J. Iron Steel Res. Int.
https://doi.org/10.1007/s42243-025-01454-z

ORIGINAL PAPER

Boosting algorithms for predicting end-point temperature in BOF steelmaking using big industrial datasets

Jian-bo Zhang[1,2] · Maryam Khaksar Ghalati[1] · Jun Fu[1,2] · Xiao-an Yang[1,2] · G.M.A.M. El-Fallah[1] · Hong-biao Dong[1]

Received: 9 May 2024 / Revised: 4 June 2024 / Accepted: 11 June 2024

Abstract
The application of machine learning was investigated for predicting end-point temperature in the basic oxygen furnace steelmaking process, addressing gaps in the field, particularly large-scale dataset sizes and the underutilization of boosting algorithms. Utilizing a substantial dataset containing over 20,000 heats, significantly bigger than those in previous studies, a comprehensive evaluation of five advanced machine learning models was conducted. These include four ensemble learning algorithms: XGBoost, LightGBM, CatBoost (three boosting algorithms), along with random forest (a bagging algorithm), as well as a neural network model, namely the multilayer perceptron. Our comparative analysis reveals that Bayesian-optimized boosting models demonstrate exceptional robustness and accuracy, achieving the highest *R*-squared values, the lowest root mean square error, and lowest mean absolute error, along with the best hit ratio. CatBoost exhibited superior performance, with its test *R*-squared improving by 4.2% compared to that of the random forest and by 0.8% compared to that of the multilayer perceptron. This highlights the efficacy of boosting algorithms in refining complex industrial processes. Additionally, our investigation into the impact of varying dataset sizes, ranging from 500 to 20,000 heats, on model accuracy underscores the importance of leveraging larger-scale datasets to improve the accuracy and stability of predictive models.

Keywords Steelmaking · Basic oxygen furnace · Machine learning · Boosting algorithm

1 Introduction

The industrial sector has become a prodigious producer of data, a trend that has been further increased by the fourth industrial revolution, with the steel industry playing an integral role in this movement [1]. Industry 4.0 represents an ecosystem with diverse information interfaces that connect people, processes, administrations, systems, and internet of things-enabled industrial assets across both digital and physical realms. It provides a comprehensive vision of the industry's future, characterized by detailed digital processes where artificial intelligence (AI) contributes significantly [2]. Consequently, data-driven modeling and machine learning have thus risen for optimizing industrial processes, leveraging the extensive data generated during manufacturing [3, 4]. Nenchev et al. [5] provide a thorough illustration on data side, outlining the six essential stages of steel production. These stages begin from the blast furnace, followed by the basic oxygen furnace (BOF), ladle furnace, continuous casting, rolling machine, and finally the end products. Each stage is characterized by the integration of inputs, control variables, and external factors, which are essential for developing a data-driven optimization model.

Currently, basic oxygen furnaces play a crucial role in global steel production, contributing to more than 70% [6]. This phase is particularly a key phase as it directly influences the quality and productivity of the end products. The main objectives of the BOF are to elevate temperature and

✉ Maryam Khaksar Ghalati
mkg18@leicester.ac.uk

✉ Hong-biao Dong
hd38@leicester.ac.uk

[1] School of Engineering, University of Leicester, Leicester LE1 7RH, UK

[2] Nanjing Iron and Steel United Co., Ltd., Nanjing 210035, Jiangsu, China

Published online: 07 April 2025

control the composition during the steel transformation. To optimize and control the BOF process effectively, it is essential to accurately predict end-point temperature [7–9] and compositions, including elements such as carbon [10, 11], phosphorus [12, 13], and oxygen [14, 15]. Initially, mathematical models were used for BOF end-point predictions [16]. However, the precision of these models often fails to satisfy the industry's stringent requirements, even after integrating monitoring and analytic systems like flue gas analysis, flame spectrum assessment, and secondary lance detection [17]. This shortfall highlights the necessity for more advanced modeling techniques, capable of handling the complexity and dynamic nature of the BOF process.

From the beginning to the end of the BOF process, various data in different modalities are recorded. Tabular data are captured at the onset of the process, including several controllable inputs. This prevalence of tabular data has directed much of the research on machine learning (ML) applications within the BOF process toward utilizing tabular datasets [4]. Different ML algorithms have been employed on this data type, with support vector machine (SVM) being the most prevalent [4]. SVM is used in several studies for prediction of end-point temperature and other elements [18–21].

Ensemble learning algorithms, categorized into bagging algorithms and boosting algorithms, have become a cornerstone in machine learning, particularly for handling tabular data [22]. Their ability to integrate insights from multiple models has proven effective in enhancing predictive accuracy and robustness, making them suitable for intricate industrial applications [23, 24]. Despite this, their extensive deployment in the BOF context remains infrequent.

The utilization of machine learning algorithms on big datasets is exemplified by Nenchev et al. [5], Bae et al. [25], and Sala et al. [26], who employed substantial datasets, ranging from 7600 to 9708 heats, for training BOF end-points prediction models. These endeavors collectively validated the efficacy of leveraging big datasets in training models for the BOF process. Among 70 studies that explicitly mention the size of their training datasets, only a few utilize vast datasets exceeding 5000 heats/operations [4].

These highlight the need for more sophisticated ML algorithms such as ensemble models and bigger comprehensive datasets that can capture and encapsulate the BOF process's complexity and industrial scenario more accurately. Our research addresses these gaps by introducing ensemble models trained on an extraordinary industrial-scale dataset comprising over 20,000 heats, with more than 20 significant features. This is the first study of its kind to use a dataset of this scale, including full range of variability in industrial production line.

To explore the predictive accuracy of these extensive data, five ML models were developed to predict the end-point temperature of the BOF process. These models include four ensemble learning algorithms: XGBoost, LightGBM, CatBoost and random forest (RF), and one neural network algorithm (multilayer perceptron (MLP)). A comprehensive comparative analysis was performed to assess the performance of these models, with particular focus on the impact of dataset size on their predictive accuracy.

2 BOF data and ML modeling

2.1 Data from BOF process

BOF is a widely prevalent process in steelmaking, but there are slight variations in the process flow across different plants and sizes. Here is a description of the BOF process flow for the 120-t furnaces involved in this study. The key objectives, timeline and data flow are illustrated in Fig. 1. The main objectives are to elevate temperature and control the composition during the steel transformation. The process starts by charging the furnace with liquid iron and scrap steel, at which point data can be obtained before blowing begins, including the liquid iron composition, liquid iron temperature, liquid iron mass and scrap steel mass. Oxygen is then blown into the furnace to initiate a series of oxidation reactions, necessary for temperature elevation and decarbonization, generating data on blowing parameters such as blowing time and oxygen consumption. Various additives are also added during the process, and the quantities added are recorded. When approximately 80%–90% of the oxygen-blowing process is complete, a sub-lance is utilized for TSC (temperature, sampling, and carbon) detection. Another sub-lance is deployed at the end of oxygen blowing for TSO (temperature, sampling, and oxygen) detection. The results from TSC and TSO testing are crucial data records representing the composition and temperature of the steel in BOF at specific moments. The entire process lasts for 30 to 40 min, with the blowing phase taking up to 10 to 20 min.

