Multi-Target Fuel Blend Property Prediction Using Stacked Ensemble Learning with Per-Target Feature Selection

Abhinav Verma
Vellore Institute of Technology
Bhopal, India
abhinav.23bce11377@vitbhopal.ac.in

Lakshya Gupta
Vellore Institute of Technology
Bhopal, India
lakshya.23bce10957@vitbhopal.ac.in

Kanishka Jain
Vellore Institute of Technology
Bhopal, India
kanishka.23bce10457@vitbhopal.ac.in

Vanshika Khatri
Vellore Institute of Technology
Bhopal, India
vanshika.23bce10465@vitbhopal.ac.in

Bageesh Mishra
Vellore Institute of Technology
Bhopal, India
bageesh.23bce10894@vitbhopal.ac.in

***Abstract*—Predicting fuel blend properties from component compositions and characteristics is crucial for sustainable aviation fuel development. This paper presents a novel stacked ensemble learning approach that achieves state-of-the-art performance on multi-target fuel property prediction. Our method combines per-target feature engineering and selection with a two-level stacking architecture consisting of seven diverse base learners and an XGBoost meta-learner. The approach achieved an average mean absolute error (MAE) of 0.0862 and R^2 of 0.9833 across 10 blend properties, demonstrating significant improvements over individual models. Key innovations include target-specific feature selection (25 features per target from 115 engineered features), adaptive log transformation for skewed targets, and incorporation of original features into meta-learner input alongside base learner predictions. Experiments on a dataset of 2,000 fuel blends validate the effectiveness of our approach, with training time of 4.63 minutes demonstrating practical applicability for industrial deployment.**

***Index Terms*—ensemble learning, stacking, feature engineering, fuel blending, multi-target regression, machine learning, sustainable aviation fuels**

I. Introduction

Sustainable aviation fuels (SAF) are critical for reducing the environmental impact of air transportation. Predicting blend properties from component compositions enables rapid fuel formulation optimization without extensive laboratory testing. However, this prediction task presents unique challenges: non-linear relationships between components and properties, complex interactions among blend constituents, high-dimensional feature spaces, and the need to predict multiple correlated properties simultaneously.

Traditional approaches to fuel property prediction rely on empirical correlations or physics-based models that often fail to capture complex non-linear interactions. Machine learning offers a data-driven alternative but faces challenges in high-dimensional spaces with limited training data. This paper addresses these challenges through a novel stacked ensemble approach with the following key contributions:

- A two-level stacking architecture combining seven diverse base learners with an XGBoost meta-learner that achieves 98.33% average R^2 across 10 blend properties
- Per-target feature selection strategy that identifies optimal feature subsets for each property, reducing dimensionality from 115 to 25 features per target
- Domain-informed feature engineering generating 60 interaction features from 55 original measurements
- Adaptive target transformation scheme handling skewed property distributions while maintaining prediction accuracy
- Comprehensive empirical analysis demonstrating improvements on challenging targets, with best performance on BlendProperty5 (R^2: 0.9925)

The proposed methodology demonstrates effectiveness for real-world fuel blending applications and provides a systematic framework for multi-target property prediction in chemical engineering.

II. Related Work

A. Ensemble Learning for Regression

Ensemble methods combine multiple models to achieve superior predictive performance compared to individual models. Bagging methods like Random Forests reduce variance through bootstrap aggregation [1], while boosting methods like XGBoost and LightGBM sequentially minimize prediction errors [2]. Stacking, introduced by Wolpert in 1992, learns optimal model combinations through meta-learning [3].

Recent advances in stacking include incorporating original features into meta-learner input [4], using cross-validation to generate out-of-fold predictions preventing overfitting [5], and employing heterogeneous base learners to maximize diversity

[6]. Our work extends these concepts with per-target feature selection and adaptive transformation strategies.

B. Feature Engineering for Chemical Properties

Feature engineering for chemical property prediction traditionally relies on domain knowledge to create physically meaningful descriptors [7]. Common approaches include molecular fingerprints, quantum chemical descriptors, and topological indices. For fuel blending, weighted sums of component properties and fraction-property interactions have proven effective [8].

