scientific reports

OPEN

Developing a cost-effective tool for choke flow rate prediction in sub-critical oil wells using wellhead data

Zhiwei Xun[1]✉, Farag M. A. Altalbawy[2], Prakash Kanjariya[3], R. Manjunatha[4], Debasish Shit[5], M. Nirmala[6], Ajay Sharma[7], Sarbeswara Hota[8], Shirin Shomurotova[9], Fadhil Faez Sead[10,11,12], Hojjat Abbasi[13]✉ & Mohammad Mahtab Alam[14,15]

Accurate prediction of oil production rates through wellhead chokes is critical for optimizing crude oil production and operational efficiency in the petroleum industry. The central thrust of this investigation involves the systematic creation of machine learning (ML) paradigms for the robust prediction of choke flow performance. This endeavor is rigorously informed by comprehensive data acquired from an operational petroleum production facility in the Middle East. Within the dataset, produced gas-oil ratio (GOR), choke size, basic sediment and water (BS&W), wellhead pressure (THP), and crude oil API stand out as key parameters. Each plays a vital role in forecasting the oil production rate. To ensure reliability, robust data preprocessing was conducted using the Monte Carlo outlier detection (MCOD) method to recognize and manage data outliers. The models were trained using 198 data points, employing K-fold cross-validation (five folds) to ensure generalization. Gradient boosting machine (GBM) models were optimized using advanced algorithms like self-adaptive differential evolution (SADE), evolution strategy (ES), Bayesian probability improvement (BPI), and Batch Bayesian optimization (BBO). Among these, SADE demonstrated superior performance based on metrics such as average absolute relative error (AARE%), R^2, and mean squared error (MSE). Furthermore, SHAP (SHapley Additive exPlanations) analysis was used to interpret the models and highlight the dominant influence of choke size and THP on the predictions. Overall, this research work presents a data-driven framework for highly accurate and interpretable predictions, significantly contributing to production optimization initiatives in the oil and gas sector.

Keywords Choke flow modeling, Machine learning, Crude oil production, Optimization, SHAP analysis

Remaining a critical component of the global economy, oil serves as a primary energy source and raw material. It fuels diverse industries and underpins the production of chemicals, plastics, medicines, and many other goods[1–3]. Oil-rich nations, particularly those with vast production capabilities, harness their petroleum

[1]China University of Geosciences (Beijing), Beijing 100083, China. [2]Department of Chemistry, University College of Duba, University of Tabuk, Tabuk, Saudi Arabia. [3]Department of Physics, Marwadi University Research Center, Faculty of Science, Marwadi University, Rajkot, Gujarat, India. [4]Department of Data Analytics and Mathematical Sciences, School of Sciences, JAIN (Deemed to be University), Bangalore, Karnataka, India. [5]Centre for Research Impact and Outcome, Chitkara University Institute of Engineering and Technology, Chitkara University, Rajpura, Punjab 140401, India. [6]Department of Mathematics, Sathyabama Institute of Science and Technology, Chennai, Tamil Nadu, India. [7]Department of Applied Sciences-Mathematics, NIMS Institute of Engineering and Technology, NIMS University Rajasthan, Jaipur, India. [8]Department of Computer Application, Siksha 'O' Anusandhan (Deemed to be University), Bhubaneswar, Odisha 751030, India. [9]Department of Chemistry Teaching Methods, Tashkent State Pedagogical University Named After Nizami, Bunyodkor Street 27, Tashkent, Uzbekistan. [10]Department of Dentistry, College of Dentistry, The Islamic University, Najaf, Iraq. [11]Department of Medical Analysis, Medical Laboratory Technique College, The Islamic University of Al Diwaniyah, Al Diwaniyah, Iraq. [12]Department of Medical Analysis, Medical Laboratory Technique College, The Islamic University of Babylon, Babylon, Iraq. [13]Chemistry Department, Herat University, Herat, Afghanistan. [14]Central Labs, King Khalid University, AlQura'a, P.O. Box 960, Abha, Saudi Arabia. [15]Department of Basic Medical Sciences, College of Applied Medical Science, King Khalid University, 61421 Abha, Saudi Arabia. ✉email: Zhiwei0914@outlook.com; hojjatabbasimeybodi@gmail.com

wealth as a primary engine for prosperity. The substantial financial returns from exporting oil provide these countries with the means to bolster their foundational infrastructure, stimulate industrial expansion, and foster innovation through technological uptake, thereby fast-tracking their overall economic and social evolution[4–6]. For companies, the industry represents a major revenue stream through activities such as extraction, refining, and distribution[7,8]. As a result, the accurate prediction and optimal management of oil production are essential for both oil-producing countries and the companies operating within the sector[9–11].

Accurately predicting the future output of oil wells—that is, the quantity of crude extracted over a specific period—is an absolutely vital strategic endeavor for both oil and gas corporations and national economic strategist[12,13]. Reliable forecasts allow for efficient resource management, cost optimization in extraction, and the fulfillment of domestic and global energy demands[14–16]. Furthermore, these accurate predictions are instrumental in shaping key operational decisions, from determining where to drill new wells and when to schedule essential maintenance, to implementing effective long-term resource management strategies. Production forecasting holds immense importance for both corporate strategies and macroeconomic planning[17]. However, due to the complexities of oil reservoirs and the various factors influencing production including geological conditions, reservoir pressure, and oil characteristics, accurate forecasting remains a significant challenge in the oil industry[18,19].

Since the early twentieth century, petroleum engineers have continuously worked on forecasting oil well production rates. This has involved developing and employing various models to simulate reservoir oil flow and refine production predictions for over a hundred years[20]. Such efforts have led to the development of empirical equations which are derived from experimental data and rely on parameters pertaining to reservoir and well characteristics[21]. Among these, the Gilbert equation is a well-known approach often cited for its simplicity and cost-efficiency. These equations can provide reasonable predictions for well production under specific conditions resembling their development data. However, their reliability diminishes when applied to new or diverse scenarios[22,23].

One of the primary limitations of empirical equations is their lack of comprehensiveness. These equations are typically derived from experimental data specific to a particular reservoir or set of conditions that may not generalize well to different wells or environments[24–26]. Geological differences, as well as variations in reservoir pressure, temperature, and oil properties, can lead to inaccurate predictions when these equations are applied outside their original context[27]. Consequently, in intricate scenarios, their effectiveness plummets, frequently leading to erroneous forecasts and suboptimal strategic choices[28,29].

Moving beyond the deficiencies of empirical equations, the advent of physical modeling and simulation methods has established a more resilient framework for forecasting oil production rates[30–32]. Leveraging specialized software, these simulation-driven approaches develop models of oil well and reservoir conditions. This is accomplished by weaving in diverse data, including geological structures, reservoir characteristics, and crucial physical parameters like pressure and temperature[28,33]. While these models offer potential for higher accuracy, their effectiveness is often hindered by uncertainties in the input data and the complexity of the models themselves[24,28,34]. Additionally, the challenges in collecting precise and extensive datasets, together with the sophistication of simulations, can result in low-accuracy or unreliable predictions, especially in cases involving highly complex reservoirs[35,36].

This study addresses the vital need for accurate predictions of choke flow performance to enhance crude oil production efficiency. Using a robust dataset from a Middle Eastern oil production site, the workflow begins with detailed data preprocessing to ensure reliability, including outlier detection via the MCOD algorithm. Key parameters, such as choke size, THP, GOR, BS&W, and oil API, were analyzed for statistical metrics and correlations. GBM models were developed and optimized using advanced algorithms like SADE and ES, with K-fold cross-validation employed to ensure model generalization. Performance evaluation, using metrics such as R^2, MSE, and AARE%, shows SADE's superior predictive accuracy. SHAP analysis further interprets the model, revealing choke size and THP as the most influential factors. Finally, visualizations including raincloud plots, cross-plots, and SHAP diagrams illustrate the robustness and interpretability of the developed framework. Figure 1 is the schematic workflow implemented in this research.

Our research offers several key contributions to choke flow rate prediction in sub-critical oil wells. First, we present a cost-effective methodology that accurately predicts flow rates using only readily available wellhead data, removing the need for expensive downhole sensors or complex fluid characterization. Second, we provide a comprehensive comparative analysis of advanced metaheuristic-optimized machine learning algorithms (SADE, ES, PI, BBO), identifying the most robust and accurate model for this specific application. Third, we emphasize enhanced generalization and reliability, rigorously evaluating models on unseen data to ensure high predictive accuracy for real-world deployment. Finally, our developed tool offers practical industry applicability as a low-computational-cost solution that integrates seamlessly into existing field monitoring systems, advancing cost-effective production optimization and well management.

Methodology

Collected data analysis

This study developed and rigorously assessed its models using meticulously gathered field data from a Middle Eastern production unit. Table 1 offers a comprehensive statistical overview of this dataset, detailing key input data like wellhead pressure, choke size, gas-to-oil ratio, basic sediment and water content, and oil API, as well as the crucial production flowrate (our output variable). For each, Table 1 presents metrics like the maximum, minimum, mean, mode, kurtosis, skewness, and standard deviation. Our data-driven machine learning models were built upon a dataset of 198 distinct data points. A substantial 90% of these points were dedicated to the training and validation phases, employing a robust fivefold cross-validation strategy, while the remaining 10% was carefully set aside for an unbiased evaluation of the models' performance.

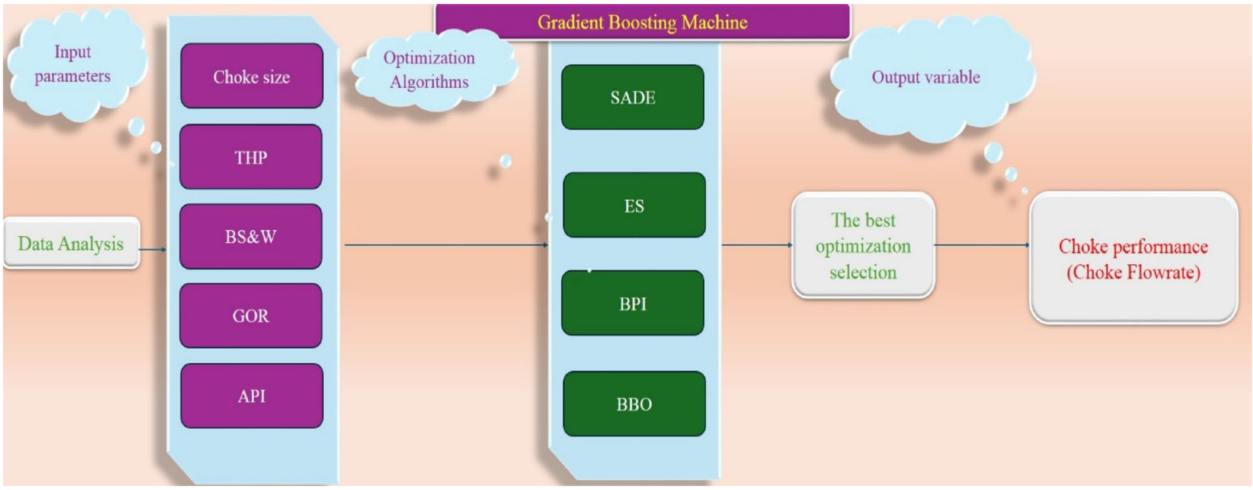

Fig. 1. Schematic of the implemented methodology.

Parameters	Minimum	Maximum	Median	Average	Kurtosis	Skewness	Standard deviation
Choke Size (1/64")	12.00	74.00	38.00	38.95	0.19	0.22	10.80
THP (psi)	0.00	1141.30	419.74	432.26	4.75	1.24	141.31
GOR (scf/stb)	156.35	1888.09	348.74	494.13	4.56	2.08	314.40
BS&W (%)	0.02	65	0.9	4.60	12.96	3.35	9.52
Oil API	9.5	35.68	18.86	19.11	11.34	2.53	3.69
Oil Rate (bbl/day)	528.30	4000	1513.20	1664.35	0.45	0.96	714.22

Table 1. Statistic-based information relevant to studied data.