2.2 ML models for BOF predicting

In the context of modeling the BOF process, Ghalati et al. [4] provide a statistical overview of the machine learning algorithms utilized in this field. Continuing this line of inquiry, this paper offers an updated statistical analysis of the various algorithms employed in the BOF domain as illustrated in Fig. 2. Despite the proven effectiveness of

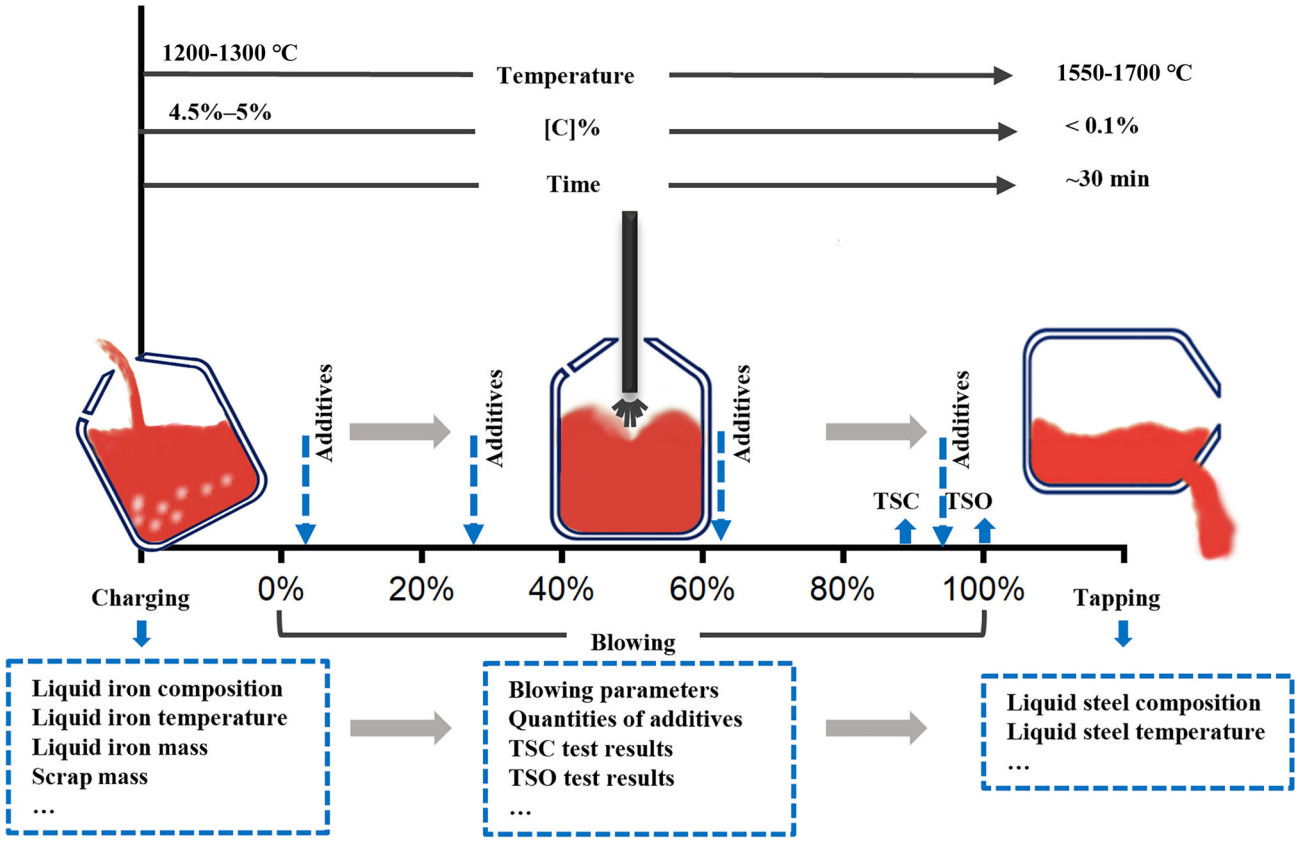
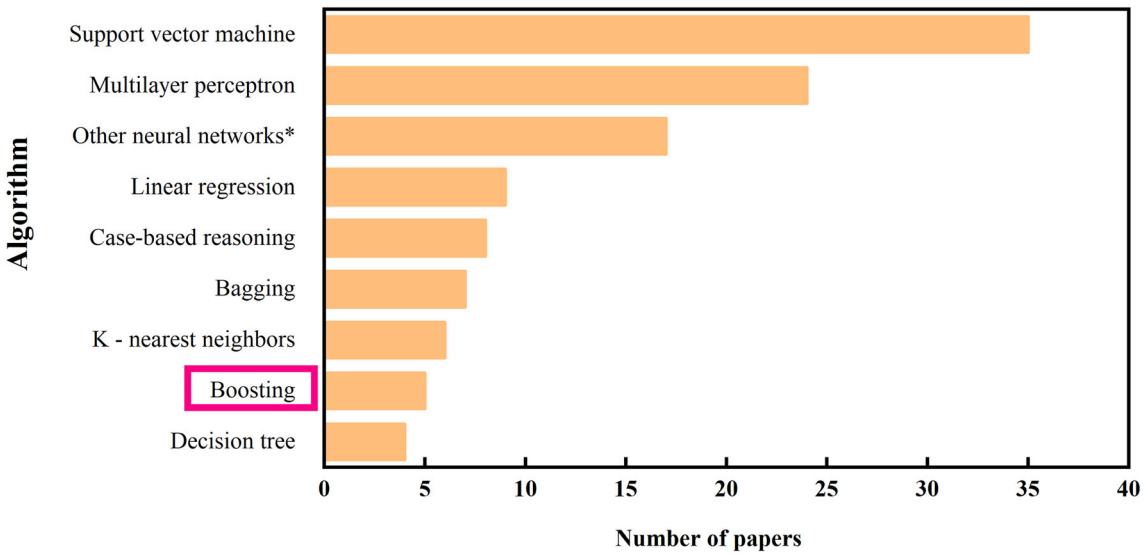


Fig. 1 Key objectives, timeline, and tabular data flow in BOF process



***Neural networks are categorized into two groups: Multilayer perceptron and Other neural networks**

Fig. 2 Application of different algorithm types in BOF data-driven models highlighting rare use of boosting algorithms in this field. *Neural networks are categorized into two groups: multilayer perceptron and other neural networks

ensemble learning algorithms in navigating the complexities of tabular data, their deployment in BOF applications is still not widespread, particularly the case for boosting algorithms.

Ensemble learning algorithms have demonstrated promise in modeling the BOF process. Zhang et al. [27] displayed the strengths of ensemble algorithms, particularly gradient-boosting regression, and random forest regression, in outperforming other models in predicting end-point phosphorus based on a training dataset with 668 samples. Similarly, Sala et al. [28] demonstrated the superior performance of gradient-boosting regression in predicting end-point temperature and various chemical compositions with 4808 training samples.

In the specific field of predicting the BOF end-point temperature, only Bae et al. [25], Sala et al. [26] and Fileti et al. [29] have used datasets containing more than 5000 heats for modeling employed neural network algorithms. Therefore, the effectiveness of boosting algorithms on big datasets in the end-point temperature prediction domain has not been verified yet. Furthermore, since different researchers use different datasets, it is not possible to directly compare the performance of various algorithms on this specific task. To address this gap, we focus on boosting algorithms applied to a large-scale dataset, comparing their performance directly with MLP and random forest. Our analysis reveals that boosting algorithms surpass these benchmarks in both accuracy and stability as dataset size increases, highlighting their robustness and effectiveness in complex BOF applications.

2.3 Ensemble learning

Ensemble learning, a pivotal advancement in machine learning, extends this concept by combining multiple inducers, or sub-models, to enhance decision-making in supervised tasks. Each inducer is an algorithm that processes a set of labeled examples to produce a model, a classifier or regressor, that generalizes from these examples. Ensemble learning is often linked to harassing the wisdom of crowd in machine learning, as it merges several individual opinions to surpass the performance of a single model [30].