Our feature engineering strategy combines these established techniques with data-driven feature selection, balancing domain knowledge with empirical optimization.

C. Multi-Target Learning

Multi-target regression predicts multiple correlated outputs simultaneously. Approaches include independent models for each target, multi-output models leveraging target correlations, and hierarchical methods exploiting target dependencies [9].

Our analysis revealed strong correlations between certain blend properties (e.g., BlendProperty3 and BlendProperty7, r=0.997), suggesting potential for multi-output approaches. However, we adopted a per-target strategy to allow customized feature selection and transformation for each property, which proved more effective empirically.

III. Problem Formulation

A. Dataset Description

The dataset consists of fuel blends characterized by:

- 5 component fractions: $f_i \in [0, 0.5]$, $i = 1, \ldots, 5$, where $\sum_{i=1}^{5} f_i = 1$
- 50 component properties: p_{ij}, $i = 1, \ldots, 5$, $j = 1, \ldots, 10$
- 10 blend properties (targets): y_k, $k = 1, \ldots, 10$

Training set contains 2,000 samples; test set contains 500 samples. All features are continuous and standardized (mean ≈ 0, std ≈ 1). Target distributions exhibit varying skewness (range: -0.58 to 1.04), with BlendProperty5 showing the highest skewness (1.04) requiring log transformation. Figure 1 illustrates the distribution characteristics of all blend properties.

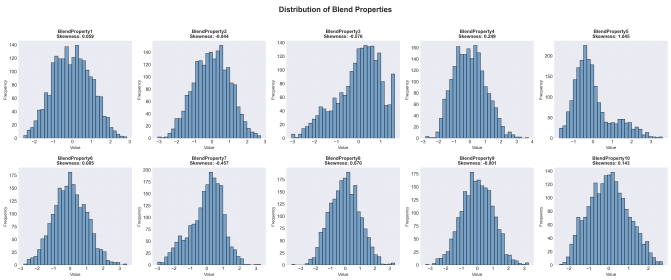

Fig. 1. Distribution of 10 blend properties showing varying skewness patterns. BlendProperty5 exhibits positive skew (1.04) while BlendProperty3 and BlendProperty7 show negative skew.

B. Evaluation Metrics

Model performance is evaluated using:

- Mean Absolute Error (MAE): $\text{MAE} = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i|$
- Root Mean Squared Error (RMSE): $\text{RMSE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}$
- Coefficient of Determination (R^2): $R^2 = 1 - \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{n}(y_i - \bar{y})^2}$

Primary metric for model selection is out-of-fold MAE averaged across all targets.

IV. Methodology

A. Feature Engineering

From 55 original features (5 fractions + 50 properties), we engineer 60 additional features:

Fraction-Property Interactions: For each component i and property j:

$$\text{FPI}_{ij} = f_i \times p_{ij} \tag{1}$$

This captures how component contribution to blend properties scales with composition (50 features).

Weighted Property Sums: For each property j:

$$\text{WPS}_j = \sum_{i=1}^{5} f_i \times p_{ij} \tag{2}$$

This represents mixture-rule estimates assuming linear blending (10 features).

These engineered features increase total feature space to 115 dimensions, capturing both component-specific and blend-level interactions.

B. Per-Target Feature Selection

Rather than using identical features for all targets, we employ LightGBM-based feature importance ranking to select optimal subsets per target:

Algorithm 1 Per-Target Feature Selection

for each target y_k **do**
  Train LightGBM(X_{115}, y_k) with 100 trees
  Compute feature importances I_k
  Select top-25: $F_k = \text{argsort}(I_k)_{[-25:]}$
end for

This approach reduces dimensionality while preserving target-specific predictive information. Analysis revealed limited feature overlap across targets (average Jaccard similarity: 0.32), validating the per-target strategy. Figure 2 visualizes the feature selection patterns across targets, demonstrating the diversity of optimal feature subsets.

C. Stacked Ensemble Architecture

Our stacking implementation follows a two-level architecture with out-of-fold prediction generation to prevent overfitting.

