Enhancing Hydrogen Consumption Predictions in Refinery Hydrotreating through Machine Learning Techniques

Alister Marc Domilies, Leo Julongbayan, Jemar Laag
National Graduate School of Engineering
University of the Philippines Diliman
Quezon City, Philippines
alistermarcdomilies@gmail.com, lljulongbayan@gmail.com, jelaag@up.edu.ph

Abstract

Accurate prediction of hydrogen consumption in refinery hydrotreating processes is crucial for maintaining efficient refinery operations, as misestimations can lead to hydrogen waste or operational inefficiencies. However, the complex and non-linear nature of hydrogen consumption limits the performance of traditional regression models. This study aims to enhance the accuracy of hydrogen consumption predictions by employing machine learning techniques. We present a comprehensive methodology that encompasses outlier detection, dimensionality reduction, and advanced regression modeling. Initially, LazyPredict is utilized to determine the most suitable models. It reveals that Extra Trees and Random Forest exhibited the lowest Root Mean Square Error (RMSE) of the Test Data and were used as baseline models for subsequent machine learning pipeline. Different outlier detection techniques are evaluated to identify and mitigate the impact of anomalous data points. Recursive Feature Elimination is then utilized to reduce the dimensionality of the feature space, retaining only the most relevant features. Subsequently, ensemble regression models, specifically Random Forest and Extra Trees, are implemented and optimized through hyperparameter tuning and cross-validation. The results demonstrate that the Tuned Extra Trees Regressor, combined with outlier detection and dimensionality reduction, achieves superior performance with a Root Mean Square Error (RMSE) of 356 and an R-squared (R^2) of 0.99, significantly outperforming traditional linear regression models. The study highlights the potential of machine learning techniques to enhance the accuracy and robustness of hydrogen consumption predictions.

Keywords: Hydrogen Consumption, Hydrotreaters, Random Forest Regression, Extra Trees Regression

1 Introduction

Refineries heavily depend on hydrogen for purifying petroleum products through processes like hydrotreating, where hydrogen plays a crucial role in removing impurities and enhancing product quality. Hydrotreating involves the reaction of hydrocarbons with hydrogen to convert contaminants such as organic sulfur, nitrogen, metals, and olefins into stabilized products. These reactions take place in a reactor under high temperatures and pressures, facilitated by hydrotreating catalysts. The simplified diagram below illustrates the hydrotreating process, where hydrogen and hydrocarbon feed with contaminants react in a reactor, and the contaminants are subsequently removed in a separator, yielding a product with minimal impurities.

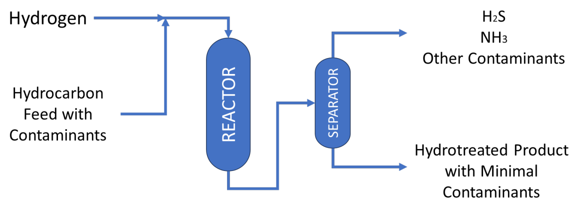
Hydrogen
Hydrocarbon Feed with Contaminants
REACTOR
SEPARATOR
H_2S
NH_3
Other Contaminants
Hydrotreated Product with Minimal Contaminants

Figure 1: Overview of Hydrotreater Process

The significance of hydrotreating has grown as refineries strive to comply with stringent emission standards such as Euro IV, which require reducing sulfur content to less than 50 parts per million (ppm). This reduction in sulfur content not only helps in minimizing the production of harmful combustion byproducts, including nitrogen oxides and sulfur oxides, but also enhances the efficiency and performance of fuels used by consumers. Achieving these standards is crucial for environmental protection and the overall efficiency and competitiveness of refinery outputs.

Given the extensive use of hydrogen in refineries by hydrotreaters and other processes, maintaining a balanced hydrogen supply is essential. The efficient operation of a refinery hinges on the availability of hydrogen, which needs to be prepared in advance. Therefore, accurate prediction of hydrogen consumption is critical. Overestimating hydrogen consumption can result in hydrogen waste, while underestimating it can lead to potential operational issues and inefficiencies.