In the general framework, building an ensemble model entails selecting a methodology for training the sub-models and choosing a suitable process G for integrating the outputs of n sub-models. Assuming there is a dataset that includes input feature X and output feature Y, the ensemble algorithm can be represented as follows:

$$Y_{\text{pred}} = F(X) = G(\omega_1, \omega_2, \omega_3, \ldots, \omega_n) \quad (1)$$

where Y_{pred} represents the predicted values; F represents the ensemble model; and ω_i represents the sub-models.

Algorithms within ensemble learning are divided into two categories: bagging-type algorithms and boosting-type algorithms, with the majority of most recent ensemble learning algorithms derived from these categories.

Bagging algorithm, a well-established ensemble technique, employs random sampling with replacement to generate subsets of the training dataset. These subsets train multiple base models independently, and the final model aggregates the outcomes of these individual models [31]. Figure 3 illustrates the schematic diagram of the fundamental framework of bagging algorithms, which is conducted in a parallel manner. Random forest is a well-known algorithm derived from the concept of bagging [32].

Boosting algorithm incrementally improves model performance by integrating the strengths of a sequence of weak learners [33]. It starts by training a basic learner on the original training dataset and then adjusts the training samples' weighting based on the different prediction performances. For classification tasks, this adjustment is based on the misclassification rate, while for regression, it is based on the residual errors of the predictions. The process is repeated until a pre-defined criterion is met, culminating in a robust predictive model from the ensemble of weak learners. Figure 4 illustrates the schematic diagram of the fundamental framework for boosting algorithms, which is conducted in a tandem manner.

Many researchers have made improvements to the basic workflow of boosting algorithms, leading to the development of new boosting algorithms. A prime example of a boosting algorithm is the gradient-boosting decision tree [34]. In recent years, many advanced boosting algorithms have emerged, such as XGBoost [35], LightGBM [36], and CatBoost [37].

Ensemble learning addresses unique machine learning challenges such as class imbalance, concept drift, and the course of dimensionality and finds extensive application in a variety of fields [30]. Furthermore, ensemble algorithms, like XGBoost, are commonly regarded as the preferred choice for addressing real-world tabular data problems [22].

3 Modeling procedures

3.1 Data description

In this study, a dataset comprising 39,591 heats, sourced from a Chinese steel company, was utilized. This dataset, collected from April 2021 to March 2022, represents a wide range of production diversity, including 259 different steel grades produced in three similar 120-t basic oxygen furnaces within the same steel plant. A meticulous selection and feature extraction process was applied to the raw

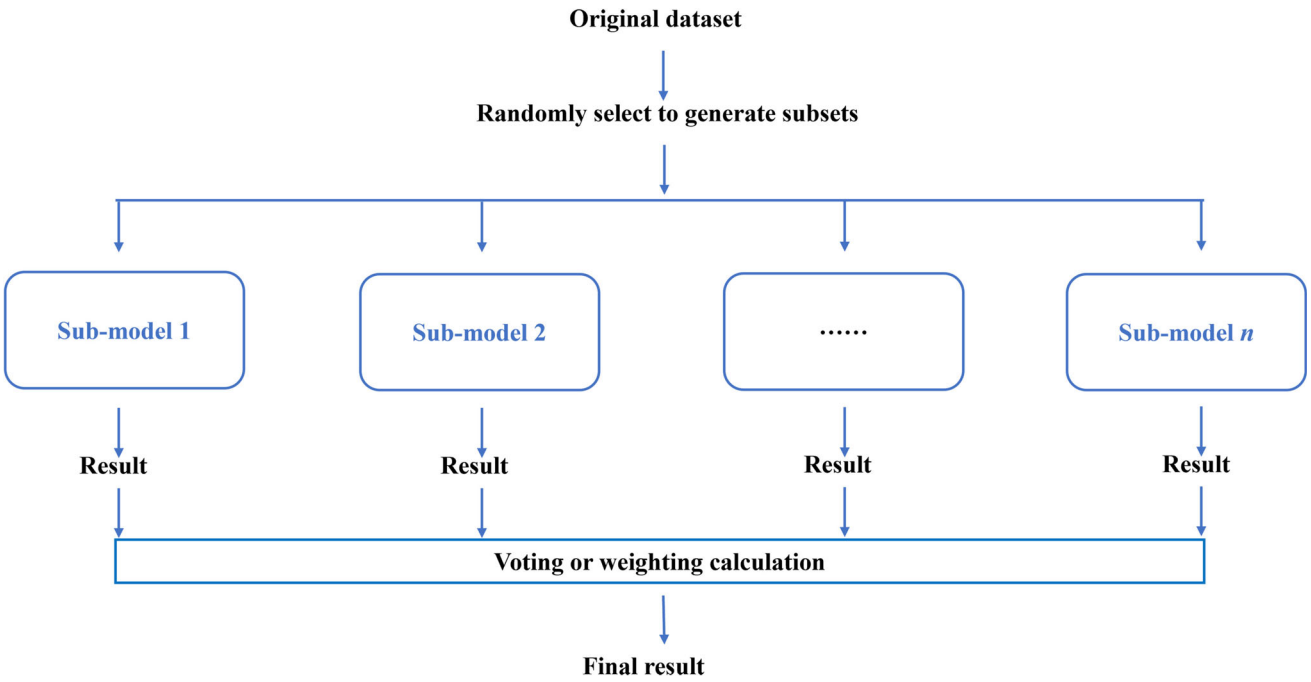


Fig. 3 Schematic diagram of fundamental framework of bagging algorithms by random sampling with replacement conducted in a parallel manner

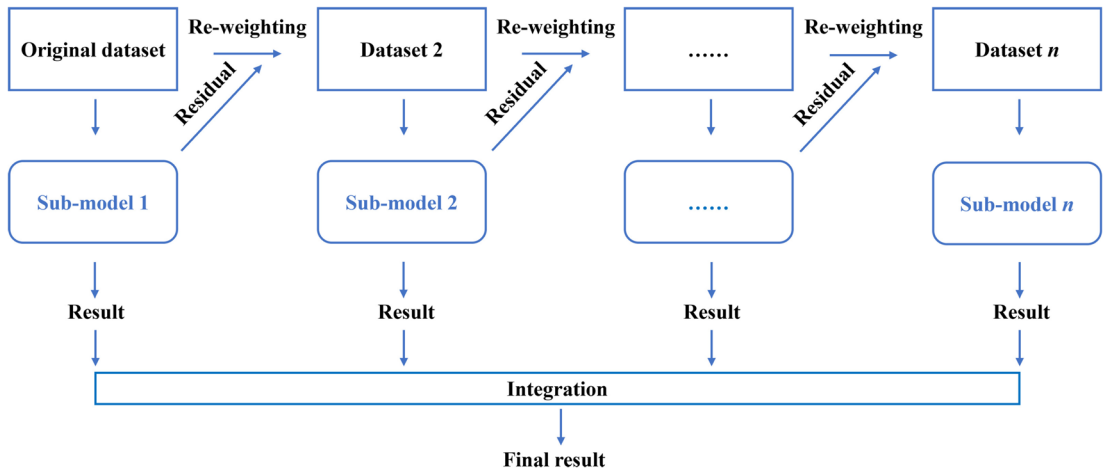


Fig. 4 Schematic diagram of fundamental framework for boosting algorithms, conducted in a tandem manner

data, converting it into a structured data frame of pertinent features for detailed analysis. These features encompassed a broad spectrum of data, including liquid iron characteristics, scrap details, additives, blowing parameters, and TSO results.