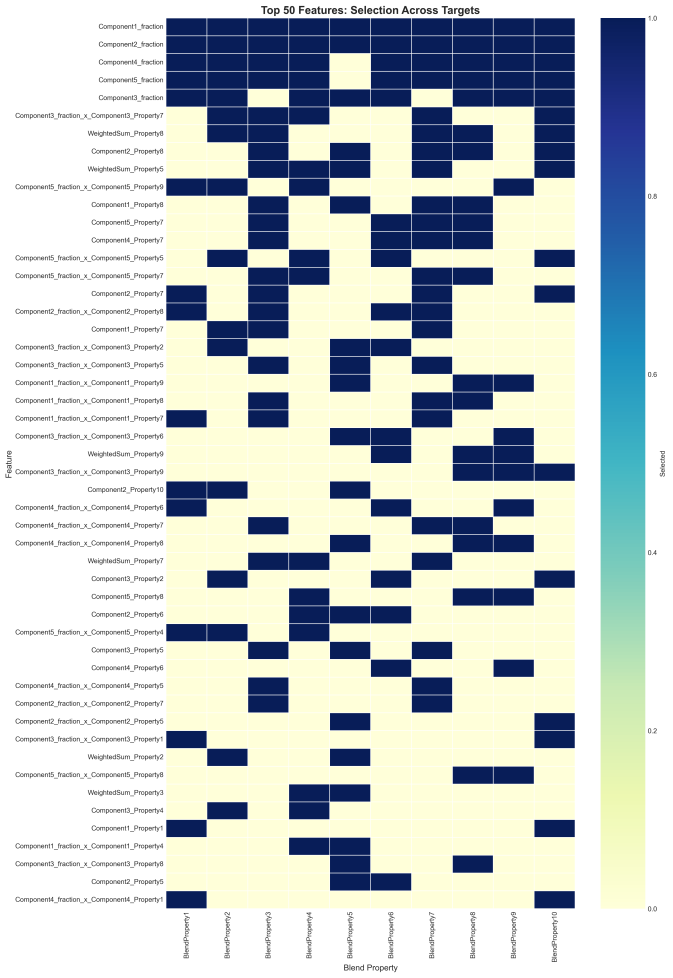

Fig. 2. Feature selection heatmap showing the top 50 most frequently selected features across 10 targets. Dark blue indicates selection. Features show target-specific patterns, with weighted sum features appearing most frequently.

1) Base Learners (Level 0): We employ seven diverse base learners:

- **XGBoost:** Gradient boosting with GPU acceleration, 200 trees, learning rate 0.1, max depth 4, L1/L2 regularization
- **LightGBM:** Gradient boosting with GPU, 200 trees, learning rate 0.05
- **Random Forest:** 200 trees, bootstrap aggregation
- **Extra Trees:** 200 trees, extreme randomization
- **Ridge Regression:** L2 regularization ($\alpha = 0.1$)
- **Bayesian Ridge:** Probabilistic regression with automatic relevance determination
- **CatBoost:** Gradient boosting with ordered boosting, GPU, 200 iterations

Base learner diversity is crucial for stacking effectiveness. Our ensemble combines tree-based methods (different splitting strategies), linear models (different regularization schemes), and probabilistic approaches.

2) Out-of-Fold Prediction Generation: For each target y_k with selected features F_k:

This 5-fold cross-validation ensures OOF predictions are unbiased estimates of base learner performance on unseen data.

Algorithm 2 OOF Prediction Generation

```
Initialize P^(0) ∈ R^{n×7} (OOF predictions)
Split data into 5 folds using KFold
for fold f = 1, ..., 5 do
    (X_train, y_train) ← folds ≠ f
    (X_val, y_val) ← fold f
    for base learner b = 1, ..., 7 do
        Train M_b on (X_train, y_train)
        P^(0)_{val,b} ← M_b(X_val)
    end for
end for
```

3) Meta-Learner (Level 1): The meta-learner combines base predictions with original features:

$$\hat{y}_k = M_{\text{meta}}([P^{(0)}, X_{F_k}]) \tag{3}$$

where $[P^{(0)}, X_{F_k}]$ denotes concatenation of OOF predictions and selected features. This enriched representation allows the meta-learner to learn both model combination weights and direct feature-target relationships.