Accurately predicting hydrogen consumption remains a significant challenge due to its highly complex and non-linear nature, compounded by the dependency on numerous operational parameters. Several studies have attempted to model hydrogen consumption in hydrotreaters. For instance, a regression model for hydrogen consumption was developed using different approaches. The best approach involved analyzing the H_2 content in the gas stream, where a regression model was created by performing statistical analysis of the effects of process conditions on H_2 consumption data. Despite improvements over previous models, the regression model still deviated from actual H_2 consumption by around 12-15% (Pinos et al., 2019). This aligns with the research that mentioned quick calculations to predict hydrogen consumption can have errors as high as 15%. Different correlations were used to estimate hydrogen consumption, determining that different correlations should be used for different product qualities. While this method is more accurate, it requires determining the individual hydrogen consumption of different feeds using different correlations. This approach is not suitable for mixed feeds due to interactions and other non-linear complexities (Castañeda et al., 2011).

To address these challenges, researchers aim to create a prediction model to accurately predict hydrogen consumption using machine learning. Machine learning can create models that learn complex non-linear relationships from data, and it has been widely used in various processes, such as predicting product properties in a refinery. For example, artificial neural network and random forest models were trained to predict the outlet gas concentrations of a reformer and regenerator in a sorption-enhanced steam methane reforming process (Nkulikiyinka et al., 2020). A data-driven soft sensor was developed using deep learning to estimate the oxygen content in flue gases in a 1000-MW ultrasupercritical unit (Yan et al., 2017). A soft sensor was built to estimate the composition of C4 hydrocarbons in the distillate stream of a splitter column (Ferreira et al., 2022).

The prediction of hydrogen consumption using correlations or simple regression techniques has proven difficult due to the complexity of the task. However, machine learning may be well-suited for this type of problem, as it can learn patterns from actual refinery data. So far, there are no known studies that have used machine learning to predict the hydrogen consumption of a hydrotreater.

2 Objectives

The objectives of this study are:

- Employ outlier detection techniques to identify and remove outliers from the dataset.
- Utilize dimensionality reduction methods to reduce the number of features while maintaining prediction accuracy.
- Implement regression models to accurately forecast hydrogen consumption in refinery hydrotreating processes.

3 Methodology

The methodology employed in this study follows a comprehensive machine learning pipeline, as depicted in Figure 2. The initial step involves collecting and preprocessing the relevant data from the refinery's process historian database. Subsequently, various outlier detection techniques are explored and evaluated to identify and mitigate the impact of anomalous data points on the predictive models. Dimensionality reduction methods are then employed to reduce the number of features while retaining crucial information, thereby enhancing model explainability and computational efficiency. Finally, advanced regression models determined selected from the top performers using autoML, including ensemble techniques like Random Forest and Extra Trees, are implemented and optimized through hyperparameter tuning and cross-validation. The following subsections provide detailed insights into each stage of the methodology.

3.1 Data

The dataset was sourced from the Petron Bataan Refinery process historian database (PHD), which collects and stores time-series data from various points in the process plant. To simplify our study, we focused on a single hydrotreater, specifically the largest one in the refinery. We collected hourly average measurements from May 1, 2023, to March 15, 2024, resulting in 7,608 data points. This period was chosen by process engineers to include only the times when the unit was in normal operation. We deliberately limited the dataset to this period to fit within our available computational resources.

Significant features were identified in collaboration with the process engineer managing the unit. The features included the feed rates of different feed qualities entering the reactor, named **Feed_1**, **Feed_2**, **Feed_3**, **Feed_4**, **Feed_5**, and **Feed_6**. These feeds vary in quality: Feed_1, Feed_2, Feed_3, and Feed_6 are all straight run feeds, while Feed_4 and Feed_5 are cracked feeds. Straight run feeds are derived directly from the distillation process and generally have lower levels of impurities. In contrast, cracked feeds come from the catalytic or thermal cracking processes, resulting in higher levels of olefins and sulfur, leading to increased H_2 consumption.

Additionally, three process parameters were deemed significant and included in the study:

- **Sulfur_Product**: The sulfur concentration in the product. Process parameters are adjusted to maitain a maximum sulfur product content of 50 ppm.
- **Inlet_Temp**: The inlet temperature of the reactor, which is gradually increased over time to maintain catalyst activity.
- **Catalyst_Age**: The number of days since the last catalyst changeout.

3.2 Outlier Detection

We performed outlier detection to identify anomalous data points. We employed and studied three different outlier detection methods: Kernel Density Estimation (KDE), Local Outlier Factor (LOF), and Isolation Forest (Iso).

3.2.1 Kernel Density Estimation

Kernel density estimation involves estimating the probability density function of a dataset using kernel functions. This smooth out the distribution around each data point. This method is particularly effective in anomaly detection as it identifies instances with a low probability of occurrence under the estimated distribution which indicates they are anomalies or outliers.