3.2 Preprocessing

In the initial phase of preprocessing, a decision was made to exclude heats with missing data. Given the considerable size of the dataset, this approach was taken to prioritize the integrity and quality of the data over the option of imputation. Following this, attention was directed to the BOF process where the TSO test results do not meet the production requirements, and a re-blow stage is initiated to continue blowing or add extra additives. Heats requiring this re-blow stage are categorized as abnormal, and were removed from the dataset, as their production data are not suitable for inclusion in the training set.

Outlier detection and removal were employed using the actual feature range and the interquartile range (IQR) method. This statistical technique, based on data's quartiles, helps establish a normal data range, with points outside this range considered outliers. This approach involved

calculating the first quartile (Q_1), also known as the lower quartile, and the third quartile (Q_3), which is the value of the upper quartile. The IQR is calculated as the difference between Q_3 and Q_1 (IQR = Q_3-Q_1) and represents the range of the middle 50% of the data.

Outlier boundaries were determined using a multiplier 'x', setting the lower bound as $Q_1-x \times$ IQR and the upper bound as $Q_3 + x \times$ IQR. Values beyond these boundaries were identified as outliers and are removed.

The 'x' multiplier was carefully chosen after testing values between 1.5 and 3, with 2 being optimal choice, and balancing the need to remove outliers against value may fail the risk of excessive data loss in the complex real-world factory setting.

Before modeling, data normalization was achieved using the MinMaxScaler, transforming each feature value to scale from 0 to 1 based on its minimum and maximum values. The MinMaxScaler transfers the original value by:

$$X_{\text{norm}} = \frac{X' - X_{\min}}{X_{\max} - X_{\min}} \tag{2}$$

where X' is the original value; $X_{\min}$ and $X_{\max}$ are the minimum and maximum values, respectively; and X_{norm} is the value after scaling.

3.3 Feature engineering

The heatmap presented in Fig. 5a illustrates low correlations among the key features, with most of the values under 0.4. Similarly, Fig. 5b shows the absolute correlation computed by the Pearson correlation coefficient, indicating that these key features have a limited correlation with the temperature detection result by TSO (T_TSO) after blowing, which serves as the target for the model, not surpassing 0.35. This low correlation underscores the necessity for powerful algorithms like ensemble algorithms to capture the relationships between features to enhance the model's predictive accuracy.

The data distribution of T_TSO presented a significant deviation from a normal distribution, as depicted in Fig. 6. This is because different grades of steel are produced due to the varying tasks even within the same furnace, which reflects a situation arising from actual onsite conditions. To mitigate the potential impact of this situation on the modeling process, the steel grades were divided into 'low T_TSO' and 'high T_TSO' steel grade groups. This categorization was implemented by creating a new feature in the dataset, aligning the model with observed data distribution.

3.4 Model establishment

Following data extraction and preprocessing, the dataset was refined to include 20,416 heats. 80% of the heats were randomly selected for training purposes and the remaining 20% for testing. For the end-point temperature prediction of BOF process, the modeling employed five algorithms: four from the ensemble learning algorithms, namely XGBoost, LightGBM, CatBoost (which are boosting algorithms), and RF (a bagging algorithm), alongside a neuronal network model, the MLP. MLP is the most used type of neural network for tabular data [38]. This neural network model serves as a benchmark to assess the comparative effectiveness of the algorithms.

3.5 Tuning hyperparameters using Bayesian optimization

Hyperparameter tuning, an indispensable phase in model refinement, has evolved from manual, experience-heavy approaches to more sophisticated methods like grid search and random search. While grid search offers systematic exploration within a parameter space, random search surpasses it in efficiency, yet neither method adjusts based on previous results.

In this research, Bayesian optimization is adeptly applied across models, utilizing Gaussian processes for its flexibility and effectiveness in model tuning. Bayesian optimization with Gaussian processes transcends the limitations of previous approaches, leveraging past search outcomes to guide future exploration, thereby optimizing performance with enhanced efficiency [39]. This method has been validated for its capacity to enhance the performance of boosting learning models [40].

Hyperparameter optimization for each model was systematically conducted using Bayesian search with the Gaussian process by fivefold cross-validation. The evaluation was conducted using 'negative mean squared error' as the scoring. Early stopping was incorporated into the search process, whereby the search would stop if no enhancement in model performance after successive 300 iterations.

The evaluation of the models' performance was grounded in three metrics; root mean square error (RMSE), mean absolute error (MAE), and R-squared (R^2), which will collectively offer a multi-facet view of model's accuracy and predictability. RMSE, representing the average of the squared differences between predicted and actual values could be determined as:

$$\text{RMSE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}\left(y_i - y_i^{\text{pred}}\right)^2} \tag{3}$$

$$\text{MAE} = \frac{1}{n}\sum_{i=1}^{n}\left|y_i - y_i^{\text{pred}}\right| \tag{4}$$