This study aims to accurately estimate choke flow performance, defined as the oil production rate through wellhead chokes, and considers it the primary output variable in the model. The estimation is derived from a series of input factors. To enhance the understanding of the distribution, variability, and relationships between these input factors and choke flow performance, scatter matrix diagrams have been constructed and presented in Fig. 2. These diagrams offer a comprehensive visualization of the dataset, highlighting trends, correlations, and potential outliers, which are crucial for analyzing the data's underlying structure and facilitating the development of a reliable predictive model. Furthermore, Fig. 3 displays raincloud plots for each variable, offering additional insights into their distribution and characteristics.

Gradient boosting machine algorithm

Introduced by Jerome Friedman in 1999, GBM are a powerful supervised ensemble learning approach. They construct a highly accurate predictive model by incrementally combining numerous decision trees. This iterative methodology specifically aims to correct the shortcomings of previous trees and minimize a predefined loss function, thereby significantly boosting predictive accuracy across both classification and regression problems[37,38]. GBM excels at managing complex, non-linear relationships and offers valuable insights into feature importance, aiding feature selection. However, its computational demands and the necessity for precise hyperparameter tuning (e.g., learning rate, tree depth) to mitigate overfitting and underfitting present notable challenges.

GBM employs an iterative, stage-by-stage methodology. It begins by fitting a simple model, such as a decision tree, to the dataset. The fundamental aim is to optimize the loss function, L(y,f(x)), where y represents the true value, f(x) the model's output, and L the chosen loss metric. With each iteration, the model undergoes incremental updates designed to progressively reduce existing errors and sharpen its predictions[39,40].

The first stage, which makes the first iteration is as below

$$F_0(x) = \arg\min_{c} \sum_{i=1}^{n} L(y_i, C)$$

where $F_0(x)$ is the initial constant prediction; $L(y_i,C)$ is the Loss function; y is the true target value; C is the weight or contribution of the weak learner.

The second step, for each iteration m, is as follows: For each data point i, determine the "pseudo-residuals" by taking the negative gradient of the loss function with respect to the model's current predictions. This effectively quantifies the error direction for that specific observation.

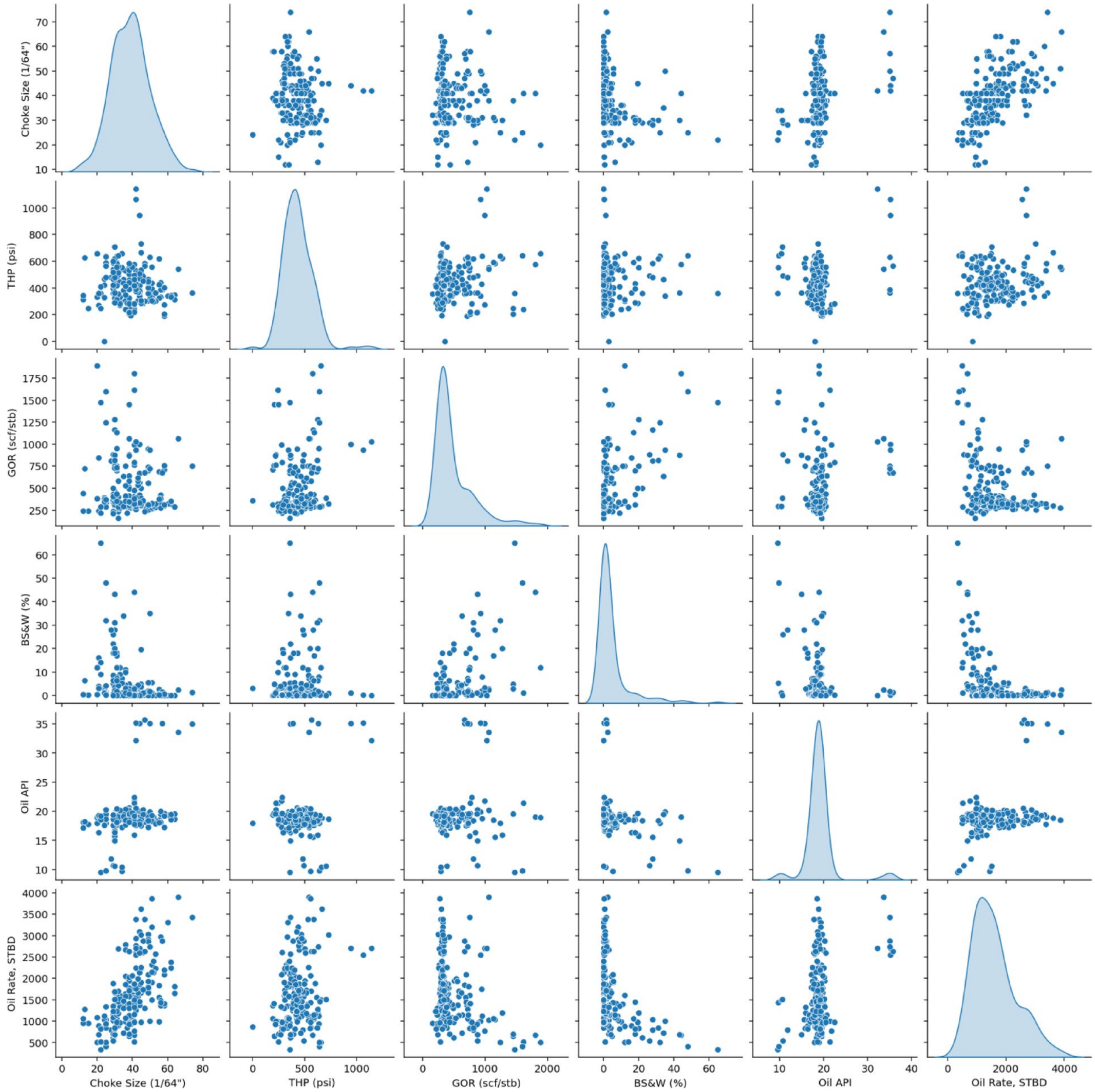

Fig. 2. Scatter matrix diagram: Relationships between variables.

$$r_{im} = -\left[\frac{\partial L(y_i, F(x_i))}{\partial F(x_i)}\right]_{F(x)=F_{m-1}(x)} \tag{1}$$

where $F_{m-1}(x)$ is the current model prediction; r_{im} is the pseudo-residuals of data poin i at the iteration step of m.

Next, a base learner is fitted to these residuals:

$$h_m(x) = \arg\min_h \sum_{i=1}^{n} \left(r_{im} - h(x_i)\right)^2 \tag{2}$$

In the fourth stage, the model is refined:

$$F_m(x) = F_{m-1}(x) + \upsilon.h_m(x) \tag{3}$$

The learning rate (υ) in the GBM algorithm dictates how much each successively added decision tree contributes to the overall model's correction. Iteratively, the algorithm introduces a new decision tree with the explicit purpose of mitigating the inaccuracies inherited from the combined output of all previous trees. GBM's inherent

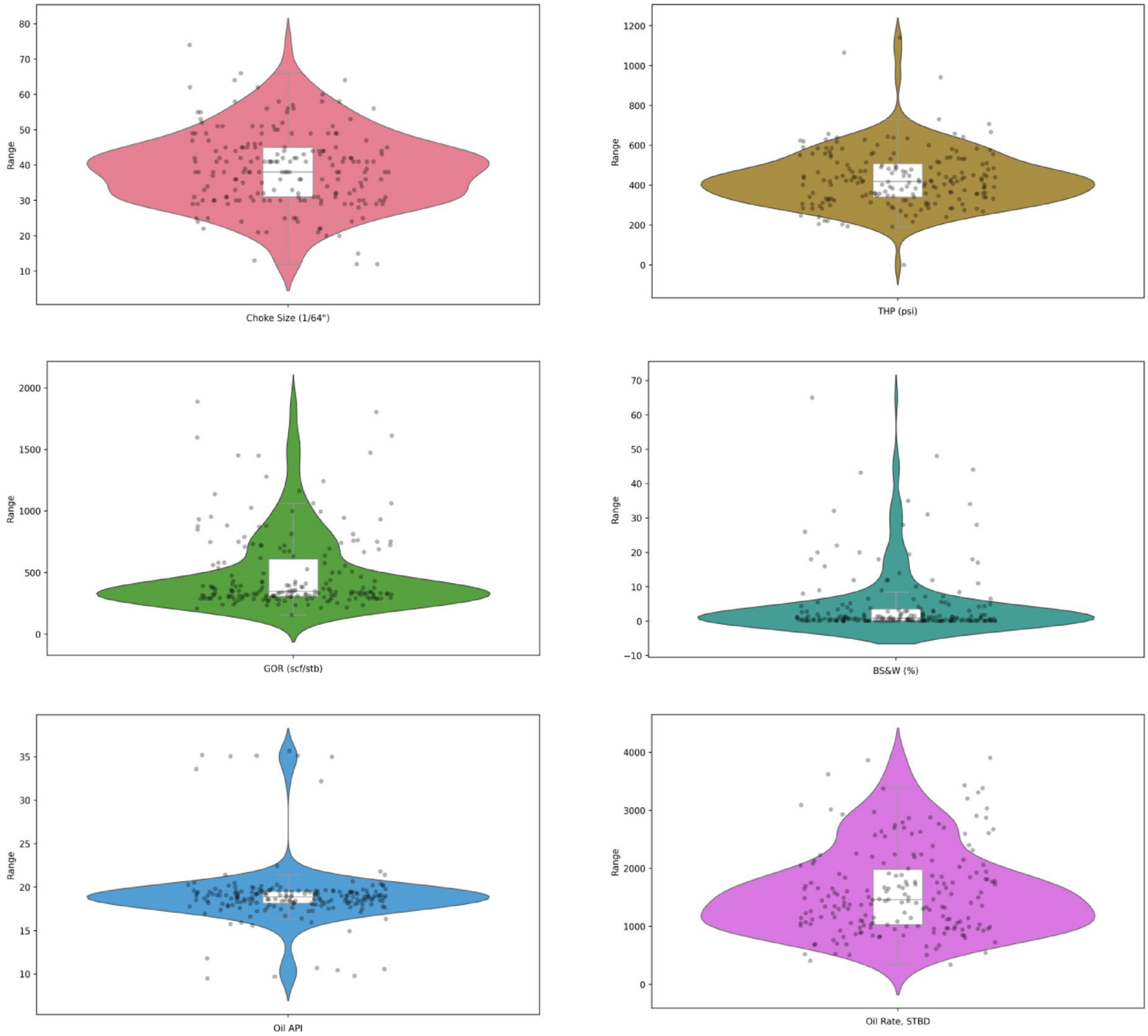

Fig. 3. Raincloud plots of all the input variables for choke flow performance modeling.

flexibility is underscored by its capacity to customize the loss function (L), enabling it to align perfectly with specific problem goals and thus proving to be a highly versatile method. To improve the model's generalization capabilities and prevent it from simply memorizing the training data, regularization techniques—specifically adjusting the learning rate (υ) and limiting tree depth—are applied. The culmination of this process is a final classifier that represents a weighted sum of predictions from its constituent trees, allowing it to gradually uncover deeper, more complex relationships within the data[41–43]. A schematic representation of the GBM manner is provided in Fig. 4.

Optimizations algorithms

Self-adaptive differential evolution (SADE)

SADE represents a significant enhancement over the conventional Differential Evolution (DE) algorithm. Unlike standard DE, SADE employs an adaptive mechanism that dynamically adjusts its control parameters throughout the optimization process. This adaptive approach has proven particularly effective for solving continuous optimization problems, demonstrating notable success in addressing complex, multimodal, and high-dimensional scenarios[44,45]. Key parameters, including the scaling factor (F) and crossover rate (CR), are automatically modified by SADE, eliminating the need for manual parameter tuning and enhancing the algorithm's robustness and efficacy[46].

The central concept of SADE lies in its ability to evolve a population of candidate solutions over successive generations while simultaneously self-adjusting control parameters based on their historical performance. This self-adaptive mechanism enables the algorithm to effectively balance exploration and exploitation within the search space, thereby improving convergence properties and yielding higher-quality solutions[47,48].

