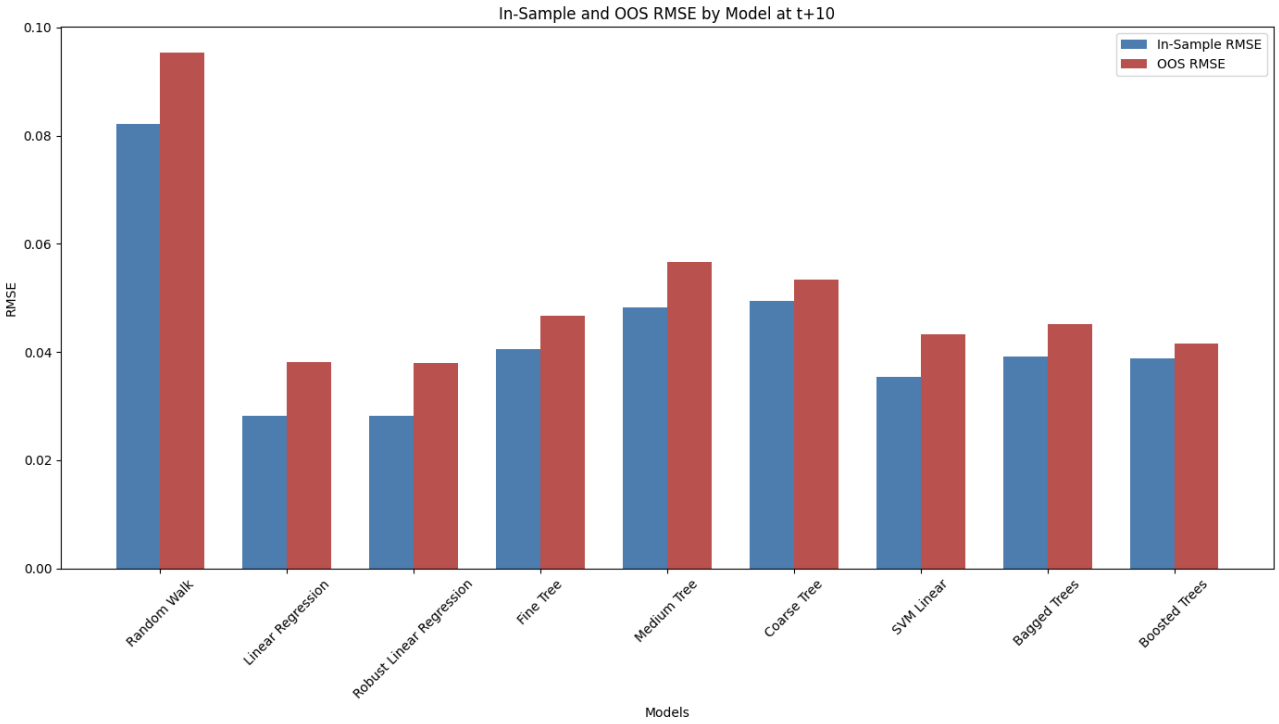


Figure 12: RMSE for t + 10 forecasting.

Model	In-Sample RMSE	Out-of-Sample RMSE
Random Walk	0.0822	0.0953
Linear Regression	0.0281	0.0381
Robust Linear Regression	0.0282	0.038
Fine Tree	0.0406	0.0467
Medium Tree	0.0483	0.0566
Coarse Tree	0.0494	0.0534
SVM Linear	0.0355	0.0432
Bagged Trees	0.0392	0.0451
Boosted Trees	0.0388	0.0415