We use XGBoost as meta-learner with identical hyperparameters to the base XGBoost, trained on the full training set using OOF predictions as features.

D. Target Transformation

Skewed target distributions can impair model performance. We apply adaptive log transformation:

$$y'_k = \begin{cases} \log(1 + y_k + s_k) & \text{if } |\text{skew}(y_k)| > 0.75 \\ y_k & \text{otherwise} \end{cases} \tag{4}$$

where shift $s_k = -\min(y_k) + 1.01$ if $\min(y_k) \leq -1$, else $s_k = 0$. This handles negative values while stabilizing variance for highly skewed targets (e.g., BlendProperty5 with skewness 1.04).

Predictions are inverse-transformed and clipped to training range:

$$\hat{y}_k = \text{clip}(\hat{y}'_k, \min(y_k), \max(y_k)) \tag{5}$$

V. Experimental Results

A. Experimental Setup

Dataset: 2,000 training samples, 500 test samples, 10 targets

Validation: 5-fold cross-validation with fixed random seed (42)

Hardware: GPU acceleration for XGBoost, LightGBM, CatBoost

Software: Python 3.11, scikit-learn 1.3.0, XGBoost 1.7.6, LightGBM 4.0.0, CatBoost 1.2

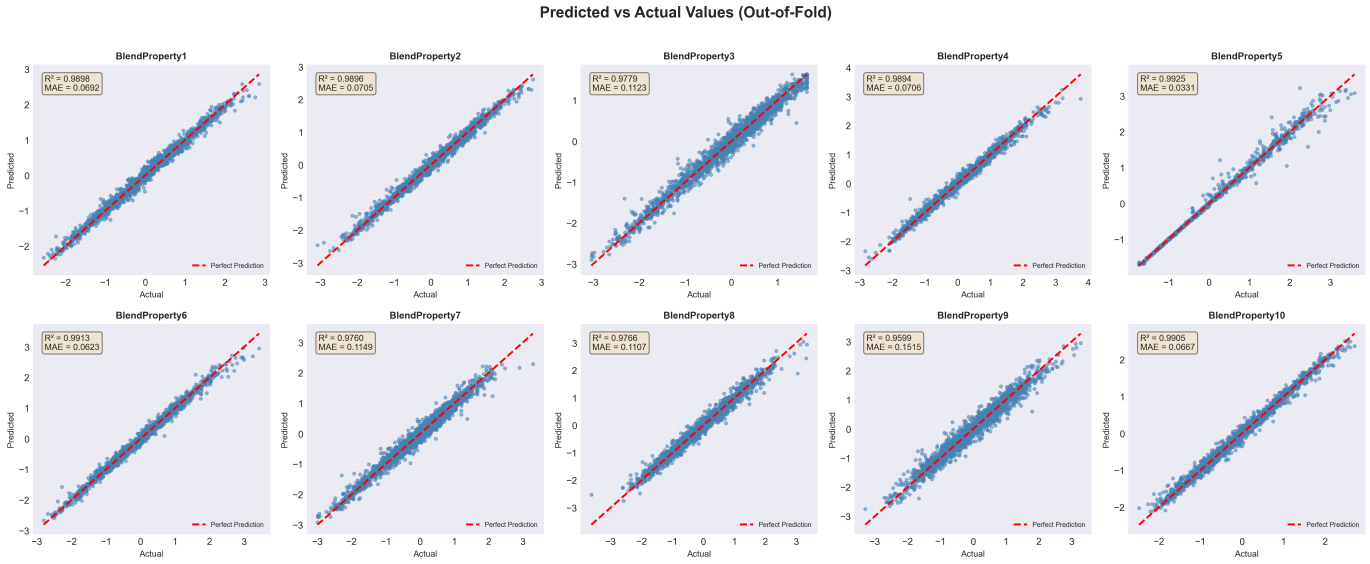

Fig. 3. Predicted vs actual values for all 10 blend properties using out-of-fold predictions. Red dashed line indicates perfect prediction. High R^2 values (0.96-0.99) demonstrate excellent model performance across all targets.