Population initialization: The optimization begins with the creation of a population containing NP candidate solutions, denoted as $X=[x_1, x_2, \ldots, x_{NP}]$, where each x_i represents an individual solution within the search space. Each solution is structured as a D-dimensional vector, where D corresponds to the number of variables

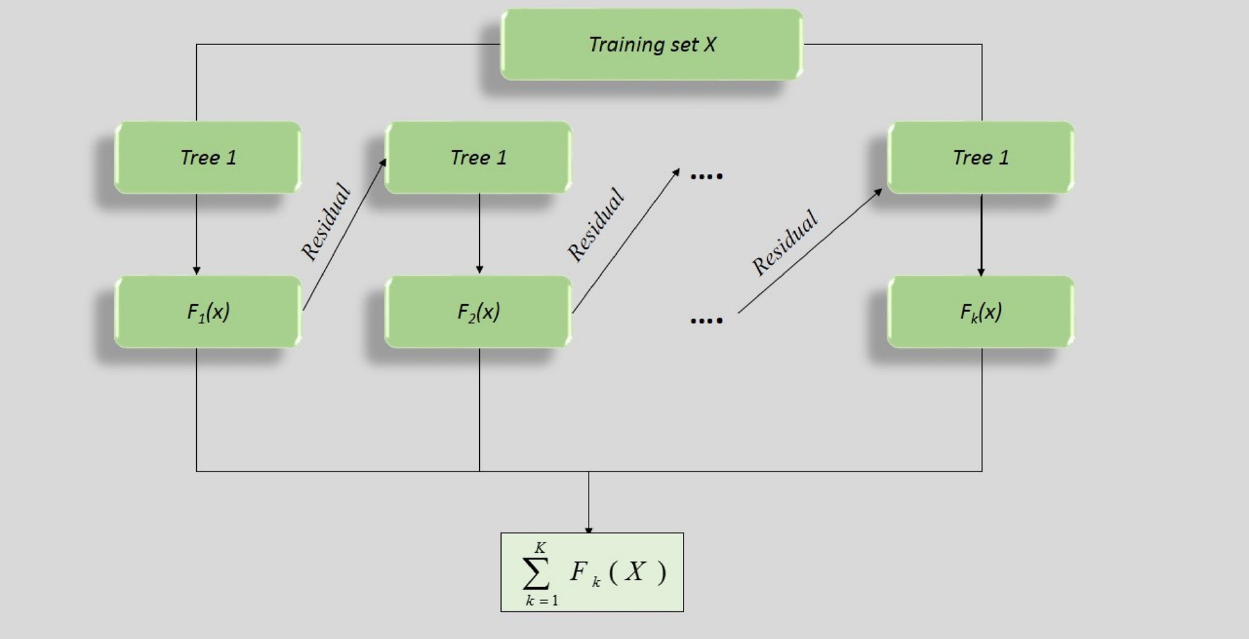

Fig. 4. Gradient Boosting Machine algorithm flowchart.

in the optimization problem. The initial population is typically generated randomly, ensuring that each solution adheres to the predefined boundaries of the search space.

Mutation: generating mutant vectors: For each individual solution x_i in the population, a mutant vector v_i is generated using a designated mutation strategy. A commonly used strategy is the "DE/rand/1" method, defined as:

$$\mathrm{v_i} = \mathrm{x_{r1}} + \mathrm{F_i} \cdot (\mathrm{x_{r2}} - \mathrm{x_{r3}}) \tag{4}$$

Here, $x_{r1,}$ x_{r2} and x_{r3} are three distinct, randomly selected individuals from the current population, while *Fi* is the scaling factor associated with *xi*. The scaling factor *Fi* determines the magnitude of the differential variation and is crucial for balancing exploration and exploitation of the search space.

Crossover: After creating the mutant vector vi, a trial vector ui is then formed. This is done by combining elements from both the mutant vector vi and the target vector xi using a crossover operation. The crossover is defined as:

$$u_{ij} = \begin{cases} v_{ij} & if\ rand(0,1) \le CR_i \text{ or } j = j_{rand} \\ x_{ij} & \text{otherwise} \end{cases} \tag{5}$$

Here, *j* represents the dimension index (j = 1, 2, …, D), rand(0,1) is a random number from a uniform distribution between 0 and 1, and CR_i is the crossover rate associated with the target vector *xi*. The index j_{rand} is a randomly selected dimension index, ensuring that at least one element in the trial vector *ui* is directly inherited from the mutant vector *vi*.

Selection: The next step involves determining whether the trial vector *ui* or the target vector *xi* will advance to the next generation. This decision is based on the values of the objective function F(0) and follows this rule[49]:

$$x_i^{new} = \begin{cases} u_i & \text{if } f(u_i) \le f(x_i) \\ x_i & \text{otherwise} \end{cases} \tag{6}$$

By comparing the fitness of u_i and x_i, the vector with the better performance is selected for the subsequent generation.

Parameter adaptation: During the optimization process, the scaling factor Fi and crossover rate CR_i are dynamically updated to enhance algorithm performance. After each generation, these parameters are adjusted based on the success of trial vectors. Specifically, when a trial vector ui is selected $f(u_i) \le f(x_i)$, the corresponding Fi and CR_i values are deemed successful and recorded in sets S_F and S_{CR}, respectively. At the end of a generation, new parameter values are calculated as:

$$F_i^{new} = \begin{cases} randc(\mu F, 0.1) & if\ S_F \ne 0 \\ F_i & \text{otherwise} \end{cases} \tag{7}$$

$$CR_i^{new} = \begin{cases} randn\left(\mu_{CR}, 0.1\right) & \text{if } S_{CR} \ne 0 \\ CR_i & \text{otherwise} \end{cases} \tag{8}$$

Here, μ_F and μ_{CR} represent the mean values of the successful parameters stored within SF and SCR sets, respectively. The functions randc and randn generate random numbers from Cauchy and normal distributions, respectively, introducing variability into the parameter values. If SF or SCR are empty, parameters will not be update[50–52].

Evolutionary strategies (ES)

ES is an optimization algorithm that mimics biological evolution to solve continuous problems. It iteratively develops a group of candidate solutions using mutation, recombination, and selection over successive generations. Each solution is represented by (x,σ), where x is the solution itself and σ dictates the magnitude of mutation for each component of x.[53].

Initialization: The ES process begins by initializing a population of μ individuals. The solution vectors (x) are typically sampled randomly within the defined search bounds of the optimization problem, while the strategy parameters (σ) are initialized with small, positive values. For instance, the solution and strategy parameters can be initialized as follows:

$$x_i \sim Uniform(x_i^{\min}, x_i^{\max}), \quad \sigma_i \sim \text{Uniform}(\sigma^{\min},\sigma^{\max}) \tag{9}$$

Here, $x_i^{\min}$ and $x_i^{\max}$ define the valid range for the solution vector, while $\sigma^{\min}$ and $\sigma^{\max}$ establish the range for the strategy parameters.

Mutation and offspring generation: In each generation, λ offspring are created through the mutation process. Mutation operates by modifying both the solution vector x and its associated strategy parameters σ. Updated strategy parameters (denoted as σ') are computed using a predefined mutation rule, which introduces variability into the mutation scale. This adaptive mechanism allows the algorithm to navigate complex search landscapes effectively. The mutation process ensures robust exploration while maintaining the ability to refine promising solutions over successive generations.

$$\overset{\prime}{\underset{i}{\sigma}} = \sigma_i \times \exp\left(\tau \times N\left(0.1\right) + \overset{\prime}{\tau} \times N_i\left(0.1\right)\right) \tag{9}$$

In this context, $N(0,1)$ represents a random variable drawn from a standard normal distribution, while $N_i(0,1)$ denotes an independent random variable sampled separately for each dimension. The parameters τ and τ' are learning rates, often chosen based on the dimensionality of the problem[54]:

$$\overset{\prime}{\underset{i}{x}} = x_i + \overset{\prime}{\underset{i}{\sigma}} N_i\left(0.1\right) \tag{10}$$

Recombination, when utilized, integrates information from multiple parent solutions to produce offspring. In the case of intermediate recombination, the offspring (x',σ') is generated as a weighted average of the μ parent solutions. The process can be mathematically expressed as:

$$\overset{\prime}{\underset{i}{x}} = \frac{1}{\mu}\sum_{j=1}^{\mu} x_{i.j} \tag{11}$$

$$\overset{\prime}{\underset{i}{\sigma}} = \frac{1}{\mu}\sum_{j=1}^{\mu} \sigma_{i.j} \tag{12}$$

Selection phase: Following the generation of offspring, the selection phase determines the individuals that will proceed to the next iteration. In the (μ + λ)-ES strategy, the top μ individuals are selected from the combined pool of μ parents and λ offspring. Alternatively, in the (μ,λ)-ES strategy, the selection occurs exclusively among the λ offspring, with λ ≥ μ. This approach ensures that the population evolves progressively toward better solutions with each generation.

One of the key strengths of Evolution Strategies (ES) lies in its self-adaptive mechanism, which enables the dynamic adjustment of strategy parameters (σ) throughout the optimization process. This self-adaptation effectively balances exploration (broad search of the solution space) with exploitation (refining optimal regions), making the method particularly effective for solving complex and high-dimensional optimization problems. Additionally, ES can be further enhanced with advanced techniques, such as Covariance Matrix Adaptation (CMA-ES), which improves the optimization process by adapting the covariance matrix of the search distribution[55–57].

In summary, Evolution Strategies evolve a population of candidate solutions by iteratively applying mutation, recombination, and selection. The dynamic adjustment of strategy parameters enhances the robustness and versatility of the algorithm for continuous optimization problems[58,59]. The general procedure involves the following steps: initialization of the population, offspring generation through mutation and recombination, fitness evaluation, selection of the best individuals, and iteration until a termination criterion is met. The best solution obtained during this iterative process is presented as the final result[60].

Bayesian probability improvement (BPI)

Bayesian Probability Improvement (BPI) represents a specialized optimization technique commonly employed within the Bayesian optimization framework. Bayesian optimization is a global optimization strategy designed for expensive black-box functions, offering particular advantages when evaluating the objective function is computationally costly, as it aims to identify the optimum with minimal function evaluations. Within this framework, BPI serves as an acquisition function that guides the optimization process by effectively balancing exploration of uncertain regions with exploitation of known promising areas[61–63].

The fundamental objective of BPI is to maximize the probability that a newly selected candidate point will yield an improved objective function value compared to the current best observation. Unlike alternative acquisition functions like Expected Improvement (EI) or Upper Confidence Bound (UCB), BPI adopts a distinctly probabilistic perspective by directly quantifying the probability of improvement. This approach proves especially valuable in scenarios where increasing the likelihood of discovering a better solution takes precedence over maximizing the expected magnitude of improvement[64].

At its core, BPI calculates the probability that the objective function amount at a new point x will surpass the current best-observed value, denoted as $f(x+)$. This calculation utilizes the posterior distribution provided by a Gaussian Process (GP) model, which is widely employed in Bayesian optimization frameworks. The GP model provides a predicted mean $\mu(x)$ and variance $\sigma^2(x)$ for any location x in the search space. The probability of improvement is then derived from the cumulative distribution function (CDF) of the normal distribution[65].

The BPI is mathematically formulated as:

$$BPI\left(x\right)=\varnothing\left(\frac{\mu\left(x\right)-f(x^{+})}{\sigma(x)}\right) \tag{13}$$

where, Φ represents the cumulative distribution function of the standard normal distribution, $\mu(x)$ denotes the predicted mean of the GP at point x, $\sigma(x)$ represents the predicted standard deviation of the GP at point x, $f(x^{+})$ refers to the best objective function value observed thus far.

The BPI acquisition function is optimized to determine the next point for evaluation. Higher BPI values indicate a greater probability that the new point will improve upon the current best solution. This approach is particularly efficient when the primary goal is to ensure consistent progress with each evaluation, as it focuses on the likelihood of improvement rather than the potential magnitude of improvement[66,67].

In conclusion, BPI represents a probabilistic acquisition function that leverages the predictive uncertainty quantified by a Gaussian Process to guide the search for optimal solutions. By prioritizing the probability of improvement, BPI provides a valuable tool within the Bayesian optimization framework, particularly when function evaluations are computationally expensive and the objective is to maximize the chances of finding superior solutions[68].