3.2.2 Local Outlier Factor

The Local Outlier Factor (LOF) is a measure used to determine how much an individual sample deviates from the expected

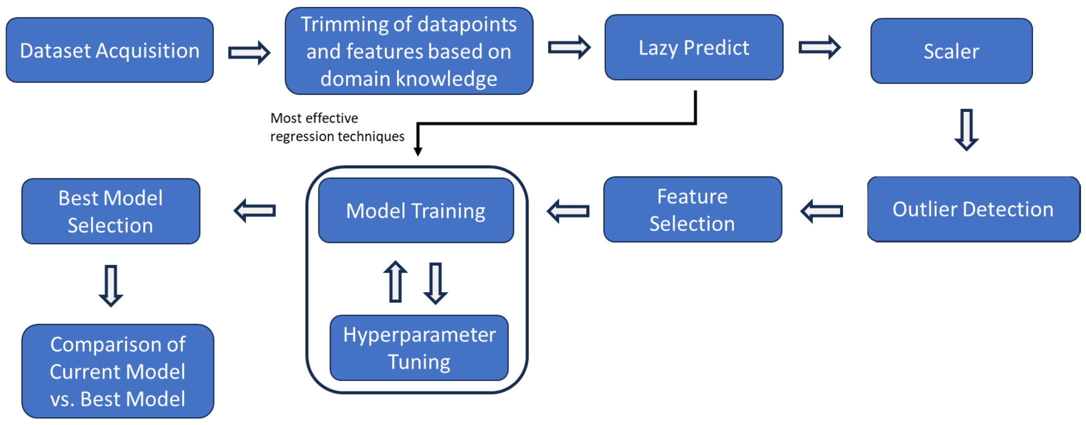
Dataset Acquisition
Trimming of datapoints and features based on domain knowledge
Lazy Predict
Scaler
Most effective regression techniques
Outlier Detection
Feature Selection
Model Training
Hyperparameter Tuning
Best Model Selection
Comparison of Current Model vs. Best Model

Figure 2: Machine Learning Pipeline

density based on its surrounding data points. This metric is ”local” because it relies on the density of neighboring samples. It uses the distance to the k-nearest neighbors to estimate this local density. By comparing a sample’s density to its neighbors, LOF identifies samples with significantly lower density as potential outliers.(Scikit-learn, 2024c)

3.2.3 Isolation Forest

Isolation Forest calculates anomaly scores for each sample. This algorithm ’isolates’ data points by randomly selecting features and split points within those features. Anomalies, which have shorter average path lengths in the resulting tree structures, are identified by the forest of random trees.(Scikit-learn, 2024a)

3.3 Machine Learning Models

We identified the best-performing machine learning models using autoML, specifically Lazy Predict. This tool automated the machine learning process, allowing us to build numerous basic models with minimal coding effort and quickly assess which models perform better without any parameter tuning. The top two models identified by Lazy Predict were Random Forest and Extremely Randomized Trees. These models were further evaluated and compared to the baseline model, which utilized only a linear regression approach.

3.3.1 Linear

Given a vector of input variables $X^T = (X_1, X_2, \ldots, X_p)$, the linear regression model predicts the output Y as:

$$\hat{Y} = \hat{\beta}_0 + \sum_{j=1}^{p} X_j \hat{\beta}_j \tag{1}$$

Here, $\hat{\beta}_0$ is the intercept term. This equation can be expressed using matrix notation:

$$\hat{Y} = X^T \hat{\beta}$$

where X^T is the transpose of X, $\hat{\beta}$ is the vector of coefficients, and $\hat{Y}$ are the predicted values.

The method of least squares minimizes the residual sum of squares (RSS):

$$\text{RSS}(\beta) = \sum_{i=1}^{n} \left(y_i - x_i^T \beta \right)^2 \tag{2}$$

where β are the coefficients. This is represented in matrix form as:

$$\text{RSS}(\beta) = (y - X\beta)^T (y - X\beta) \tag{3}$$

The solution $\hat{\beta}$ that minimizes RSS satisfies the normal equations:

$$X^T (y - X\beta) = 0 \tag{4}$$

If $X^T X$ is invertible, $\hat{\beta}$ is given by:

$$\hat{\beta} = (X^T X)^{-1} X^T y \tag{5}$$

We use $\hat{\beta}$ as estimators for the coefficients to compute the predicted values $\hat{Y}$. (Hastie et al., 2009)

3.3.2 Random Forest

A Random Forest is an ensemble learning method that utilizes multiple decision trees. Each tree in the forest is built using a random subset of the features (variables) from the input data. First, each tree is trained on an independent bootstrap sample drawn from the original data. This bootstrap sampling introduces randomness into each tree’s training process. Second, at each node of a tree, Random Forests select a random subset of features and find the best split within this subset.