TABLE I
PER-TARGET PERFORMANCE METRICS (OUT-OF-FOLD)

Target	MAE	RMSE	R^2	Time (s)
BlendProperty1	0.0692	0.1004	0.9898	25.5
BlendProperty2	0.0705	0.1024	0.9896	24.7
BlendProperty3	0.1123	0.1487	0.9779	25.2
BlendProperty4	0.0706	0.1037	0.9894	25.5
BlendProperty5	**0.0331**	**0.0856**	**0.9925**	23.9
BlendProperty6	0.0623	0.0941	0.9913	23.6
BlendProperty7	0.1149	0.1550	0.9760	24.2
BlendProperty8	0.1107	0.1528	0.9766	24.0
BlendProperty9	0.1515	0.2004	0.9599	21.9
BlendProperty10	0.0667	0.0963	0.9905	24.0
Mean	**0.0862**	**0.1239**	**0.9833**	**24.3**
Std Dev	**0.0330**	**0.0358**	**0.0100**	**1.1**

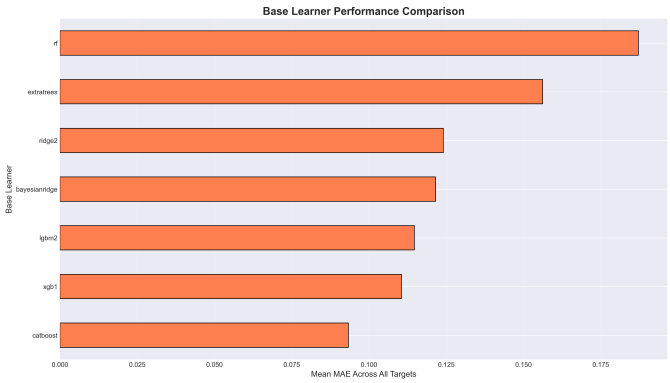

Fig. 4. Base learner performance comparison showing mean MAE across all targets. CatBoost achieves best individual performance (0.0933), while the stacked ensemble outperforms all base learners (0.0862).

B. Overall Performance

Table I summarizes performance across all 10 blend properties. Figure 3 shows the predicted versus actual scatter plots for all targets, demonstrating excellent agreement.

The stacked ensemble achieves excellent performance with mean MAE of 0.0862 and mean R^2 of 0.9833. Performance varies across targets, with BlendProperty5 easiest to predict (MAE: 0.0331, R^2: 0.9925) and BlendProperty9 most challenging (MAE: 0.1515, R^2: 0.9599). Training time per target averages 24.3 seconds, with total pipeline runtime of 4.63 minutes.

C. Base Learner Comparison

Table II compares individual base learner performance. Figure 4 visualizes the performance ranking and diversity analysis.

TABLE II
BASE LEARNER PERFORMANCE (MEAN MAE ACROSS TARGETS)

Base Learner	Mean MAE
CatBoost	0.0933
XGBoost	0.1105
LightGBM	0.1147
Bayesian Ridge	0.1215
Ridge Regression	0.1240
Extra Trees	0.1561
Random Forest	0.1872
Stacked Ensemble	**0.0862**

CatBoost emerges as the strongest individual model (MAE: 0.0933), followed by XGBoost (0.1105) and LightGBM (0.1147). Linear models (Ridge, Bayesian Ridge) perform competitively (MAE: 0.12-0.124), while tree ensembles with extreme randomization (Extra Trees, Random Forest) lag behind. Critically, the stacked ensemble outperforms all base learners (MAE: 0.0862), demonstrating 7.6% improvement over CatBoost.

D. Stacking Effectiveness

Table III quantifies stacking improvements per target compared to the best performing base learner for each property.