Batch bayesian optimization (BBO)

BBO extends the traditional Bayesian optimization framework to enable the simultaneous evaluation of multiple points, referred to as a "batch," rather than evaluating points sequentially. This adaptation is particularly advantageous in environments where parallel computing resources, such as high-performance computing clusters or distributed systems, are available. By leveraging parallelism, BBO seeks to significantly reduce the total optimization time while preserving the ability of Bayesian optimization to efficiently identify the global optimum[69,70].

The primary innovation in BBO is the selection of a group of points for simultaneous evaluation instead of iteratively selecting individual points. This requires modifying the acquisition function to account for the interdependencies and correlations among points within the batch, as the evaluation of one point may provide information that affects the value of others in the group.

BBO commonly uses acquisition functions that balance exploring uncertain regions with exploiting promising areas within a batch. A popular method is Parallel Expected Improvement (q-EI), which extends the standard Expected Improvement (EI) function to evaluate multiple points at once. The q-EI function calculates the expected improvement beyond the current best value for a batch of q points, factoring in their correlations as predicted by the Gaussian Process (GP). The formula for q-EI is[71]:

$$q-EI\left(x\right)=E[maxf\left(x\right)-f\left(x^{+}\right).0] \tag{14}$$

where, $X=[x_1,x_2,\ldots,x_q]$ represents the batch of q points selected for evaluation, f(X) denotes the objective function values at the batch points, f(x+) is the current best-observed value of the objective function, E denotes the expectation taken over the joint posterior distribution of the GP at the batch points.

By explicitly considering the joint distribution over the batch, q-EI ensures that the selected points maximize the expected improvement collectively rather than in isolation[72].

BBO) enhances the efficiency of traditional Bayesian optimization by enabling simultaneous evaluation of multiple points, thus reducing the total optimization time. The use of specially adapted acquisition functions, such as q-EI and Thompson Sampling, ensures that the optimization process continues to balance exploration and exploitation effectively. BBO is especially beneficial in scenarios where parallel computational resources are available, as it accelerates the optimization process while maintaining the high performance of Bayesian optimization in locating the global optimum[73].

Models evaluation

A prevalent and powerful strategy for assessing the generalization capability of machine learning models is K-fold cross-validation. This method meticulously divides the complete dataset into K discrete and uniformly sized partitions, known as folds. In each iteration of the process, the model is trained on K-1 folds while the remaining fold is used to evaluate its performance. This approach is renowned for its straightforward implementation and ability to produce consistent, reliable results, making it a preferred choice among various cross-validation techniques. Its simplicity and balanced approach ensure its applicability across a broad range of machine learning tasks[55–57].

To grasp the mechanics of K-fold cross-validation, let's consider a scenario where K = 5 (as visually depicted in Fig. 5). The entire dataset is systematically segmented into five equally sized partitions, with each segment designated as a "fold." The validation process then unfolds iteratively: In the initial round, four of these folds collectively serve as the training data for the model, while the first fold is reserved exclusively for testing its performance. This pattern continues: in the subsequent iteration, the second fold becomes the dedicated test set, with the remaining four constituting the training material. This meticulous rotational scheme ensures that every one of the five folds functions precisely once as the test set, leading to a more robust and unbiased assessment of the model's generalization capabilities[74,75].

To assess the accuracy and effectiveness of each model, several key performance indicators were calculated, including RE%, AARE%, MSE, and R^2. Detailed explanations of each metric are presented in the following sections[15,76–78]:

$$RE\% = \left(\frac{V^{pred} - V^{\exp}}{V^{\exp}}\right) \times 100 \tag{10}$$

$$AARE\% = \frac{100}{N}\sum_{i=1}^{N}\left(\left|\frac{V_i^{pred} - V_i^{\exp}}{V_i^{\exp}}\right|\right) \tag{11}$$

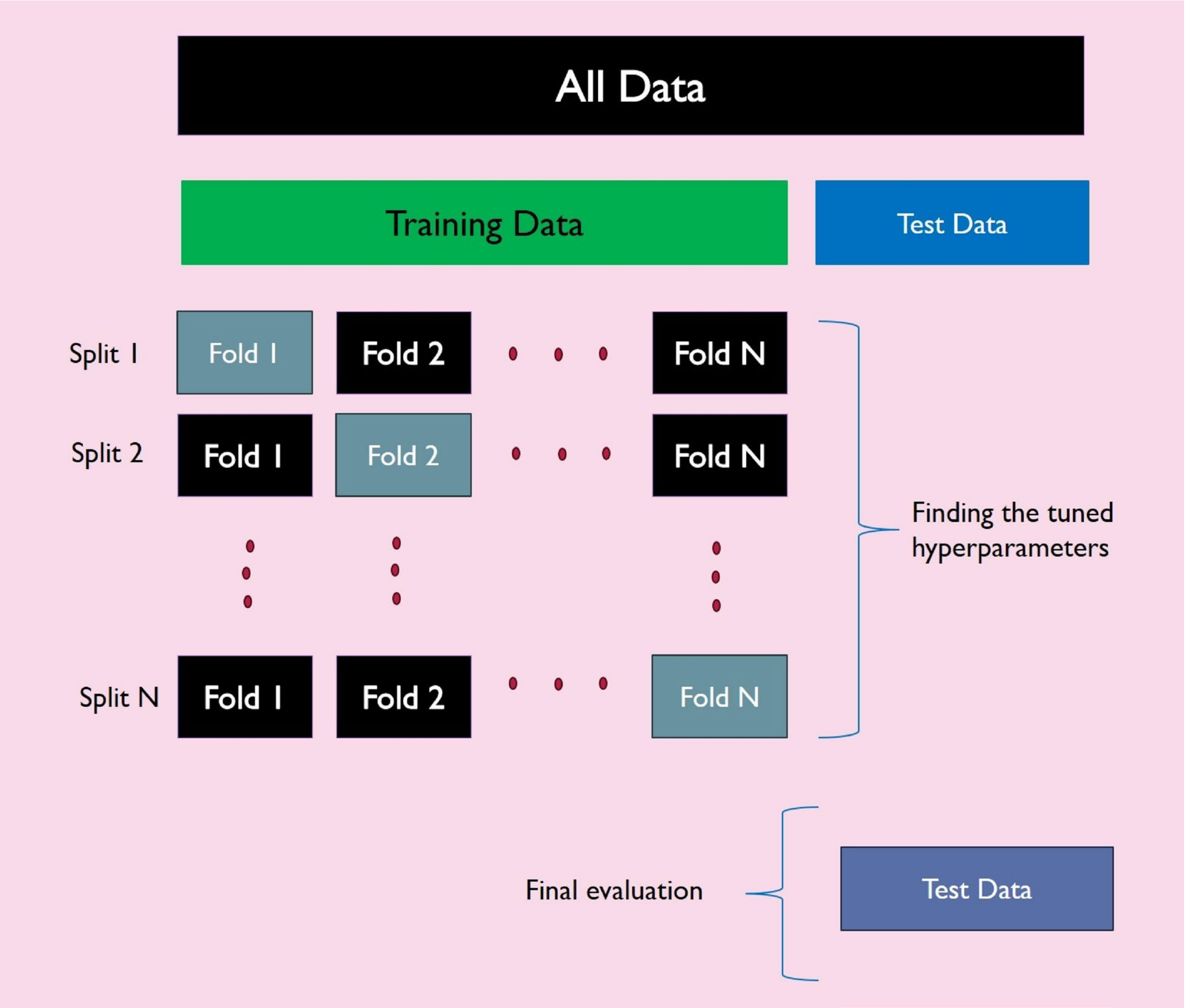

Fig. 5. K-fold cross-validation Algorithm schematic.

$$MSE = \frac{\sum_{i=1}^{N} \left(V_i^{pred} - V_i^{\exp}\right)^2}{N} \tag{12}$$

$$R^2 = 1 - \frac{\sum_{i=1}^{N} \left(V_i^{pred} - V_i^{\exp}\right)^2}{\sum_{i=1}^{N} \left(V_i^{\exp} - \overline{V}\right)^2} \tag{13}$$

Within these equations, the subscript i serves to identify a unique data point from our entire dataset. For any given i, 'pred' signifies the value projected by our model, while 'exp' refers to the corresponding actual or experimentally observed value. Furthermore, N consistently represents the grand total data points.

Results and discussion

Comprehensive data characterization

Prior to developing machine learning models for predicting choke flow performance, ensuring data reliability through outlier management is paramount. This research implemented the Monte Carlo Outlier Detection (MCOD) algorithm, a robust technique for identifying outliers in large datasets using random sampling and density-based approaches. MCOD assesses local data density to pinpoint data points that deviate significantly from their neighbors. By employing Monte Carlo sampling to estimate data subsets, the algorithm reduces computational burden. Its effectiveness and scalability render it suitable for high-dimensional datasets and real-time applications. There's a natural give-and-take between how accurate the MCOD method is and how fast it can compute results. This balance is influenced by the sample size and the number of nearest neighbors (k) you choose. Even with this trade-off, MCOD is incredibly useful for initial data exploration and finding anomalies, especially when you don't need perfect precision or when you're working with limited computing power. Its ability to balance accuracy and efficiency makes it a great tool for spotting outliers in complicated datasets.

A boxplot in Fig. 6 illustrates our dataset's distribution and acceptable range, with most data points falling within this range, indicating high data quality. To maximize the models' ability to generalize, the complete collected dataset was used for training. This comprehensive approach allows the models to discern underlying patterns effectively, resulting in more reliable and accurate predictions on unseen data.

Models' optimization

This section focuses on how we applied various optimization algorithms to improve the Gradient Boosting Machine (GBM). Our goal was to fine-tune GBM's performance by optimizing its hyperparameters, which we did both by directly applying optimization techniques and by using them with k-fold cross-validation. The specific hyperparameters we worked on were maximum depth, the number of estimators, minimum samples and maximum features needed for splitting, the learning rate, subsample size, and minimum samples required for leaves. Table 2 provides a complete overview of the parameter ranges we have explored and the best values identified by each optimization algorithm.

Figure 7 visualizes the Mean Squared Error (MSE) progression for each optimization algorithm over 200 iterations, identifying the best hyperparameter configurations (also in Table 2). Figure 8 then compares their computational times, which were measured on a machine with Intel Core i7-6700 CPU (3.40 GHz) and 16 GB RAM. The SADE method was the slowest at roughly 4200 s, whereas the ES algorithm was the fastest.

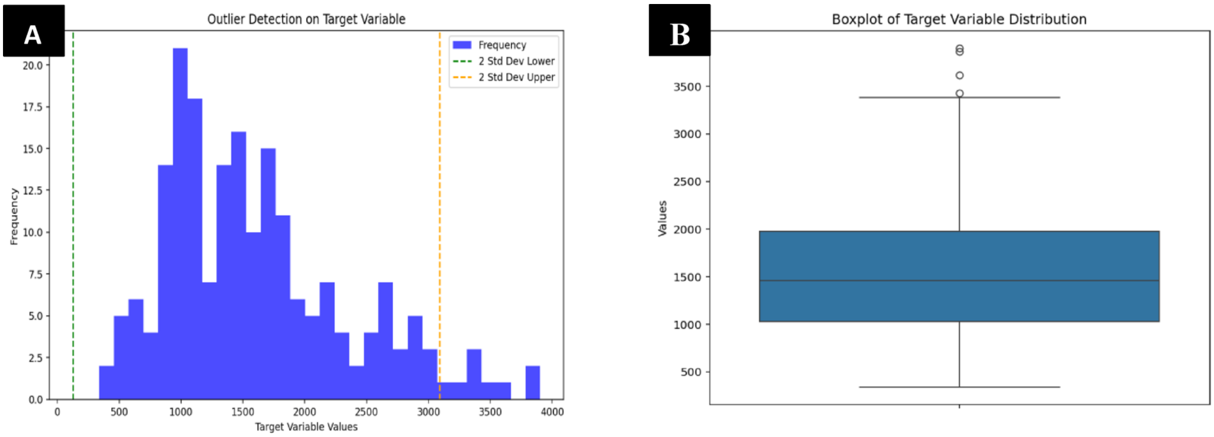

Fig. 6. (**A**) Detection of outliers using the MCOD algorithm and (**B**) Boxplot depicting the distribution of the dataset.