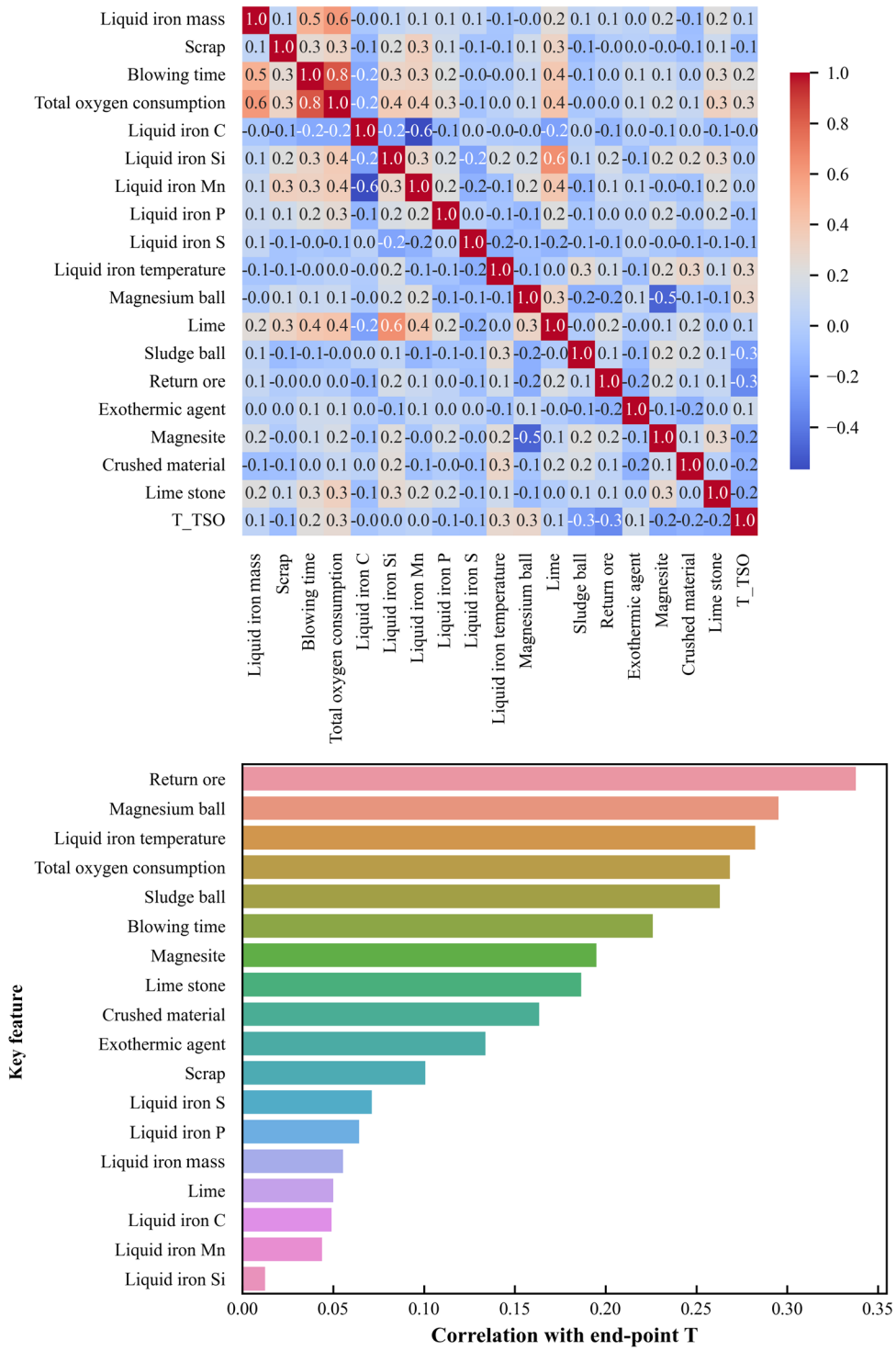


Fig. 5 Correlation map of features, illustrating low correlation among key features (**a**) and sorted absolute correlation between target T_TSO and key features ranging up to 0.35 showing low correlation of BOF variables with T_TSO (**b**)

$$R^2 = 1 - \frac{\sum \left(y_i - y_i^{\text{pred}}\right)^2}{\sum \left(y_i - y_{\text{mean}}\right)^2} \tag{5}$$

where R^2 reflects the proportion of variance in the dependent variable explained by the independent variables in the model; y_i is the actual values; y_i^{pred} is the predicted values; and y_{mean} is the mean of the actual values.

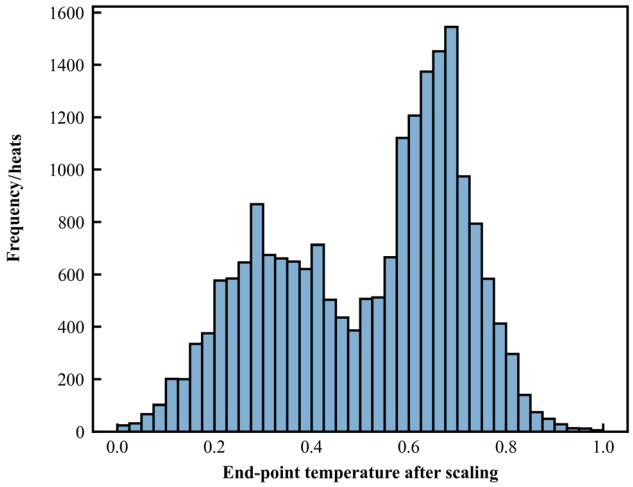


Fig. 6 Distribution of T_TSO presenting a significant deviation from a normal distribution with a new feature for steel grade clustering created to mitigate impact of this distribution on modeling

4 Results and discussion

In this section, a detailed analysis of machine learning model's performance in the context of BOF is presented. This includes examining the model performance before and after hyperparameter tuning to identify the most effective model in terms of accuracy and efficiency and assess the impact of varying training set sizes.

4.1 Comparative analysis of model's performance

Bayesian optimization significantly improves the performance of algorithms on the testing dataset, with boosting algorithms showcasing their strengths.

To explore the optimal performance of various algorithms, a Bayesian search with Gaussian process optimization was executed. Each algorithm's parameters were meticulously selected based on data characteristics, and a reasonable search space was established. The optimal parameter combinations identified for each model are detailed in Appendix (Table A1). Following the optimization, models were built using the optimal parameters and then evaluated on both training and testing data using RMSE, MAE, and R^2. Table 1 provides the performance metrics of all models on training and test data before and after hyperparameter tuning. The Bayesian optimization approach for tuning has proven highly effective in improving model performance on the testing dataset. This demonstrates its effectiveness in mitigating overfitting and enhancing the model's generalizability of all models except random forest.

Although random forest algorithm, employing bagging, demonstrated superior performance on the training data with the highest R^2 value of 0.968, accompanied by the lowest MAE and RMSE compared to other algorithms, it suffers from significant overfitting. The Bayesian hyperparameter tuning, despite its effectiveness for boosting algorithms, has been unable to resolve this issue, leading to a noticeable difference in R^2 values between training and test datasets.

As shown in Fig. 7, CatBoost outperformed others in the testing phase, with an R^2 of 0.818, closely followed by XGBoost and LightGBM with R^2 values of 0.815 and 0.813, respectively. MLP and random forest recorded R^2 values of 0.812 and 0.785 on testing data, respectively. R^2 of CatBoost test improved by 4.2% compared to that of the random forest and by 0.8% compared to that of the multilayer perceptron. A correlation was noted where higher R^2 values coincide with lower MAE and RMSE,

Table 1 Performance of various models before and after hyperparameter tuning

Condition	Algorithm	Train result			Test result		
		RMSE	MAE	R^2	RMSE	MAE	R^2
Before hyperparameter tuning	XGBoost	0.056	0.043	0.917	0.089	0.068	0.794
	LightGBM	0.075	0.058	0.852	0.086	0.065	0.810
	CatBoost	0.059	0.053	0.876	0.085	0.065	0.815
	RF	0.036	0.027	0.967	0.091	0.070	0.785
	MLP	0.036	0.068	0.790	0.087	0.066	0.806
After hyperparameter tuning	XGBoost	0.072	0.055	0.866	0.085	0.065	0.815
	LightGBM	0.078	0.060	0.841	0.085	0.065	0.813
	CatBoost	0.067	0.051	0.882	0.084	0.064	0.818
	RF	0.035	0.027	0.968	0.091	0.070	0.785
	MLP	0.087	0.066	0.805	0.085	0.065	0.812

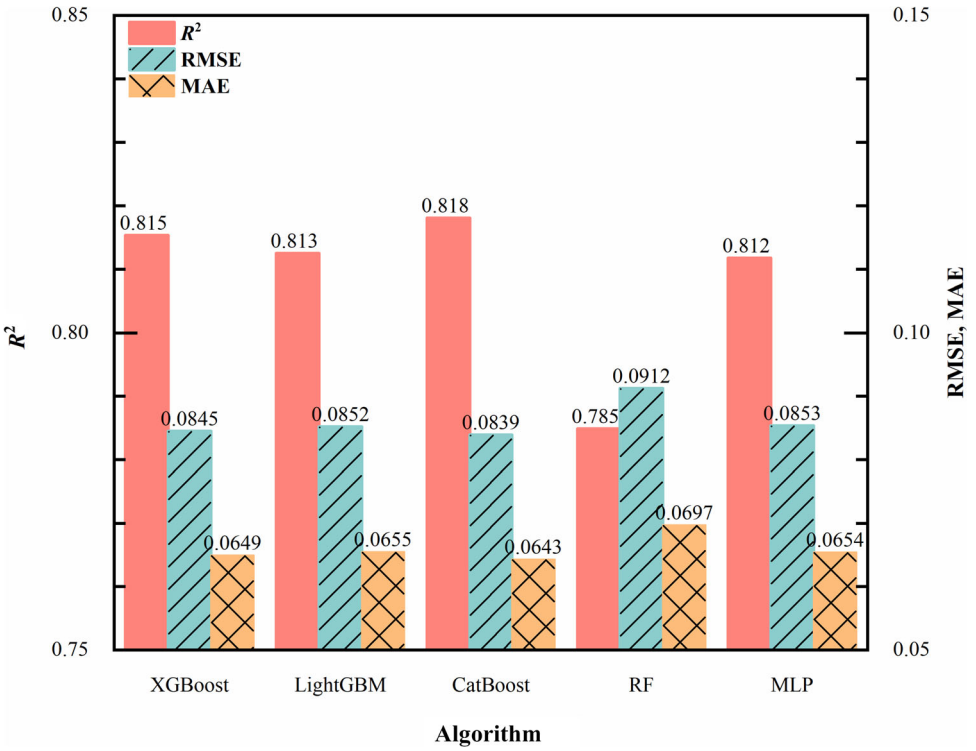


Fig. 7 Performance of models on testing dataset after hyperparameter tuning, with CatBoost showing best outcomes

validating R^2 as an effective performance evaluator. Compared to the performance of MLP, the performance of boosting models is slightly higher, demonstrating the advantages of the boosting algorithm. The CatBoost, thus, was proved to be the most effective model, considering R^2, MAE, and RMSE on the testing data.