Ensembles like Random Forests construct a prediction function $f(x)$ by combining several base learners $h_j(x), j = 1, \ldots, J$, where J is the number of trees in the forest. Each base learner h_j is an individual decision tree trained on different subsets of the training data and features. In regression tasks, the ensemble predictor $f(x)$ aggregates predictions by averaging the outputs of all trees:

$$f(x) = \frac{1}{J} \sum_{j=1}^{J} h_j(x) \tag{6}$$

Here, $f(x)$ represents the final predicted value. (Cutler et al., 2012)

3.3.3 Extremely randomized trees

The Extremely randomized trees or the Extra-Trees algorithm builds an ensemble of decision or regression trees with distinct characteristics compared to other tree-based ensemble methods. It differs in two main ways: first, it randomly selects cut-points to split nodes. Second, unlike methods that rely on bootstrap replicas, Extra-Trees uses the entire learning dataset to grow its trees. This approach aims to enhance diversity and robustness within the ensemble. Similar to Random Forests, predictions in Extra-Trees are made by averaging the outputs of all trees in the ensemble (Geurts et al., 2006)

3.4 Feature Selection Technique

We employed recursive feature elimination (RFE) for the feature selection process. Recursive feature elimination is a technique that recursively removes the least significant features based on the performance of a model built on the remaining features. By evaluating the importance of each feature, RFE aims to select the most relevant subset of features that contribute the most to the predictive power of the model.

RFE works by fitting a model and removing the weakest feature (or features) until the specified number of features is reached. The process involves training the model on the initial set of features, ranking all features based on their importance or the magnitude of their coefficients, removing the least important features, and repeating the process with the reduced set of features until the desired number of features is reached. (Scikit-learn, 2024b)

Initially, we considered 9 features based on the domain knowledge of the process engineers managing the unit. These features were selected due to their perceived importance and relevance to the process. Using RFE, we systematically evaluated the contribution of each feature to the model's performance. This involved iteratively removing the least significant features and assessing the impact on model accuracy.

By applying RFE, we aimed to achieve several goals. We sought to improve model performance by eliminating irrelevant or redundant features, which can enhance predictive accuracy and generalization to new data. Additionally, we aimed to reduce overfitting, as removing unnecessary features helps prevent the model from performing well on training data but poorly on unseen data. Enhancing interpretability was another goal, as a model with fewer features is easier to understand and use for decision-making. Finally, we aimed to increase computational efficiency, as fewer features reduce the computational cost and time required for training the model, making it more efficient to deploy and use.

We assessed whether the elimination of certain features would result in comparable performance to using all 9 features. This thorough evaluation ensured that the final model retained its predictive power while being more streamlined and efficient.

3.5 Performance Evaluation Metrics

The evaluation of our predictive models hinges on two fundamental metrics: root mean square error (RMSE) and coefficient of determination (R^2).

RMSE measures the average magnitude of the errors between predicted values and actual observed values in a regression model. The formula is given in equation 7.

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2} \tag{7}$$

where n is the total number of observations, y_i represents the actual value of the dependent variable at the i-th observation and $\hat{y}_i$ represents the predicted value of the dependent variable for the i-th observation.

On the other hand, R^2 quantifies the proportion of the variance in the dependent variable that is predictable from the independent variables. It ranges from 0 to 1, with higher values indicating a better fit of the model to the data.

$$R^2 = 1 - \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{n}(y_i - \bar{y})^2} \tag{8}$$

where n is the number of observations, y_i represents the actual value of the dependent variable at the i-th observation and $\hat{y}_i$ represents the predicted value of the dependent variable for the i-th observation, and $\bar{y}$ represents the mean of the observed values.

3.6 Cross-validation and Hyperparameter tuning

Optuna was employed in this paper. This is a state-of-the-art hyperparameter optimization framework introduced in (Akiba et al., 2019). It utilizes Bayesian Optimization techniques and is particularly well-suited for research on algorithm selection and hyperparameter tuning in the context of the CASH framework. This framework allows users to define their models and specify the hyperparameters to optimize (Akiba et al., 2019).