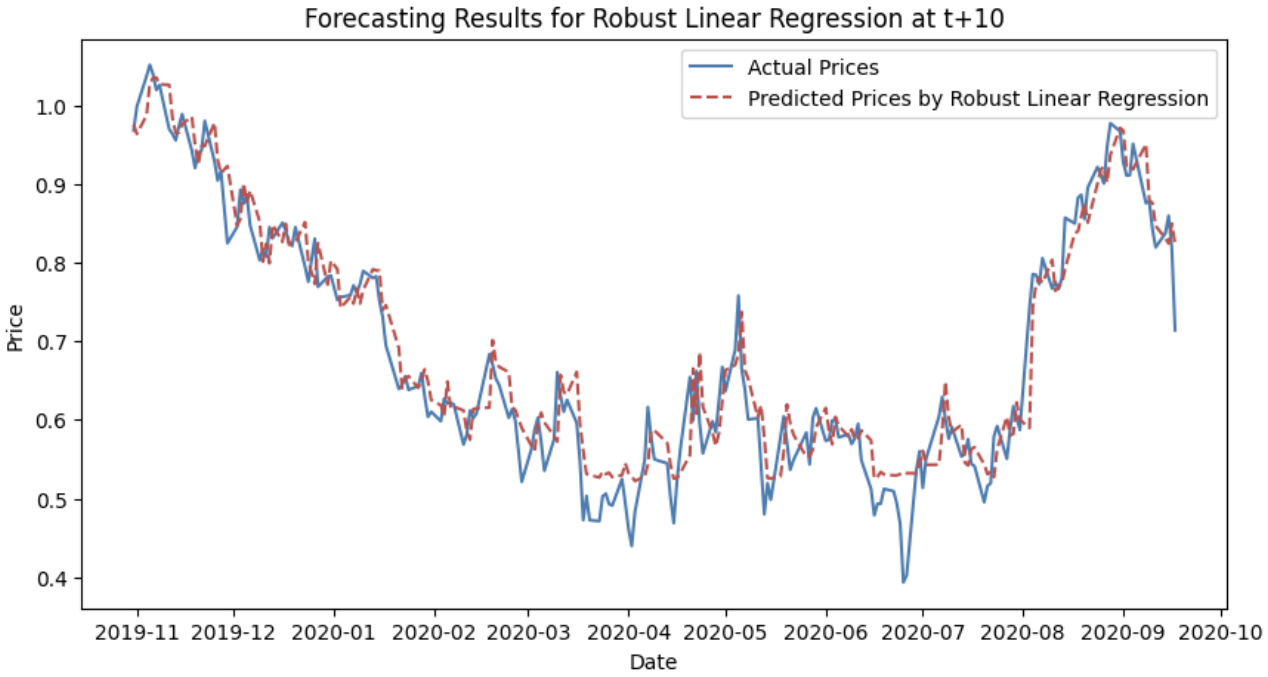


Figure 13: Comparison of the actual natural gas spot prices and the predicted prices with the robust linear regression model for t + 10 in the out-of-sample part of the dataset.

Overall, while the random walk model performed surprisingly well for the t + 1 horizon, linear and robust linear regression models consistently demonstrated the best performance across different forecasting horizons, particularly for out-of-sample predictions. These results highlight the robustness and reliability of linear models in predicting natural gas spot prices.

4. Conclusions

Accurate forecasting of natural gas prices has significant practical implications. It enables stakeholders on both the supply and demand sides to reduce associated risks by anticipating future price changes. This anticipation allows them to optimize their participation in the market, whether through storage strategies, substitution decisions, or budget adjustments, ultimately reducing uncertainty for suppliers and consumers alike. Moreover, government officials can leverage this information for large-scale planning, better anticipating price swings and managing resources effectively.

The effective forecasting of natural gas prices is critical for suppliers, distributors, consumers, investors, and regulatory agencies. Accurate predictions aid in making informed decisions for risk management, reducing the demand-supply gap, and optimizing resource utilization. For investors in the U.S. energy equity markets, especially amid the current focus on green energy, coping with information frictions, market risk fluctuations, and regulatory changes is crucial for sound investment decisions.

In this report, I evaluated the effectiveness of various machine learning algorithms in forecasting natural gas spot prices. I trained multiple machine learning models, including a naïve random walk model. These models used the optimal number of lagged natural gas spot prices and 21 other explanatory variables, selected based on economic theory and relevant literature. The variables included macroeconomic indicators, stock market indicators, exchange rates, interest rates, spot prices and futures contracts of Oklahoma West Texas Intermediate Crude Oil, natural gas futures contracts, momentum of the last 5 and 10 days, and the 5- and 10-day moving averages. The models were trained to forecast horizons of one, three, five, and ten days ahead ($t+1$, $t+3$, $t+5$, and $t+10$).

The dataset comprised 2423 daily observations from December 3, 2010, to September 18, 2020. This dataset was divided into two subsets: the first subset, spanning November 19, 2010, to September 19, 2019, included 2180 observations used for training the models. The second subset, from September 20, 2019, to September 18, 2020, included 243 observations used to test the models' generalization abilities. To avoid overfitting, I employed a 5-fold cross-validation method.

I determined the optimal autoregressive representation to be 10 lags using a linear SVM model. Subsequently, I trained 19 models. Ten models exhibited overfitting and were excluded from further analysis. These overfitted models included the interaction linear, SVM (quadratic, cubic, fine Gaussian, medium Gaussian, coarse Gaus-

sian), and GPR (squared exponential, Matern 5/2, exponential, and rational quadratic) models. The models that did not show overfitting were the random walk, linear regression, robust linear regression, fine tree, medium tree, coarse tree, linear SVM, boosted trees, and bagged trees models.

The results showed that the random walk model performed surprisingly well for the $t + 1$ horizon, with an in-sample RMSE of 0.027760 and an out-of-sample RMSE of 0.037453. However, linear regression and robust linear regression models consistently demonstrated the best performance across different forecasting horizons, particularly for out-of-sample predictions. For $t + 3$, $t + 5$, and $t + 10$ horizons, the robust linear regression model provided the best out-of-sample performance.

The bagged trees model also exhibited strong performance, particularly for in-sample data, indicating its robustness and reliability in capturing complex patterns within the dataset. However, its out-of-sample performance was slightly less competitive compared to the linear and robust linear regression models.

The boosted trees model showed good performance across various horizons, particularly in out-of-sample predictions. For instance, it was one of the top-performing models for the $t + 3$ horizon. The ability of boosted trees to combine weak learners into a strong predictive model makes it a valuable tool for forecasting tasks, although it occasionally lagged behind the linear models in terms of overall accuracy.

In conclusion, the most effective methods for natural gas spot price forecasting, as evidenced by my analysis, are the linear regression and robust linear regression models. These models demonstrated strong performance and reliability, making them suitable for short-term natural gas spot price forecasting. Additionally, the bagged trees and boosted trees models also provided valuable insights and performed well, highlighting their potential utility in specific forecasting scenarios.

Acknowledgments

I would like to extend my sincere gratitude to the authors of the report "A Survey on the Gas Price Forecasting Based on Machine Learning" published in MDPI Energies for their valuable insights and methodologies that significantly contributed to my research. The comprehensive review and empirical analysis presented in their work provided a solid foundation for my study on natural gas spot price forecasting. Their findings and discussions have been instrumental in shaping the direction and depth of my research. You can find their detailed report at MDPI Energies.