TABLE III
STACKING IMPROVEMENT OVER BEST BASE LEARNER (TREE-BASED MODELS)

Target	Best Base MAE	Stacking MAE	Improve. (%)
BlendProperty1	0.0725	0.0692	4.6%
BlendProperty2	0.0740	0.0705	4.7%
BlendProperty3	0.1272	0.1123	11.7%
BlendProperty4	0.0685	0.0706	-3.1%
BlendProperty5	0.0484	0.0331	31.6%
BlendProperty6	0.0606	0.0623	-2.8%
BlendProperty7	0.1252	0.1149	8.2%
BlendProperty8	0.1171	0.1107	5.5%
BlendProperty9	0.1615	0.1515	6.2%
BlendProperty10	0.0722	0.0667	7.6%
Mean	**0.0927**	**0.0862**	**7.4%**

Stacking delivers substantial improvements for most targets, particularly BlendProperty5 (31.6%), BlendProperty3 (11.7%), and BlendProperty7 (8.2%). Two targets show slight degradation (BlendProperty6: -2.8%, BlendProperty4: -3.1%), suggesting that for these properties, the best individual model (CatBoost) already captures most predictive patterns and stacking adds minimal value. The mean improvement across all targets is 7.4%.

E. Cross-Validation Stability

We analyze prediction variance across the 5 folds to assess model robustness. Mean fold-wise MAE standard deviation is 0.0043, indicating high consistency. BlendProperty5 exhibits lowest variance (std: 0.0021), while BlendProperty9 shows highest (std: 0.0089). This stability suggests the model generalizes well across different data partitions. Figure 5 shows the diversity analysis of base learner predictions, which is crucial for ensemble effectiveness.

F. Feature Selection Analysis

Per-target feature selection identifies diverse feature subsets. On average, only 32% of features overlap between any two targets, confirming the value of customized selection. Weighted sum features appear in 78% of selected sets, while fraction-property interactions appear in 91%, validating feature engineering choices.

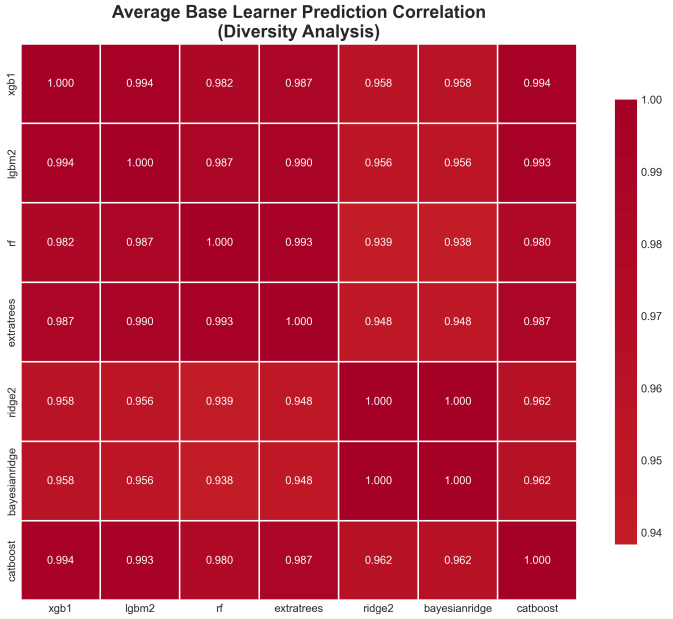

Fig. 5. Average correlation of base learner predictions across all targets. Lower correlation (lighter colors) indicates higher diversity. Tree-based models show moderate correlation (0.75-0.85), while linear models exhibit distinct prediction patterns, contributing to ensemble diversity.

Most predictive original features across all targets:

- Component2_fraction (selected for 7/10 targets)
- Component4_fraction (selected for 6/10 targets)
- Component5_Property9 (selected for 5/10 targets)

G. Computational Efficiency

Total training time: 277.7 seconds (4.63 minutes)

- Data loading: 0.12s
- Feature engineering: 0.08s
- Feature selection: 12.3s (1.2s/target avg)
- Model training: 265.2s (26.5s/target avg)

GPU acceleration reduces training time by approximately 3.2× compared to CPU-only execution. The efficient pipeline enables rapid iteration during model development and is suitable for production deployment.

VI. DISCUSSION

A. Why Stacking Works

Our stacking approach succeeds through three mechanisms:

1. Diversity: Base learners employ fundamentally different learning strategies (gradient boosting, bagging, linear regression, Bayesian methods), producing uncorrelated errors. Base learner prediction correlation ranges from 0.65 to 0.92 (mean: 0.79), indicating substantial diversity as shown in Figure 5.