In this study, a 10-fold cross-validation approach was utilized. This method involves dividing the training data into 10 subsets of equal size. During each iteration of the cross-validation process, 9 subsets are allocated for training the model, while the remaining subset is used for validation.
This entire process is repeated 10 times, ensuring that each subset serves as the validation set exactly once. After completing the 10 iterations, the performance metrics of the model (RMSE) are averaged across all folds.
In this paper, we tuned hyperparameters including `n_estimators`, `max_depth`, `min_samples_leaf`, `min_samples_split`, and `ccp_alpha`. Their definitions of these hypermaters taken from (**?**) are given below:

- `n_estimators`: The number of trees in the forest.
- `max_depth`: The maximum depth of each tree in the forest.
- `min_samples_leaf`: The minimum number of samples required to be at a leaf node.
- `min_samples_split`: The minimum number of samples required to split an internal node.
- `ccp_alpha`: Complexity parameter used for Minimal Cost-Complexity Pruning. This parameter is used to control the complexity of trees in ensemble methods.

These hyperparameters were tuned to optimize the performance of the model.

4 Results and Discussions

4.1 Data Analysis

Figure 3 shows the correlation matrix heatmap of various features and H2 consumption. The target variable which is H2 consumption shows the high positive correlation with Feed_5, Inlet_Temp and Catalyst_Age. Moreover, it can be observed that Feed_5, Inlet_Temp and Catalyst_Age exhibit strong correlations with each other. This may lead to multicollinearity problems. Given these findings, one might consider using ensemble models over linear models to address the issue of multicollinearity.

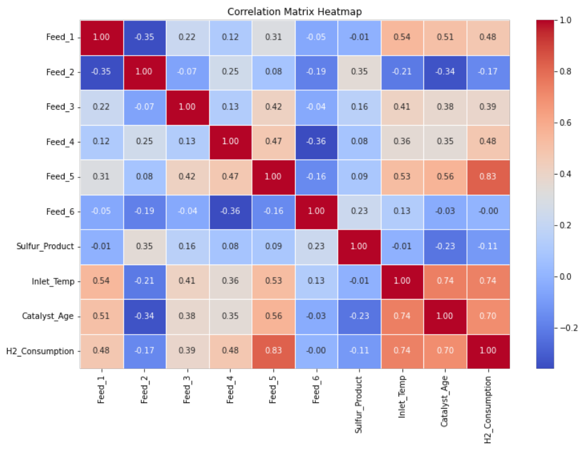

Figure 3: Feature-Target Correlation Heatmap

4.2 Model Selection

In selecting the most appropriate regression models for this study, we utilized Lazy Predict to comprehensively evaluate and compare the performance of several candidate models. Table 1 presents the RMSE of Test data obtained from this evaluation.

Among the models assessed, the Extra Trees Regressor and Random Forest Regressor emerged as standout performers with the lowest Test RMSE values of 329.39 and 381.5, respectively. These top-performing models have been chosen for integration into the study. Moreover, it can be observed that ensemble methods generally exhibited lower RMSE values compared to other regression models.

4.3 Outlier Detection Method

After selecting the regression models, the next step involves outlier detection to identify and mitigate any data points that could potentially impact model performance. It is important to note that domain knowledge is used in identifying and addressing outliers beforehand. But in this case, we conduct outlier detection for less obvious outliers which could potentially interfere with the creation of model.

These outliers may stem from extreme process values significantly deviating from the typical operating range, measurement errors, sensor malfunctions, or occasional spikes in process parameters. Removing these outliers is essential to enhance the quality and reliability of data used for analysis and modeling. This ensures that the models developed are more accurate and robust.

Our strategy incorporates three distinct methods: Kernel Density Estimation (KDE), Local Outlier Factor (LOF), and Isolation Forest (Iso). This approach is implemented on the test dataset using various contamination levels which is the percentage of outliers in the data set. RMSE values are obtained for each method.

Figure 4 illustrates the impact of different outlier removal

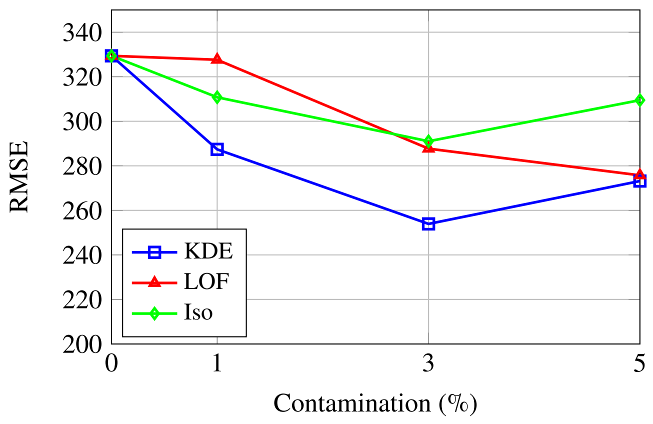

Figure 4: Scores of KDE, LOF, and Iso at different contamination levels

strategies on RMSE values of the test data across different contamination levels. KDE consistently shows the lowest RMSE compared to other methods and across varying contamination levels. However, it can be observe a decline in its effectiveness when the contamination value is at 5%. Therefore, the conclusion is reached by selecting a contamination value of 3%.