Figure 8 provides a visual comparison of the prediction values against actual values for various algorithms on the test dataset. Predicted values aligning perfectly with actual values appear on the red solid line, while predictions within a ±0.1 error range fall between the red dashed lines. The boosting algorithm (Fig. 8a, b, d) and MLP (Fig. 8c) demonstrate a more effective convergence in the distribution of data points. The hit ratio representing the percentage of prediction within an acceptable range, for these models, is close to 80% (±0.1 error range), with boosting algorithms achieving superior results.

The scatter plot for RF (Fig. 8e) shows two obvious distinct clusters, possibly due to the fact that the T_TSO distribution's dual-peak nature and the addition of engineered features based on data distribution established in Sect. 3.3 are not effective for the RF model. Despite the robustness of RF, its prediction in extreme temperature regions displays significant distortions. This highlights the superior adaptability of boosting algorithms in complex industrial data scenarios.

4.2 Impact of dataset size on prediction accuracy

The study on the significance of industrial-scale data size for model accuracy encompasses a comprehensive experimental framework. This extensive analysis involves the application of five algorithms to datasets of varying sizes, ranging from 500 to 20,000 samples with incrementing by 500 samples at each stage. The processes include 40 distinct training instances across five different models. The methodology retained consistency in the features and the data splitting strategy, as mentioned in Sect. 4.1. To minimize the influence of random sampling, 10 different values were selected as random states in both the sampling selection and the data splitting. This approach resulted in each algorithm being modeled 100 times for every instance of specific dataset size. The performance metrics for each algorithm, corresponding to each dataset size, were then determined by averaging the outcome of these 100 iterations. The performance metrics of this comprehensive analysis are detailed in Fig. 9.

With trained R^2, as depicted in Fig. 9a, the RF model demonstrated superior performance, with minimal variation as the dataset size changed. This is due to the fact that random forest builds many decision trees, using different data samples and feature subsets for each tree, which allows it to fit the training data very well. Therefore, its R^2 value on the training set is extremely high. In the case that the three

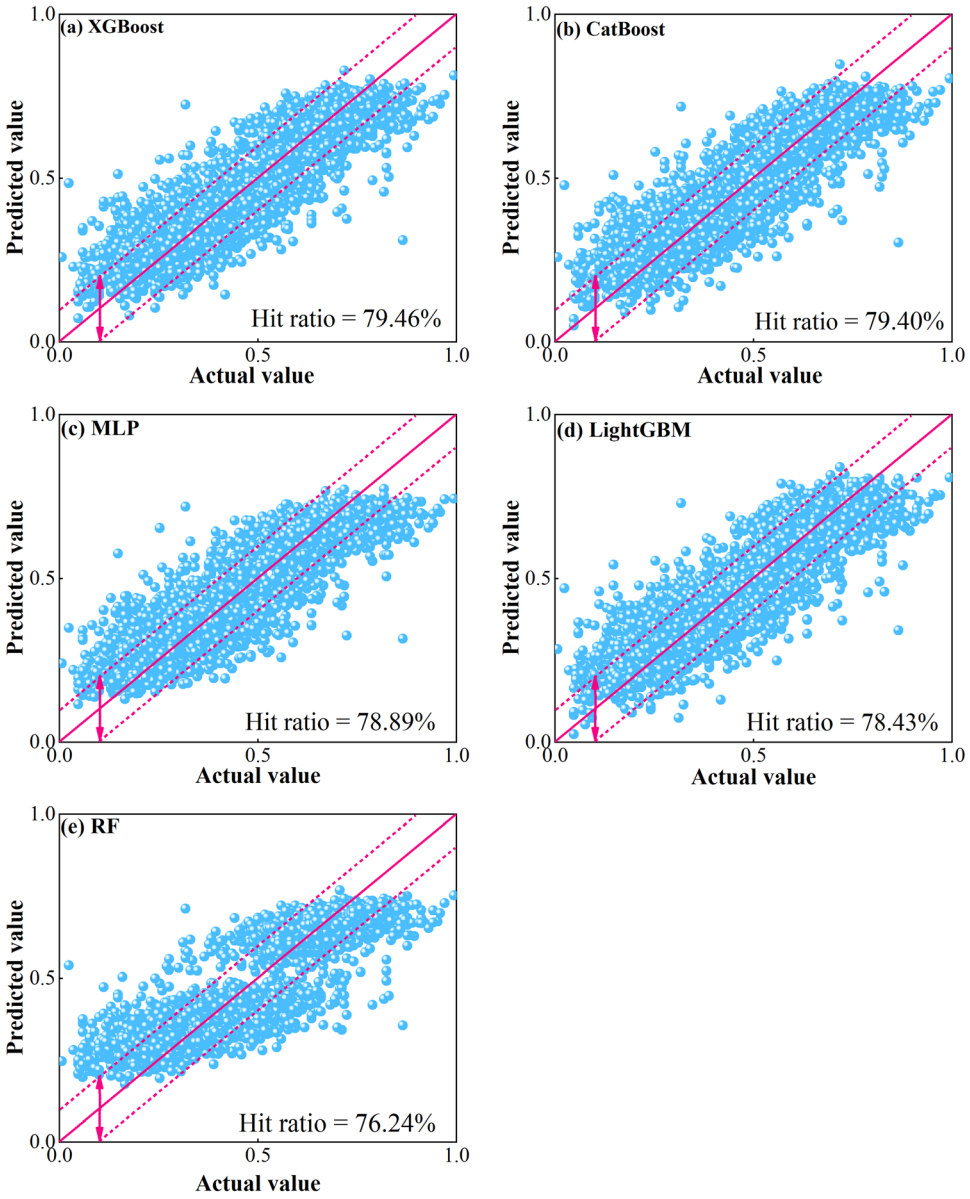


Fig. 8 Actual *vs.* predicted values for XGBoost (**a**), CatBoost (**b**), MLP (**c**), LightGBM (**d**), and RF (**e**) algorithms

boosting algorithms initially registered high train R^2 values close to 1 with smaller datasets, however, these values decreased as the dataset size increased. The boosting algorithms mentioned here are based on gradient boosted decision trees. Gradient boosting reduces errors by gradually building the model, with each new model focusing on the residuals of the previous one. As the amount of data increases, the model gradually identifies and corrects its deficiencies. Therefore, it may no longer consider the specifics of each individual case, but instead, it needs to consider the overall situation of the data. In contrast, MLP maintained a lower and more stable trained R^2 throughout. Turning attention to the test R^2 values presented in Fig. 9b, a common initial trend was observed across all algorithms demonstrating lower values with smaller datasets, likely due to the insufficiency of data for capturing the relationships between features. However, as the dataset size increased, the tested R^2 for all models gradually increased. Among all models, CatBoost emerged as the best-performing algorithm. Considering the changes in R^2 on the training dataset, it can be easily observed that the difference in R^2 between the training dataset and testing dataset for the boosting algorithm decreases as the amount of data increases. This indicates that the boosting algorithm can effectively reduce model overfitting by increasing the amount of data, which means that the generalization ability of the model is improving.