Tuning parameter	Considered range	SADE	ES	BPI	BBO
n_estimators	[50–300]	223	159	247	230
max_depth	[5–20]	5	15	14	12
max_features	[0.1–1]	0.7146	0.2846	0.2221	0.2595
min_samples_split	[0.01–0.5]	0.2850	0.3593	0.2522	0.3079
learning_rate	[0.01–0.3]	0.0191	0.0335	0.0339	0.0359
subsample	[0.5–1]	0.6535	0.9836	0.6462	0.7706
min_samples_leaf	[0.01–0.5]	3.0000	0.0173	0.0206	0.0191

Table 2. Evaluating optimal gradient boosting hyperparameters using four optimization techniques within a specified parameter range.

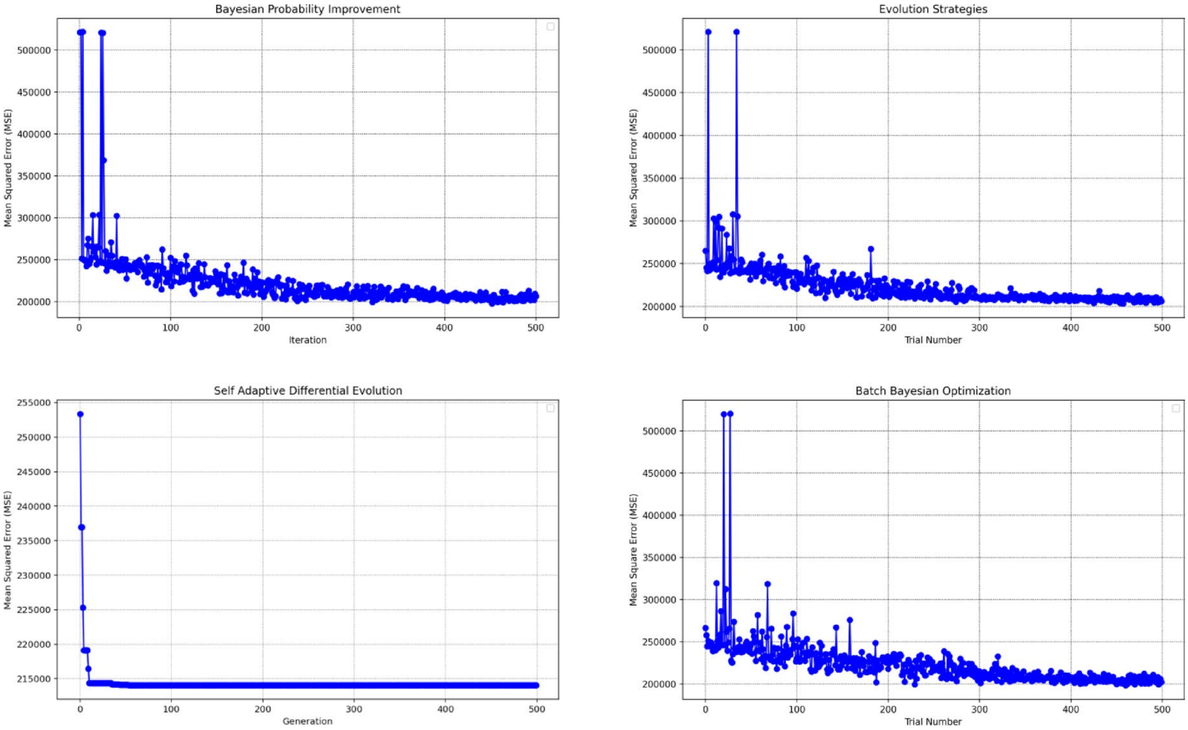

Fig. 7. Iteration-wise MSE for different optimization methods.

Figure 8 clearly shows a stark difference in optimization progression: the SADE algorithm reached its peak performance—the lowest MSE—quite rapidly in the initial generations. Conversely, BPI, ES, and BBO algorithms demonstrated a more deliberate and gradual enhancement in their performance as the iterations continued.

Table 3 provides a comprehensive comparison of hybrid optimization approaches using the GBM estimator and Fig. 9 displays these metrics for testing stage per all models. The evaluation utilizes key metrics, including R^2, MSE, and AARE%, across training, testing, and total data sets. The optimization algorithms considered are SADE, ES, PI, and BBO, with a specific focus on test-phase metrics as the primary indicators of model performance.

From Table 3, it is evident that SADE exhibits the highest testing-phase accuracy among all algorithms. SADE achieves a testing R^2 value of 0.5935, indicating moderate explanatory power in predicting choke performance compared to other methods. Furthermore, SADE demonstrates superior error minimization, evident in its lowest testing MSE. This substantial reduction in error reflects the algorithm's enhanced ability to fit the data during testing. Additionally, the testing-phase AARE% for SADE is 22.59%, which, while not the lowest observed, still aligns closely with the algorithm's robust prediction performance. The consistency between training and test performance for SADE highlights its reliability for modeling.

Despite its high accuracy, SADE comes with a trade-off in computational runtime, as its complex optimization structure results in additional computational overhead. Comparatively, while ES achieves a higher training R^2 (0.8385) and a total R^2 of 0.7985, its testing R^2 is lower at 0.5794. This drop suggests a less stable generalization capability when applied to unseen data. ES also yields a testing MSE higher than SADE, which points to reduced prediction precision. Furthermore, its testing-phase AARE% of 28.62% is notably higher, indicating greater

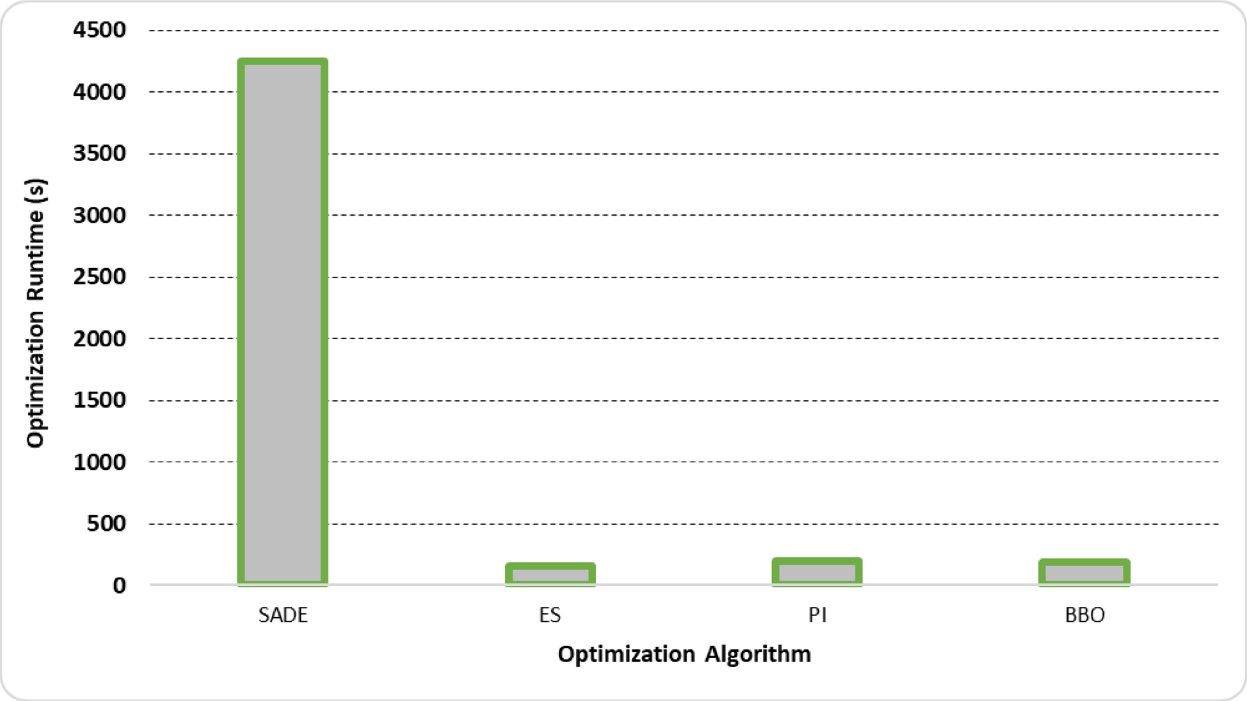

Fig. 8. Runtime comparison of optimization algorithms integrated with GBM.

Optimization algorithm	R2			MSE			AARE%		
	Training	Test	Total	Training	Test	Total	Training	Test	Total
SADE	0.8003	0.5935	0.7783	109,000	241,076.64	122,000	16.489	22.5882	17.1051
ES	0.8385	0.5794	0.7985	82,700	361,297.16	111,000	13.6684	28.6172	15.1784
PI	0.8714	0.5628	0.8234	65,800	375,541.3	97,100	12.1332	30.6041	13.999
BBO	0.8683	0.5717	0.8222	67,400	367,965.07	97,800	12.4008	30.7813	14.2574

Table 3. Efficacy benchmarking of optimization algorithms on train, test, and all datasets.

absolute errors in prediction during testing. Overall, ES demonstrates acceptable but weaker performance compared to SADE, particularly in testing metrics.

PI and BBO offer comparable results, both achieving higher testing-phase R^2 values compared to SADE, with PI slightly outperforming BBO (0.5628 vs. 0.5717). However, an examination of testing MSE reveals clear deficiencies in these algorithms compared to SADE, with PI and BBO recording testing MSE values. These errors are significantly larger than SADE's testing MSE, underscoring SADE's superior capacity to minimize prediction errors. Additionally, both PI and BBO yield high AARE% values during testing (30.60% and 30.78%, respectively), which further highlight their reduced accuracy compared to SADE. While these algorithms perform well in training (PI with the highest training R^2 of 0.8714), their weaker generalization capabilities during testing limit their overall utility.

While the SADE algorithm demands the highest computational cost (~ 4200 s), it delivers superior predictive performance, achieving the lowest testing MSE and the highest R^2 (0.5935) among all methods. In contrast, faster algorithms like ES sacrifice accuracy for efficiency, exhibiting gradual error reduction but failing to match SADE's explanatory power or error minimization. This trade-off underscores SADE's value in applications where precision is critical, despite its resource intensity, whereas ES or BBO may suffice for scenarios prioritizing rapid, approximate solutions. The 22.59% AARE% further confirms SADE's reliability, justifying its computational overhead when model robustness is paramount.

Figure 10 compellingly illustrates the enhanced precision of our proposed models, with the SADE algorithm distinctly outperforming others. Its cross-plots reveal a significantly tighter clustering of data points around the unit slope line, a clear indicator of superior accuracy. This heightened performance is further underscored by the fitted line equations in the SADE plots, which lie remarkably close to the bisector line.

Figure 11 effectively visualizes the distribution of relative deviations for each of our hybrid models. A tighter clustering of data points around the $y = 0$ line indicates greater accuracy from the estimator. Based on this, the GBM estimator tuned by SADE stands out as the most efficient predictive tool among all evaluated methods.