Table 2 presents the RMSE comparison of the train and test data for the two selected models under KDE and without any outlier removal. These models were trained using default hyperparameters. The results indicate that both models experience decreased RMSE values for both Train and Test Data when the KDE is applied. This observation suggests that KDE contributes to lowering the RMSE. Moreover, the results indicate that the removal of the outliers helped in preventing overfitting of the model as evidenced by narrower gap of train and test RMSEs of models with outliers removed as compared to those without.

4.4 Recursive Feature Elimination

The next step in the process involves Recursive Feature Elimination (RFE). This technique is used to identify a subset of features that contribute most significantly to our model's predictive power. The Elbow method is implemented in RFE by iteratively removing features and evaluating their impact on model performance.

Figure 5 illustrates the application of the elbow method across varying numbers of features and corresponding RMSE values for both the Extra Trees and Random Forest models.

Model	Test RMSE	Model	Test RMSE
1. **ExtraTreesRegressor**	**329.39**	11. AdaBoostRegressor	1,052.65
2. **RandomForestRegressor**	**381.5**	12. LassoCV	1,190.74
3. XGBRegressor	390.8	13. Lasso	1,190.76
4. LGBMRegressor	401.34	14. LassoLars	1,190.76
5. HistGradientBoostingRegressor	403.55	15. BayesianRidge	1,190.78
6. BaggingRegressor	405.23	16. Ridge	1,190.80
7. KNeighborsRegressor	436.97	17. RidgeCV	1,190.80
8. ExtraTreeRegressor	531.08	18. LassoLarsCV	1,190.81
9. DecisionTreeRegressor	532.28	19. LassoLarsIC	1,190.81
10. GradientBoostingRegressor	584.67	20. LinearRegression	1,190.81

Table 1: Test RMSE values for various regression models obtained from LazyPredict

Method	**Outlier Removal**	**Train RMSE**	**Test RMSE**
Extra Trees	None	430.2	488.6
	KDE	395.7	400.5
Random Forest	None	432.2	510.9
	KDE	385.4	390.9

Table 2: Comparison of RMSE values with and without KDE outlier removal

The figure suggests that retaining 2 features—Feed 5 and Catalyst Age—is optimal for our model. This streamlined feature set not only simplifies computational complexity and reduces training time but also ensures that it focuses on the most relevant and impactful features.

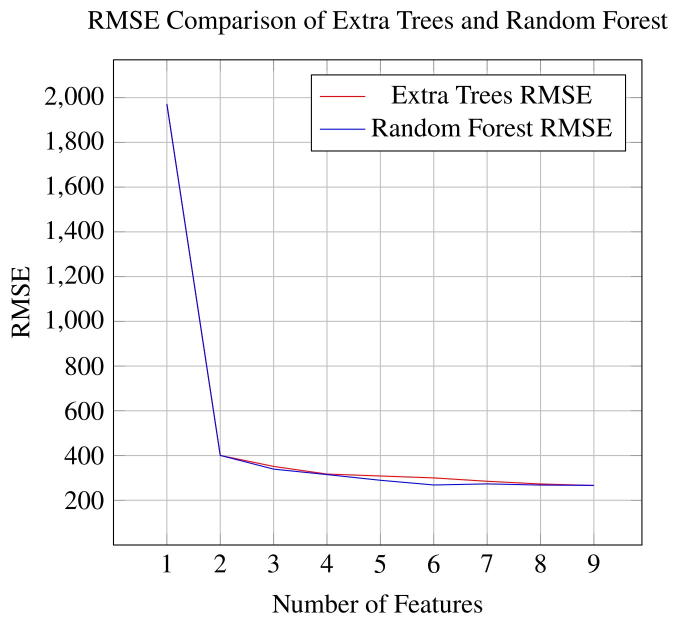

Figure 5: Comparison of RMSE values using Recursive Feature Elimination (RFE) for Extra Trees and Random Forest models

4.5 Hyperparameter tuning and model performance

After validating the impact of outlier detection on reducing the RMSE, the focus of the paper now shifted to the cross-validation and hyperparameter tuning. Optuna was employed to optimize model parameters, as detailed in Table 3. Additionally, utilizing 10-fold cross-validation, we tuned hyperparameters including n_estimators, max_depth, min_samples_leaf, min_samples_split, and ccp_alpha.