The trend observed in the tested MAE and tested RMSE metrics (Fig. 9c, d) mirrored the pattern of the tested R^2

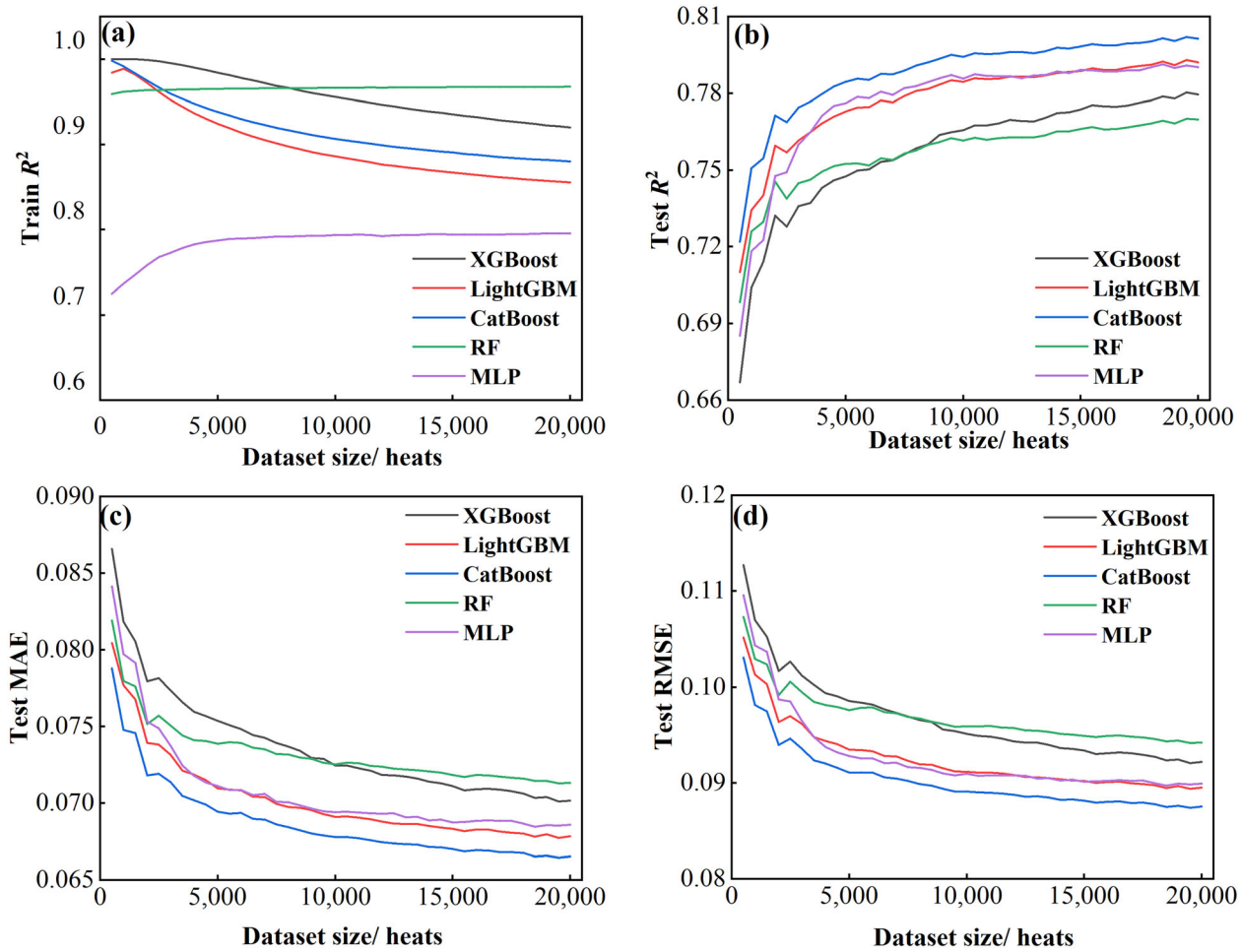


Fig. 9 Comparison of R^2 (trained) (**a**), R^2 (tested) (**b**), MAE (tested) (**c**), and RMSE (tested) (**d**) of each algorithm model on different size datasets

values. The performance of all algorithms improved, aligning with the increase in dataset size.

These results conclusively demonstrate the significant positive impact of dataset size on the performance of machine learning models, particularly in the complex BOF process. It underscores the role of dataset sizes in applying machine learning to the BOF process. This further emphasizes the advantages of the boosting algorithm in large datasets.

5 Conclusions

In this comprehensive study, we rigorously evaluated five sophisticated machine learning algorithms including three boosting algorithms (XGBoost, LightGBM, CatBoost), an RF and an MLP on predicting the end-point temperature in the basic oxygen furnace process.

We conducted extensive preprocessing, correlation analysis, and feature engineering, addressing the low correlation between features and end-point temperature. Despite this low correlation, our approach to feature engineering, combined with advanced Bayesian-optimized ML models, mitigated the challenge, and delivered significant improvements. Among these, Bayesian-optimized CatBoost distinguished itself as the most effective model with an R-squared value of 0.818 on testing data, and Bayesian-optimized XGBoost led the highest hit ratio (±0.1 error range), underscoring the superiority of boosting algorithms in this domain.

We utilized an extensive dataset of over 20,000 heats, significantly bigger than those in prior studies, encompassing a wide array of data variability including different furnaces and a wide range of steel grades. We conducted a broad-ranging analysis of how the size (spanning from 500 to 20,000 heats) of the dataset influences model performance, uncovering a clear positive effect of dataset size on the efficacy of BOF prediction models.

Our methodology, combining substantial data, and advanced machine learning techniques, serves as a foundation for future enhancements in this area, emphasizing the critical role of industrial-scale dataset size and advanced algorithms in improving predictive accuracy and operational efficiency in manufacturing processes.

Appendix

See Table A1.

Table A1 Parameters selected, search spaces, and optimal values in hyperparameter tuning process for each model

Algorithm	Hyperparameter parameter	Search space	Best value
XGBoost	colsample_bytree	[0.5, 1.0]	0.911
	gamma	[0, 6]	0.042
	learning_rate	[0.005, 0.5]	0.024
	max_depth	[3, 10]	6
	min_child_weight	[1, 10]	9
	n_estimators	[100, 3000]	1062
	reg_alpha	[0, 10]	0
	reg_lambda	[0, 10]	9.876
	subsample	[0.5, 1.0]	0.502
LightGBM	boosting_type	['gbdt', 'dart']	'dart'
	colsample_bytree	[0.5, 1.0]	0.987
	learning_rate	[0.005, 0.5]	0.181
	max_bin	[20, 2000]	504
	max_depth	[3, 10]	3
	min_child_samples	[20, 100]	46
	n_estimators	[100, 3000]	1768
	num_leaves	[31, 127]	105
	reg_alpha	[0, 10]	1.195
	reg_lambda	[0, 10]	6.994
	subsample	[0.5, 1.0]	0.953
CatBoost	colsample_bylevel	[0.5, 1.0]	0.840
	learning_rate	[0.005, 0.5]	0.022
	max_depth	[1, 10]	7
	min_data_in_leaf	[1, 10]	6
	iterations	[100, 3000]	2616
	bagging_temperature	[0, 10]	9.942
	l2_leaf_reg	[0, 10]	6.188
	subsample	[0.5, 1.0]	0.520
RF	max_depth	[None, [3, 20]]	None
	max_features	['sqrt', 'log2']	'sqrt'
	min_samples_leaf	[1, 20]	1
	min_samples_split	[2, 20]	2
	n_estimators	[100, 3000]	2300
MLP	hidden_layer_size_1	[50, 200]	199
	hidden_layer_size_2	[25, 100]	74
	hidden_layer_size_3	[5, 50]	44
	activation	['tanh', 'relu']	'relu'
	alpha	[0.000,05, 0.5]	0.000,05
	learning_rate_init	[0.005, 0.5]	0.0053
	max_iter	[100, 500]	119

Acknowledgements All authors gratefully acknowledge the support from Nanjing Iron & Steel United Co., Ltd. (NISCO), particularly for providing a Ph.D. scholarship for this study.