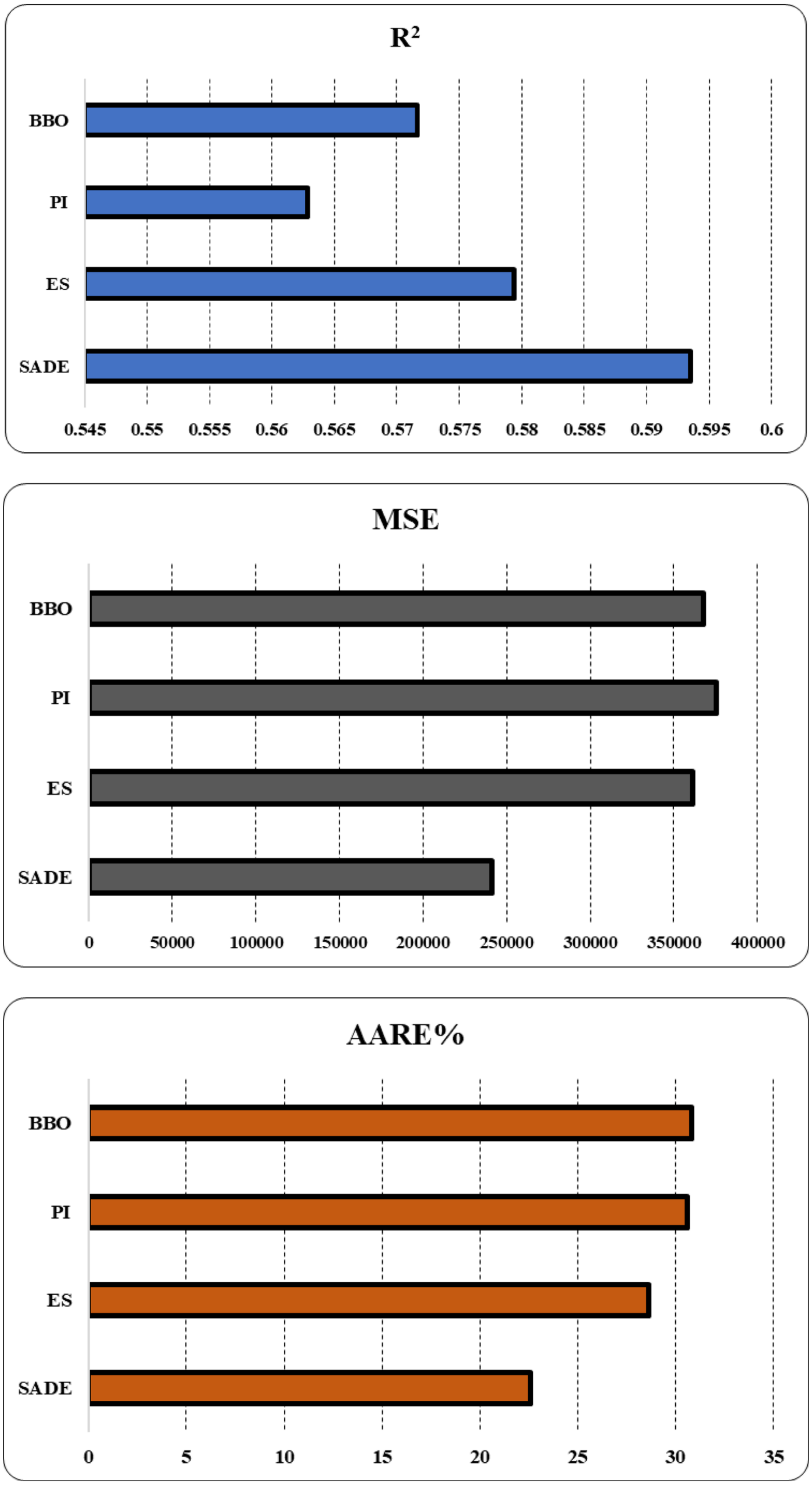

Fig. 9. Testing performance of optimization algorithms: R^2, AARE%, and MSE.

Complementing this, Fig. 12 offers a direct comparison between the estimated and real data points across all four algorithms.

Figure 13 provides a powerful look into our predictive model's decision-making process through SHAP (SHapley Additive exPlanations) analysis. This visualization clearly highlights which individual features are most

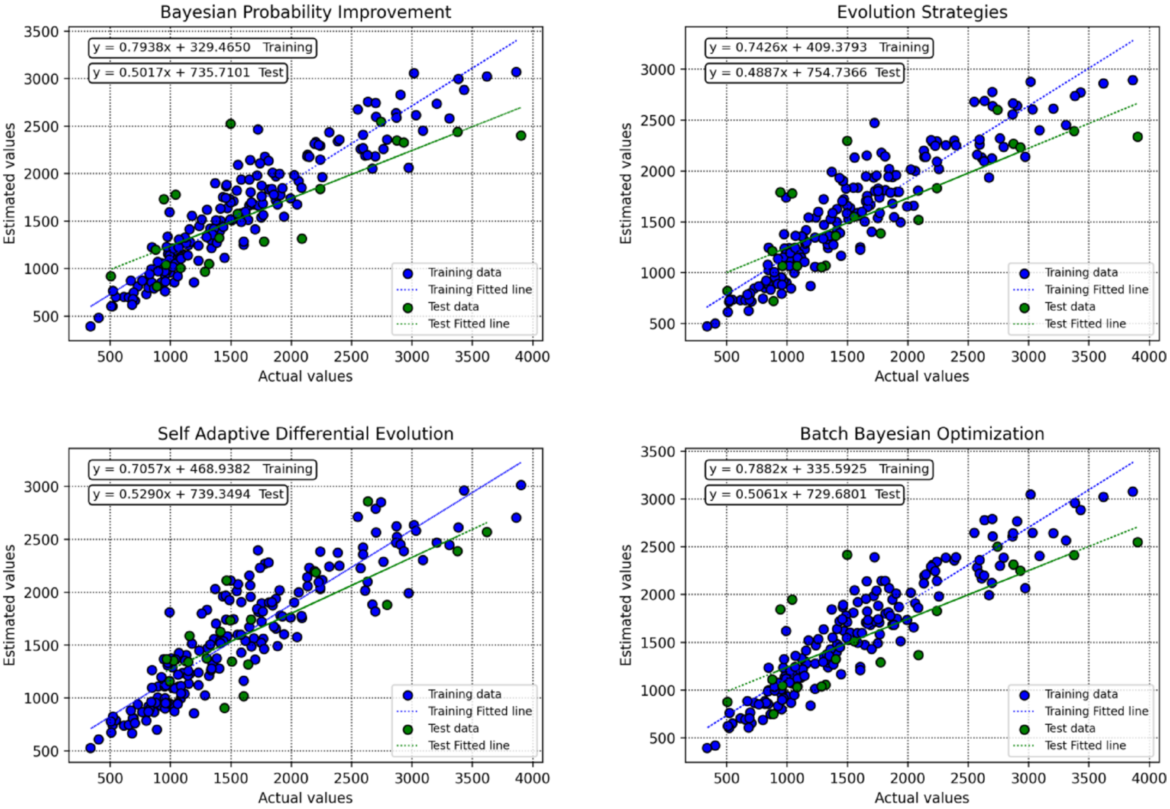

Fig. 10. Visualizing discrepancies between modeled and real data for all optimizers during train and test.

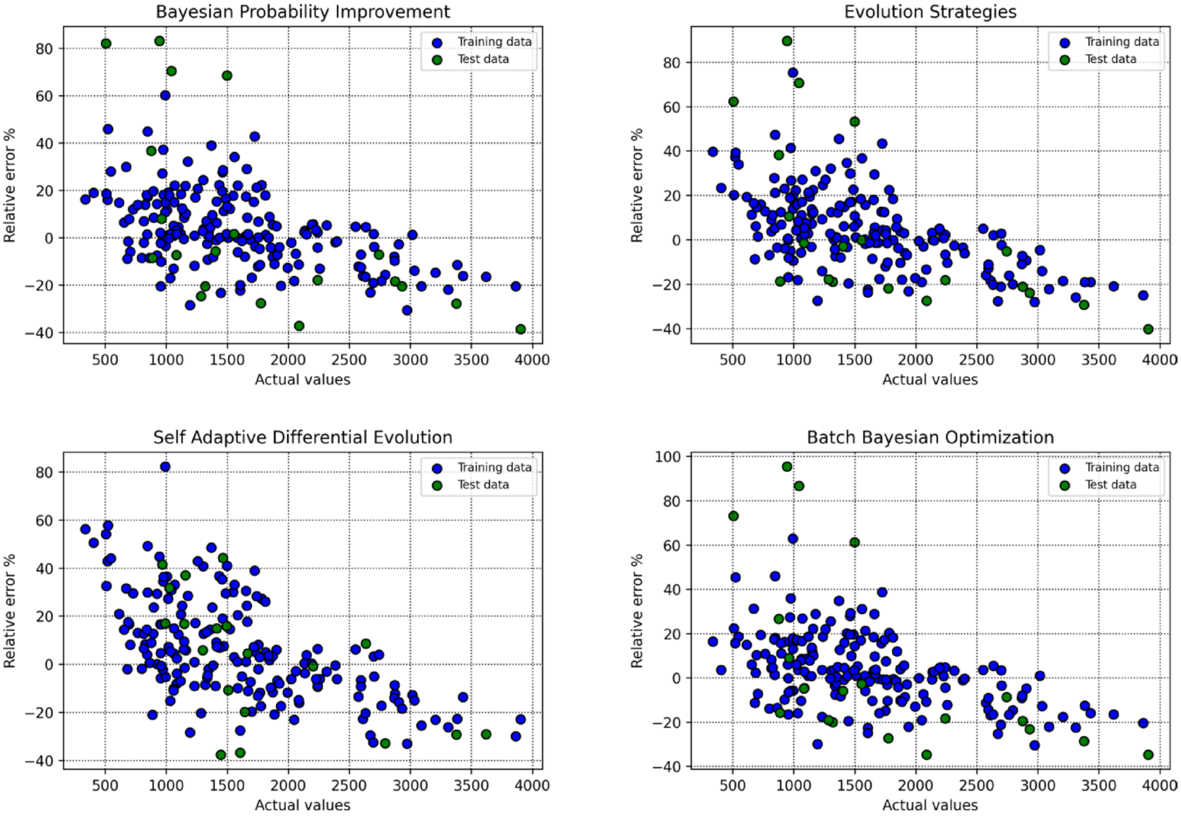

Fig. 11. RE% distribution against actual data for all optimizers (training & testing).

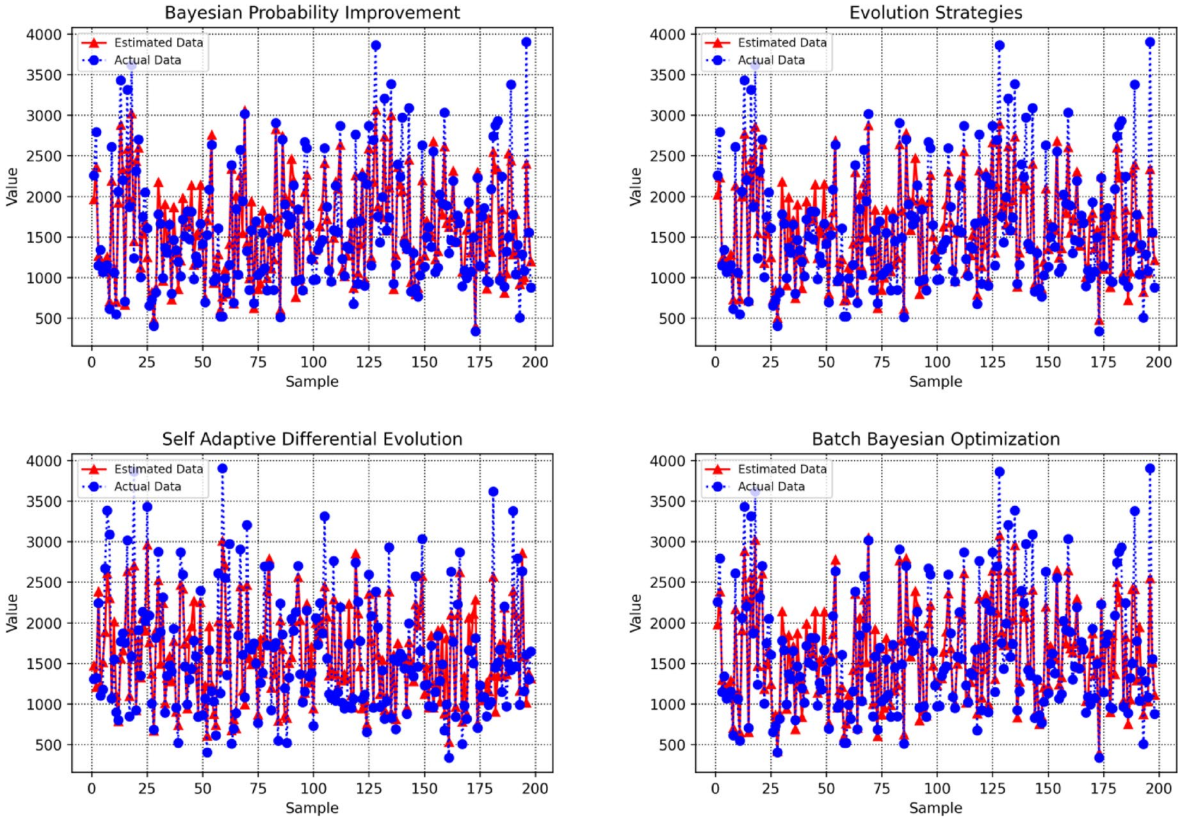

Fig. 12. Comparative plots of predicted vs. real data for all algorithms.