Hyperparameters	**Search**	**Random Forest**	**Extra Trees**
n_estimators	50 - 150	82	112
max_depth	15 - 25	20	15
min_samples_leaf	1 - 5	1	2
min_samples_split	2 - 10	5	5
ccp_alpha	0 - 0.2	0.055	0.007

Table 3: Best hyperparameters and search values for Random Forest and Extra Trees models.

The importance of these hyperparameters is illustrated in Figure 6. It is revealed that min_samples_leaf stands out as the most influential for both models.

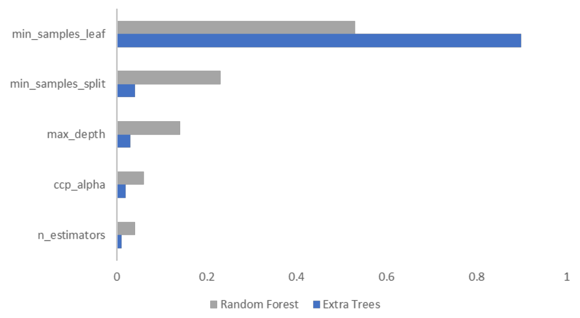

Figure 6: Hyperparameter Importance.

We compared the performance of two tuned regression models: Extra Trees and Random Forest. Table 4 presents the results in terms of their test R^2 scores, test RMSE values, and training times. The Extra Trees model achieved a higher Test R^2 and RMSE. Moreover, Extra Trees model showed lower training time compared to random forest. With these results, we can say the Extra Trees is the better model between the two.

Metric	**Extra Trees**	**Random Forest**
Test R^2	0.988	0.986
Test RMSE	356.2	385.8
Training Time	517.7 sec	1808.8 sec

Table 4: Comparison of Extra Trees and Random Forest models

4.6 Explainability Analysis

Lastly, we performed an explainability analysis among the remaining features. Figure 7 shows the beeswarm plot of the data points of the features. We can see that the feature Feed_5 exhibits a significant impact on the target variable which is the hydrogen consumption. High values of Feed_5 (indicated by red points) are associated with an increase in hydrogen consumption, as shown by the positive SHAP values. Conversely, low values of Feed_5 (blue points) tend to decrease hydrogen consumption, as reflected by the negative SHAP values. This wide distribution of SHAP values for Feed_5 indicates its substantial influence on the model's predictions. In addition, Catalyst_Age is also impactful. Higher values of Catalyst_Age slightly increase hydrogen consumption, whereas lower values tend to decrease it.

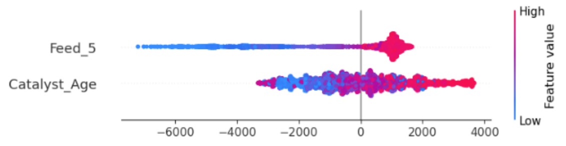

Figure 7: SHAP Beeswarm Plot Showing Feature Importance and Impact on Hydrogen Consumption.

4.7 Model Comparison

Since we have already finalized the model, we now proceed to compare it with the existing models that the company is using. This comparison is important to evaluate whether our trained model has effectively enhanced the current deployment.

Table 5 presents a comparative analysis of various models used for predicting H2 consumption using performance metrics such as RMSE (Root Mean Square Error) and R^2 (coefficient of determination).

The performance of the current model serves as our research baseline, representing the company's existing linear regression model, which shows only moderate predictive accuracy. Currently, the company utilizes a linear regression model based solely on feed streams entering the unit.

By enhancing this linear regression model with additional critical features such as `Inlet_Temp`, `Sulfur_Product`, and `Catalyst_Age`, significant performance improvements were achieved. Specifically, this enhancement reduced the RMSE from 2194 to 1398 and increased the R^2 from 0.58 to 0.82.

However, the most substantial improvement is observed in the tuned Extra Trees Regressor developed in this research, which further reduces the RMSE to 356 and increases the R^2 to 0.99. This indicates that the ensemble learning method used in the Extra Trees Regressor, along with hyperparameter tuning, provides superior performance compared to traditional linear regression models. Therefore, the Tuned Extra Trees Regressor is identified as the best model.