Declarations

Conflict of interest The authors declare no conflict of interest.

References

[1] T.A. Branca, B. Fornai, V. Colla, M.M. Murri, E. Streppa, A.J. Schröder, Metals 10 (2020) 288.
[2] Z. Jan, F. Ahamed, W. Mayer, N. Patel, G. Grossmann, M. Stumptner, A. Kuusk, Expert Syst. Appl. 216 (2023) 119456.
[3] J. Brandenburger, V. Colla, G. Nastasi, F. Ferro, C. Schirm, J. Melcher, IFAC-PapersOnLine 49 (2016) 55–60.
[4] M.K. Ghalati, J. Zhang, G.M.A.M. El-Fallah, B. Nenchev, H. Dong, Mater. Genome Eng. Adv. 1 (2023) e6.
[5] B. Nenchev, C. Panwisawas, X. Yang, J. Fu, Z. Dong, Q. Tao, J.C. Gebelin, A. Dunsmore, H. Dong, M. Li, B. Tao, F. Li, J. Ru, F. Wang, Steel Res. Int. 93 (2022) 2100813.
[6] World Steel Association, Steel-Statistical-Yearbook, Brussels, 2020.
[7] Q. Li, C. Liu, Q. Guo, Mathematics 10 (2022) 2659.
[8] J.W. Guo, D.P. Zhan, G.C. Xu, N.H. Yang, B. Wang, M.X. Wang, G.W. You, J. Iron Steel Res. Int. 30 (2023) 875–886.
[9] K. Feng, L. Yang, B. Su, W. Feng, L. Wang, Steel Res. Int. 93 (2022) 2100433.
[10] D. Wang, F. Gao, L. Xing, J. Chu, Y. Bao, Metals 12 (2022) 151.
[11] M.Q. Gu, A.J. Xu, F. Yuan, X.M. He, Z.F. Cui, ISIJ Int. 61 (2021) 2564–2570.
[12] B.K. Mahanta, P. Gupta, I. Mohanty, T.K. Roy, N. Chakraborti, Digit. Chem. Eng. 7 (2023) 100094.
[13] K.X. Zhou, W.H. Lin, J.K. Sun, J.S. Zhang, D.Z. Zhang, X.M. Feng, Q. Liu, J. Iron Steel Res. Int. 29 (2022) 751–760.
[14] Z. Wang, Y. Bao, C. Gu, Steel Res. Int. 94 (2023) 2370011.
[15] Z. Liu, S. Cheng, P. Liu, High Temp. Mater. Process. 41 (2022) 505–513.
[16] A.A.R. Hanafy, M.M.F. Sakr, IFAC Proc. Vol. 20 (1987) 537–542.
[17] Z. Wang, Q. Liu, H. Liu, S. Wei, High Temp. Mater. Process. 39 (2020) 653–662.
[18] Y. Shao, M. Zhou, Y. Chen, Q. Zhao, S. Zhao, Optik 125 (2014) 2491–2496.
[19] C.J. Zhang, Y.C. Zhang, Y. Han, J. Ind. Inf. Integr. 28 (2022) 100356.

[20] J. Schlueter, H.J. Odenthal, N. Uebber, H. Blom, K. Morik, in: Proceedings of the Iron & Steel Technology Conference, AIST, Nashville, USA, 2013, pp. 923–928.
[21] C. Gao, M. Shen, X. Liu, L. Wang, M. Chen, Trans. Indian Inst. Met. 72 (2019) 257–270.
[22] R. Shwartz-Ziv, A. Armon, Inf. Fusion 81 (2022) 84–90.
[23] X. Yang, G.M.A.M. El-Fallah, Q. Tao, J. Fu, C. Leng, J. Shepherd, H. Dong, Mater. Today Commun. 34 (2023) 105162.
[24] M. Bertolini, D. Mezzogori, M. Neroni, F. Zammori, Expert Syst. Appl. 175 (2021) 114820.
[25] J. Bae, Y. Li, N. Ståhl, G. Mathiason, N. Kojola, Metall. Mater. Trans. B 51 (2020) 1632–1645.
[26] D.A. Sala, A. Van Yperen-De Deyne, E. Mannens, A. Jalalvand, Appl. Intell. 53 (2023) 15163–15173.
[27] R. Zhang, J. Yang, S. Wu, H. Sun, W. Yang, Steel Res. Int. 94 (2023) 2200682.
[28] D.A. Sala, A. Jalalvand, A. Van Yperen-De Deyne, E. Mannens, in: M.A. Wani, M. Kantardzic, M.S. Mouchaweh, J. Gama, E. Lughofer (Eds.), 2018 17th IEEE International Conference on Machine Learning and Applications, ICMLA, IEEE, pp. 1419–1426.
[29] A.M.F. Fileti, T.A. Pacianotto, A.P. Cunha, Eng. Appl. Artif. Intell. 19 (2006) 9–17.
[30] O. Sagi, L. Rokach, Wires Data Min. Knowl. Discov. 8 (2018) e1249.
[31] L. Breiman, Mach. Learn. 24 (1996) 123–140.
[32] L. Breiman, Mach. Learn. 45 (2001) 5–32.
[33] R.E. Schapire, Mach. Learn. 5 (1990) 197–227.
[34] J.H. Friedman, Ann. Statist. 29 (2001) 1189–1232.
[35] T. Chen, C. Guestrin, Handbook on knowledge management, Springer Nature, Berlin, Heidelberg, Germany, 2023.
[36] G. Ke, Q. Meng, T. Finley, T. Wang, W. Chen, W. Ma, Q. Ye, T.Y. Liu, in: I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, R. Garnett (Eds.), Advances in Neural Information Processing Systems 30 (NIPS 2017), Long Beach, CA, USA, 2017.
[37] L. Prokhorenkova, G. Gusev, A. Vorobev, A.V. Dorogush, A. Gulin, in: S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, R. Garnett (Eds.), Advances in Neural Information Processing Systems 31 (NeurlPS 2018), Montréal, Canada, 2018.
[38] M.C. Popescu, V.E. Balas, L. Perescu-Popescu, N. Mastorakis, WSEAS Transactions on Circuits and Systems 8 (2009) 579–588.
[39] J. Wu, X.Y. Chen, H. Zhang, L.D. Xiong, H. Lei, S.H. Deng, Journal of Electronic Science and Technology 17 (2019) 26–40.
[40] V.H. Nguyen, T.T. Le, H.S. Truong, M.V. Le, V.L. Ngo, A.T. Nguyen, H.Q. Nguyen, Math. Probl. Eng. 1 (2021) 6815802.