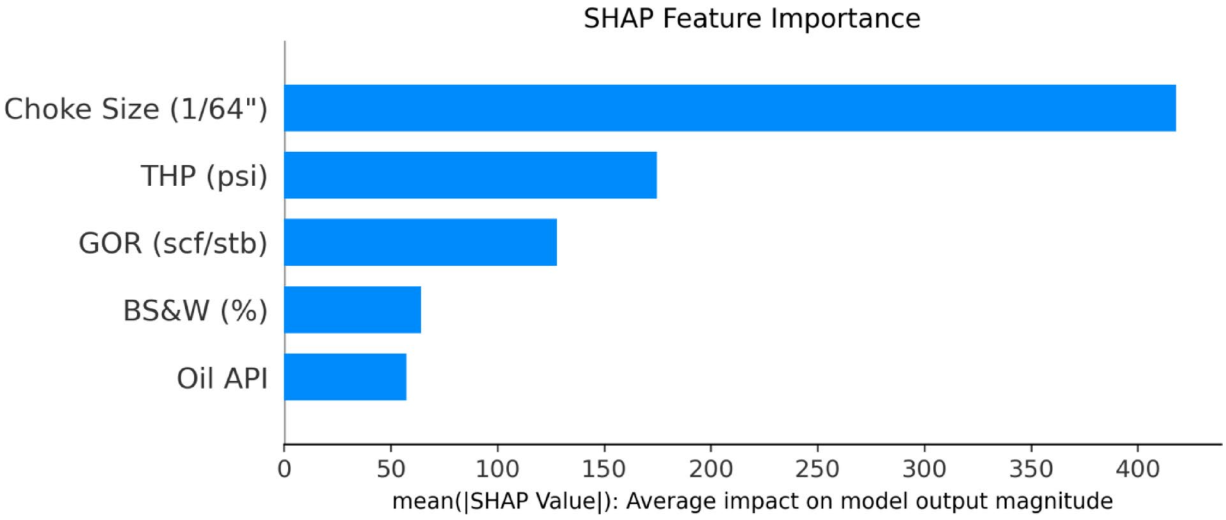

Fig. 13. SHAP feature importance.

crucial in shaping the model's output. By ranking input variables—including choke size, tubing head pressure (THP), gas-oil ratio (GOR), basic sediment and water (BS&W), and API—based on their average absolute SHAP values, we can quantify each feature's contribution to the magnitude of the model's predictions. The analysis strikingly reveals that choke size and THP have the highest mean SHAP values, solidifying their role as the primary drivers of the model's predictions and overall performance.

Figure 14 further investigates the SHAP analysis results for a choke flow performance model by demonstrating the impact of various input parameters on the model outputs. The plot visually represents how individual features affect choke flow predictions, where positive SHAP values correlate with an increase in choke flow and negative SHAP values indicate a decrease. The red points correspond to higher values for each feature. For instance, larger

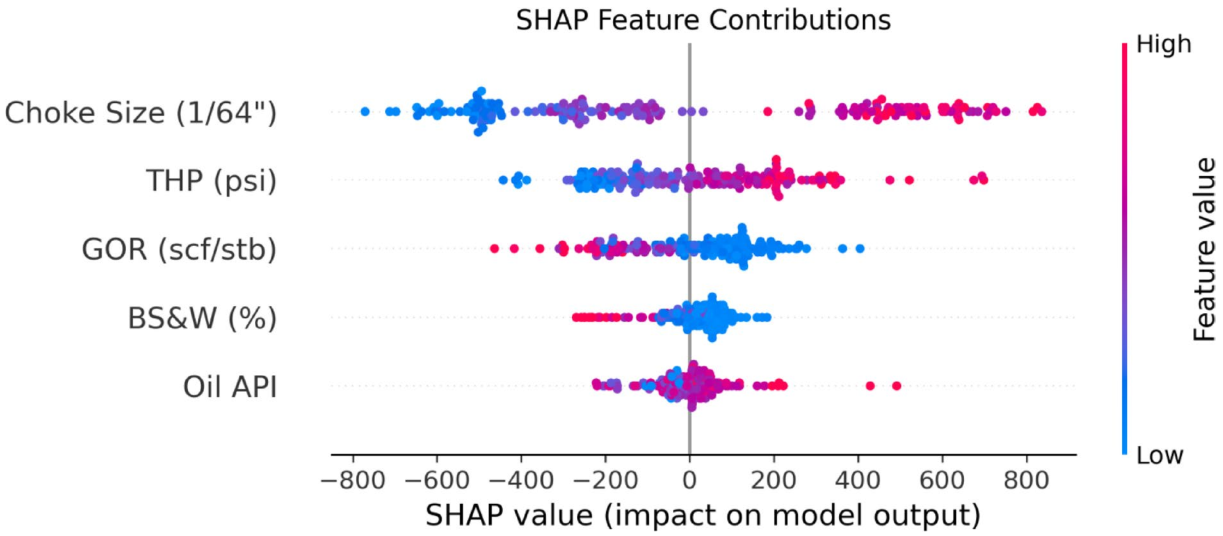

Fig. 14. SHAP feature conditions.

choke sizes are associated with elevated choke flow rates, while higher THP promotes greater oil production through the choke. Conversely, higher values of GOR and BS&W are linked to reductions in oil production or choke performance. These findings align with theoretical and practical observations from the literature.

Specifically, the SHAP results align with established principles associated with reservoir and well performance. Larger choke sizes reduce flow resistance[79,80], enabling higher production rates. Similarly, increased THP reflects stronger reservoir pressure[81,82], which facilitates greater oil production and higher flow rates. On the other hand, higher BS&W values signify poor reservoir quality, leading to increased pressure drops across the well and reduced production rates[83,84]. Excessively high GOR values not only result in increased oil flashing at the surface but also negatively impact total production rate (TPR) due to the adverse effects on oil well performance, such as reduced tubing pressure and efficiency[85]. These insights confirm the predictive relevance and practical validity of the SHAP analysis results in capturing the dynamics of choke flow performance.

The developed tools can be easily used to predict choke oil flow rate based on wellhead data. In addition, the same workflow can be applied in other fields so as to develop cost-effective, fast, reliable and accurate models.

Limitations and future work

The performance of any data-driven optimization approach is inherently tied to the quality and representativeness of the dataset. We acknowledge potential limitations stemming from data collection biases, the completeness and noise within the dataset, and whether the dataset size and diversity fully capture the problem's complexity, which could affect the generalizability of our findings. We also recognize the risk of overfitting our optimization methods to the training data. While we employed strategies like cross-validation and regularization to mitigate this, some residual risk may remain, impacting performance on unseen data. Further validation on independent or larger datasets will be crucial.

Our current optimization framework operates under specific assumptions. We critically examine model simplifications (e.g., assumed linearity), the representativeness of parameter space boundaries, and any potential limitations or biases in the objective function design that might not fully align with real-world outcomes. These identified limitations serve as a foundation for clear and impactful future research directions. These include, but are not limited to, exploring advanced validation techniques, investigating hybrid optimization approaches, addressing data sparsity or imbalance, incorporating real-time or dynamic optimization, integrating multi-objective optimization, and conducting deployment and real-world impact studies to assess the practical implementation and evaluation of our optimized solutions.

Conclusions

This study demonstrates the successful application of machine learning to predict choke flow performance, utilizing a high-quality dataset from a crude oil production site. Preprocessing techniques, including the MCOD algorithm, were instrumental in ensuring reliable data for model development. The SADE algorithm was the top performer among the optimization methods used, achieving both the lowest MSE and the highest R^2 value during the testing phase. The models developed in this study exhibited strong predictive accuracy, with SHAP analysis providing valuable insights into the relative importance of the input parameters. Specifically, choke size and THP emerged as the dominant factors influencing choke flow performance. These findings align with known physical principles, validating the models' practical relevance. This research provides a robust framework for applying data-driven approaches to production optimization, with the potential for broader applications in the oil and gas industry. Our comparative analysis demonstrates that the SADE algorithm consistently delivers the highest accuracy in predicting choke flow rate during the crucial testing phase. SADE achieved a testing R2 of 0.5935, indicating superior explanatory power compared to other methods. Crucially, SADE exhibited the lowest testing MSE, reflecting its enhanced ability to minimize prediction errors and fit unseen data accurately. While its testing-phase AARE% of 22.59% was not the absolute lowest, it aligns well with its robust predictive performance and the overall consistency observed between its training and test results, highlighting its reliability for modeling. In comparison, other algorithms demonstrated weaker generalization capabilities. For instance, ES, despite a higher training R2 of 0.8385, showed a noticeable drop in testing R2 to 0.5794, along with a higher testing MSE and an AARE% of 28.62%, indicating reduced prediction precision for unseen data. Similarly, PI and BBO, while achieving training R2 values as high as 0.8714 (PI), recorded testing R2 values of 0.5628 and 0.5717 respectively, which are comparable to SADE's R2 but come with significantly higher testing MSE values and high AARE% values of 30.60% (PI) and 30.78% (BBO). These metrics underscore their reduced accuracy and generalization when confronted with new data, limiting their overall utility compared to SADE.

Data availability

Data supporting this study's findings will be available from the corresponding author upon reasonable request.