5 Conclusion

In this paper, we combined outlier detection, dimensionality reduction, and advanced regression modeling to significantly improve the accuracy and robustness of predictive models.

Model Description	RMSE	R^2
Current Model (Linear Regression)	2194	0.58
Refitted Current Model using new data and additional Critical Features	1398	0.82
Tuned ExtraTrees Regressor	356	0.99

Table 5: Comparison of RMSE and R^2 for Different Models

By implementing outlier detection, we removed anomalous data points. This results in a more reliable models with reduced overfitting. This has been demonstrated by having lower training and test RMSE compared to the model trained without outlier detection. Dimensionality reduction further streamlined the models. This reduces the number of features from 9 to 2 while maintaining high accuracy. This simplification facilitated more efficient and interpretable models.

The use of advanced regression models, particularly the Tuned Extra Trees Regressor, led to a dramatic decrease in RMSE from an initial 2194 to 356. This demonstrates a significant boost in predictive capability.

By making use of only 2 features, particularly `Feed_5` and `Catalyst_Age`, predicting the H2 consumption of the hydrotreater becomes more straightforward. This approach eliminates the need to account for all feed streams entering the hydrotreater, each with different yields. Instead, focusing on `Feed_5` and `Catalyst_Age` simplifies the determination of H2 consumption, leveraging their significant impacts identified through our modeling process.

Overall, the application of these techniques not only enhanced the precision of predictions but also streamlined the modeling process, making it more efficient and interpretable for practical applications in refinery operations.

6 Recommendation

The current study focuses on ensemble models based on LazyPredict, leaving unexplored the potential of other machine learning techniques for this task. It is recommended that future research investigate the efficacy of alternative regression models, such as neural networks and support vector machines (SVMs), for this specific problem. Neural networks and SVMs have demonstrated remarkable performance in various regression tasks, and with appropriate tuning, they could potentially outperform the ensemble models explored in this study.

Additionally, future studies should explore the application of these machine learning techniques to other hydrotreaters within the refinery context. This broader application could enhance the accuracy of hydrogen balances by integrating predictive models with real-time operational data from multiple units.

References

Takuya Akiba, Shotaro Sano, Takeru Yanase, Takeru Ohta, and Masanori Koyama. 2019. Optuna. In *Proceedings* *of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*.

L. C. Castañeda, J. A. D. Muñoz, and J. Ancheyta. 2011. Comparison of approaches to determine hydrogen consumption during catalytic hydrotreating of oil fractions. *Fuel*, 90(12):3593–3601.

Adele Cutler, D. Richard Cutler, and John R. Stevens. 2012. Random forests. In *Ensemble Machine Learning*, pages 157–175. Springer.

J. Ferreira, M. Pedemonte, and A. I. Torres. 2022. Development of a machine learning-based soft sensor for an oil refinery's distillation column. *Computers & Chemical Engineering*, 161:107756.

Pierre Geurts, Damien Ernst, and Louis Wehenkel. 2006. Extremely randomized trees. *Machine Learning*, 63(1):3–42.

Trevor Hastie, Robert Tibshirani, and Jerome H. Friedman. 2009. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. Springer, New York, 2nd edition.

P. Nkulikiyinka, Y. Yan, F. Güleç, V. Manovic, and P. T. Clough. 2020. Prediction of sorption enhanced steam methane reforming products from machine learning based soft-sensor models. *Energy and AI*, 2:100037.

A. A. R. Pinos, S. Badoga, A. K. Dalai, and J. Adjaye. 2019. Modelling of h2 consumption and process optimization for hydrotreating of light gas oils. *The Canadian Journal of Chemical Engineering*, 97(6):1828–1837.

Scikit-learn. 2024a. sklearn.ensemble.IsolationForest. `https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.IsolationForest.html`. [Online; accessed 18 June 2024].

Scikit-learn. 2024b. sklearn.feature_selection.RFE. `https://scikit-learn.org/stable/modules/generated/sklearn.feature_selection.RFE.html`. [Online; accessed 18 June 2024].

Scikit-learn. 2024c. sklearn.neighbors.LocalOutlierFactor. `https://scikit-learn.org/stable/modules/generated/sklearn.neighbors.LocalOutlierFactor.html`. [Online; accessed 18 June 2024].

W. Yan, D. Tang, and Y. Lin. 2017. A data-driven soft sensor modeling method based on deep learning and its application. *IEEE Transactions on Industrial Electronics*, 64(5):4237–4245.