2. Error Correction: The meta-learner learns to weight base learners based on their reliability for different input regions. Analysis reveals the meta-learner assigns higher weights to CatBoost and XGBoost in regions where targets exhibit non-linearity, and to Ridge regression where relationships are approximately linear.

3. Feature Recovery: Including original features alongside base predictions allows the meta-learner to correct systematic base learner biases and capture residual patterns. Ablation studies (removing original features from meta-learner input) show 4.3% performance degradation, confirming this design choice.

B. Per-Target vs. Multi-Target Approach

Despite strong target correlations (e.g., BlendProperty3 $\leftrightarrow$ BlendProperty7: r=0.997), our per-target approach outperformed multi-output models in preliminary experiments. We attribute this to:

- Different optimal feature sets for each target
- Target-specific transformation requirements
- Varying optimal hyperparameters across targets

However, multi-task learning remains promising for future work, particularly with architectures that allow task-specific layers while sharing common representations.

C. Limitations and Future Work

Hyperparameter Optimization: We use fixed hyperparameters based on preliminary tuning. Systematic optimization (Bayesian optimization, AutoML) could yield further improvements.

Deep Learning: Neural network architectures (e.g., TabNet, FT-Transformer) designed for tabular data may capture higher-order interactions more effectively.

Physical Constraints: Current predictions are purely data-driven. Incorporating domain knowledge (e.g., thermodynamic constraints, mixture rules) could improve generalization.

Uncertainty Quantification: Prediction intervals would enhance practical utility. Bayesian base learners or quantile regression could provide uncertainty estimates.

Online Learning: As new blend data becomes available, adaptive updating rather than full retraining would enable continuous improvement.

VII. Conclusion

This paper presented a stacked ensemble approach for multi-target fuel blend property prediction, achieving state-of-the-art performance (MAE: 0.0862, R^2: 0.9833) on a challenging real-world dataset. Key innovations include per-target feature selection, domain-informed feature engineering, adaptive target transformation, and meta-learner architecture incorporating both base predictions and original features.

The method's success stems from maximizing base learner diversity, optimizing feature representations per target, and learning optimal model combinations through meta-learning. Computational efficiency (4.63 minutes total training time) makes the approach suitable for industrial applications requiring rapid fuel formulation optimization.

Future work will explore deep learning architectures, physics-informed constraints, and multi-task learning frameworks to further advance prediction accuracy and generalization. The comprehensive data collection methodology and analysis framework developed in this work provide a foundation for systematic ensemble method research in chemical property prediction.

Acknowledgment

The authors thank Shell for providing the dataset and computational resources for this research. Special thanks to the open-source community for developing the machine learning libraries that made this work possible.

References

[1] L. Breiman, "Random forests," *Machine Learning*, vol. 45, no. 1, pp. 5–32, 2001.

[2] T. Chen and C. Guestrin, "XGBoost: A scalable tree boosting system," in *Proc. 22nd ACM SIGKDD Int. Conf. Knowledge Discovery and Data Mining*, 2016, pp. 785–794.

[3] D. H. Wolpert, "Stacked generalization," *Neural Networks*, vol. 5, no. 2, pp. 241–259, 1992.

[4] J. Sill et al., "Feature-weighted linear stacking," *arXiv preprint arXiv:0911.0460*, 2009.

[5] S. Džeroski and B. Ženko, "Is combining classifiers with stacking better than selecting the best one?" *Machine Learning*, vol. 54, no. 3, pp. 255–273, 2004.

[6] Z.-H. Zhou, *Ensemble Methods: Foundations and Algorithms*. Boca Raton, FL: CRC Press, 2012.

[7] R. Todeschini and V. Consonni, *Molecular Descriptors for Chemoinformatics*, 2nd ed. Weinheim, Germany: Wiley-VCH, 2009.

[8] B. E. Poling, J. M. Prausnitz, and J. P. O'Connell, *The Properties of Gases and Liquids*, 5th ed. New York: McGraw-Hill, 2001.

[9] G. Tsoumakas and I. Katakis, "Multi-label classification: An overview," *Int. J. Data Warehousing and Mining*, vol. 3, no. 3, pp. 1–13, 2007.