Received: 9 March 2025; Accepted: 1 July 2025
Published online: 16 July 2025

References

1. Ma, Q., Li, H. & Li, Y. The study to improve oil recovery through the clay state change during low salinity water flooding in sandstones. *ACS Omega* **5**(46), 29816–29829 (2020).
2. Nasralla, R. A., Alotaibi, M. B. & Nasr-El-Din, H. A. *Efficiency of oil recovery by low salinity water flooding in sandstone reservoirs*. In *SPE Western North American Region Meeting*. (OnePetro, 2011).
3. Fogang, L. T. et al. Oil/water interfacial tension in the presence of novel polyoxyethylene cationic Gemini surfactants: Impact of spacer length, unsaturation, and aromaticity. *Energy Fuels* **34**(5), 5545–5552 (2020).
4. Bangtang, Y. I. N. et al. Deformation and migration characteristics of bubbles moving in gas-liquid countercurrent flow in annulus. *Pet. Explor. Dev.* **52**(2), 471–484 (2025).
5. Cao, D. et al. Correction of linear fracture density and error analysis using underground borehole data. *J. Struct. Geol.* **184**, 105152 (2024).
6. Yin, B. et al. An experimental and numerical study of gas-liquid two-phase flow moving upward vertically in larger annulus. *Eng. Appl. Comput. Fluid Mech.* **19**(1), 2476605 (2025).
7. Kim, S., Kim, T.-W. & Jo, S. Artificial intelligence in geoenergy: Bridging petroleum engineering and future-oriented applications. *J. Petrol. Explor. Prod. Technol.* **15**(2), 35 (2025).
8. Alakbari, F. S. et al. Prediction of Poisson's ratio for a petroleum engineering application: Machine learning methods. *PLoS ONE* **20**(2), e0317754 (2025).
9. Honarvar, B. et al. Smart water effects on a crude oil-brine-carbonate rock (CBR) system: Further suggestions on mechanisms and conditions. *J. Mol. Liq.* **299**, 112173 (2020).
10. Gomez, S., Mansi, M. & Fahes, M. Quantifying the non-monotonic effect of salinity on water-in-oil emulsions towards a better understanding of low-salinity-water/oil/rock interactions. In *Abu Dhabi International Petroleum Exhibition & Conference* D031S088R002 (2018).
11. Nasr-El-Din, H. A. et al. Field treatment to stimulate an oil well in an offshore sandstone reservoir using a novel, low-corrosive, environmentally friendly fluid. *J. Can. Pet. Technol.* **54**(05), 289–297 (2015).
12. Sualihu, M. A. et al. Financial planning and forecasting in the oil and gas industry. In *The Economics of the Oil and Gas Industry* 180–199 (Routledge, 2023).
13. Zhang, J. et al. Integrating petrophysical, hydrofracture, and historical production data with self-attention-based deep learning for shale oil production prediction. *SPE J.* **29**(12), 6583–6604 (2024).
14. Hasankhani, G. M. et al. Experimental investigation of asphaltene-augmented gel polymer performance for water shut-off and enhancing oil recovery in fractured oil reservoirs. *J. Mol. Liq.* **275**, 654–666 (2019).
15. Abbasi, P., Aghdam, S. K. Y. & Madani, M. Modeling subcritical multi-phase flow through surface chokes with new production parameters. *Flow Meas. Instrum.* **89**, 102293 (2023).
16. Khezerlooe-ye Aghdam, S. et al. Mechanistic assessment of Seidlitzia Rosmarinus-derived surfactant for restraining shale hydration: A comprehensive experimental investigation. *Chem. Eng. Res. Des.* **147**, 570–578 (2019).
17. Alkouh, A. et al. Explicit data-based model for predicting oil-based mud viscosity at downhole conditions. *ACS Omega* **9**(6), 6684–6695 (2024).
18. Chen, S.-S. & Chen, H.-C. Oil prices and real exchange rates. *Energy Econ.* **29**(3), 390–404 (2007).
19. El-Sebakhy, E. A. Forecasting PVT properties of crude oil systems based on support vector machines modeling scheme. *J. Petrol. Sci. Eng.* **64**(1), 25–34 (2009).
20. Agwu, O. E. et al. Carbon capture using ionic liquids: An explicit data driven model for carbon (IV) oxide solubility estimation. *J. Clean. Prod.* **472**, 143508 (2024).
21. Agwu, O. E., et al. Applications of artificial intelligence algorithms in artificial lift systems: A critical review. In *Flow Measurement and Instrumentation* 102613 (2024).
22. Espinoza, R. Digital oil field powered with new empirical equations for oil rate prediction. In *SPE Middle East Intelligent Oil and Gas Conference and Exhibition* (2015).
23. Kargarpour, M. A. Oil and gas well rate estimation by choke formula: Semi-analytical approach. *J. Petrol. Explor. Prod. Technol.* **9**(3), 2375–2386 (2019).
24. Farag, W. A. Virtual multiphase flow meter for high gas/oil ratios and water-cut reservoirs via ensemble machine learning. *Exp. Comput. Multiphase Flow* **8**, 1–16 (2025).
25. Souza, B. G. Jr., da Fontoura, S. A. B. & Inoue, N. Adaptive criterion for iterative hydromechanical coupling in black-oil reservoir using pseudocompressibility. *Int. J. Geomech.* **25**(5), 04025066 (2025).
26. Abugoffa, R. H., Almabruk, A. A. & Abozaid, H. H. *Troubleshooting Techniques for Electric Submersible Pumps (ESPs).*
27. Agwu, O. E. et al. Utilization of machine learning for the estimation of production rates in wells operated by electrical submersible pumps. *J. Petrol. Explor. Prod. Technol.* **14**(5), 1205–1233 (2024).
28. Jiang, Y. et al. Predicting gas flow rates of wellhead chokes based on a cascade forwards neural network with a historically limited penetrable visibility graph. *Appl. Intell.* **55**(6), 1–17 (2025).
29. Kurtz, P. W. et al. Low-energy electron beam modification of metallic biomaterial surfaces: Oxygen and silicon-rich amorphous carbon as a wear-resistant coating. *J. Biomed. Mater. Res. Part A* **113**(2), e37849 (2025).
30. Sun, H. et al. Theoretical and numerical methods for predicting the structural stiffness of unbonded flexible riser for deep-sea mining under axial tension and internal pressure. *Ocean Eng.* **310**, 118672 (2024).

31. Yanchun, L. I. et al. Surrogate model for reservoir performance prediction with time-varying well control based on depth generative network. *Pet. Explor. Dev.* **51**(5), 1287–1300 (2024).
32. Yu, H. et al. Modeling thermal-induced wellhead growth through the lifecycle of a well. *Geoenergy Sci. Eng.* **241**, 213098 (2024).
33. Yang, M. et al. Probing structural modification of milk proteins in the presence of pepsin and/or acid using small-and ultra-small-angle neutron scattering. *Food Hydrocolloids* **159**, 110681 (2025).
34. Agwu, O. E. et al. Modelling the flowing bottom hole pressure of oil and gas wells using multivariate adaptive regression splines. *J. Petrol. Explor. Prod. Technol.* **15**(2), 22 (2025).
35. Schlussel, E. J. et al. Flow characteristics in an optically accessible solid fuel scramjet. *J. Propul. Power* **8**, 1–10 (2025).
36. Paxton, B. T., Sykes, J. & Rankin, B. A. *Pattern Factor and Combustion Efficiency Measurements in a Full-Annular Partially Premixed Pre-Vaporized Small-Scale Combustor*.
37. Hastie, T., et al., Ensemble learning. In *The Elements of Statistical Learning: Data Mining, Inference, and Prediction* 605–624 (2009).
38. Luna, J. M. et al. Building more accurate decision trees with the additive tree. *Proc. Natl. Acad. Sci.* **116**(40), 19887–19893 (2019).
39. Zulfiqar, H. et al. Identification of cyclin protein using gradient boost decision tree algorithm. *Comput. Struct. Biotechnol. J.* **19**, 4123–4131 (2021).
40. Ayyadevara, V. K. Gradient boosting machine. In *Pro Machine Learning Algorithms: A Hands-On Approach to Implementing Algorithms in Python and R* 117–134 (Apress, 2018).
41. Fan, J. et al. Light gradient boosting machine: An efficient soft computing model for estimating daily reference evapotranspiration with local and external meteorological data. *Agric. Water Manag.* **225**, 105758 (2019).
42. Taha, A. A. & Malebary, S. J. An intelligent approach to credit card fraud detection using an optimized light gradient boosting machine. *IEEE Access* **8**, 25579–25587 (2020).
43. Cha, G.-W., Moon, H.-J. & Kim, Y.-C. Comparison of random forest and gradient boosting machine models for predicting demolition waste based on small datasets and categorical variables. *Int. J. Environ. Res. Public Health.* https://doi.org/10.3390/ijerph18168530 (2021).
44. AlKhulaifi, D. et al. An overview of self-adaptive differential evolution algorithms with mutation strategy. *Math. Modell. Eng. Problems* **9**(4), 84 (2022).
45. Srivastava, G. & Pradhan, N. Handling imbalanced class in melanoma: Kemeny-Young rule based optimal rank aggregation and self-adaptive differential evolution optimization. *Eng. Appl. Artif. Intell.* **125**, 106738 (2023).
46. Brest, J., Maučec, M. S. & Bošković. B. *Self-Adaptive Differential Evolution Algorithm with Population Size Reduction for Single Objective Bound-Constrained Optimization: Algorithm j21*. IEEE.
47. Fister, I. et al. Design and implementation of parallel self-adaptive differential evolution for global optimization. *Logic J. IGPL* **31**(4), 701–721 (2023).
48. Gouda, S. K. & Mehta, A. K. Software cost estimation model based on fuzzy C-means and improved self adaptive differential evolution algorithm. *Int. J. Inf. Technol.* **14**(4), 2171–2182 (2022).
49. Mohaideen Abdul Kadhar, K. et al. Parameter evaluation of a nonlinear Muskingum model using a constrained self-adaptive differential evolution algorithm. *Water Pract. Technol.* **17**(11), 2396–2407 (2022).
50. Yang, Z., Tang, K. & Yao, X. *Self-Adaptive Differential Evolution with Neighborhood Search*. IEEE.
51. Deng, W. et al. An improved self-adaptive differential evolution algorithm and its application. *Chemom. Intell. Lab. Syst.* **128**, 66–76 (2013).
52. Fan, Q. & Yan, X. Self-adaptive differential evolution algorithm with zoning evolution of control parameters and adaptive mutation strategies. *IEEE Trans. Cybern.* **46**(1), 219–232 (2015).
53. Hansen, N., Arnold, D. V. & Auger, A. *Evolution Strategies.* Springer Handbook of Computational Intelligence 871–898 (2015).
54. Beyer, H.-G. & Schwefel, H.-P. Evolution strategies–a comprehensive introduction. *Nat. Comput.* **1**, 3–52 (2002).
55. Sui, X., Chen, Q. & Gu, G. Adaptive bias voltage driving technique of uncooled infrared focal plane array. *Optik* **124**(20), 4274–4277 (2013).
56. Zhao, L.-C. et al. Fast and sensitive LC-DAD-ESI/MS method for analysis of Saikosaponins c, a, and d from the roots of *Bupleurum falcatum* (Sandaochaihu). *Molecules* **16**(2), 1533–1543 (2011).
57. Zhu, B., et al. *KNN-Based Single Crystal High Frequency Transducer for Intravascular Photoacoustic Imaging*. IEEE.
58. Fang, T. et al. Multi-scale mechanics of submerged particle impact drilling. *Int. J. Mech. Sci.* **285**, 109838 (2025).
59. Zhang, L. et al. Seepage characteristics of broken carbonaceous shale under cyclic loading and unloading conditions. *Energy Fuels* **38**(2), 1192–1203 (2023).
60. Mezura-Montes, E. & Coello, C. A. C. An empirical study about the usefulness of evolution strategies to solve constrained optimization problems. *Int. J. Gen. Syst.* **37**(4), 443–473 (2008).
61. Jiang, L. et al. Improving tree augmented Naive Bayes for class probability estimation. *Knowl. Based Syst.* **26**, 239–245 (2012).
62. Ament, S. et al. Unexpected improvements to expected improvement for bayesian optimization. *Adv. Neural. Inf. Process. Syst.* **36**, 20577–20612 (2023).
63. Laitila, P. & Virtanen, K. Improving construction of conditional probability tables for ranked nodes in Bayesian networks. *IEEE Trans. Knowl. Data Eng.* **28**(7), 1691–1705 (2016).
64. Alatefi, S., Agwu, O. E. & Alkouh, A. Explicit and explainable artificial intelligent model for prediction of CO_2 molecular diffusion coefficient in heavy crude oils and bitumen. *Results Eng.* **24**, 103328 (2024).
65. Liu, Y. et al. Improved naive Bayesian probability classifier in predictions of nuclear mass. *Phys. Rev. C* **104**(1), 014315 (2021).
66. Dai, T. et al. Waste glass powder as a high temperature stabilizer in blended oil well cement pastes: Hydration, microstructure and mechanical properties. *Constr. Build. Mater.* **439**, 137359 (2024).
67. Zhang, L. et al. Seepage characteristics of coal under complex mining stress environment conditions. *Energy Fuels* **38**(17), 16371–16384 (2024).
68. Farid, D. M. & Rahman, M. Z. Anomaly network intrusion detection based on improved self adaptive bayesian algorithm. *J. Comput.* **5**(1), 23–31 (2010).
69. González, J., et al. *Batch Bayesian Optimization via Local Penalization*. PMLR.
70. Azimi, J., Jalali, A. & Fern, X. *Hybrid Batch Bayesian Optimization.* arXiv preprint arXiv:1202.5597 (2012).
71. Oh, C. et al. Batch Bayesian optimization on permutations using the acquisition weighted kernel. *Adv. Neural. Inf. Process. Syst.* **35**, 6843–6858 (2022).
72. Liu, J., Jiang, C. & Zheng, J. Batch bayesian optimization via adaptive local search. *Appl. Intell.* **51**(3), 1280–1295 (2021).
73. Tamura, C., et al., *Autonomous Organic Synthesis for Redox Flow Batteries via Flexible Batch Bayesian Optimization* (2025).
74. Vujović, Ž. Classification model evaluation metrics. *Int. J. Adv. Comput. Sci. Appl.* **12**(6), 599–606 (2021).
75. Buran, B. & Erçek, M. Public transportation business model evaluation with spherical and intuitionistic fuzzy AHP and sensitivity analysis. *Expert Syst. Appl.* **204**, 117519 (2022).
76. Madani, M., Moraveji, M. K. & Sharifi, M. Modeling apparent viscosity of waxy crude oils doped with polymeric wax inhibitors. *J. Petrol. Sci. Eng.* **196**, 108076 (2021).
77. Hasanzadeh, M. & Madani, M. Deterministic tools to predict gas assisted gravity drainage recovery factor. *Energy Geosci.* **5**(3), 100267 (2024).
78. Madani, M. & Alipour, M. Gas-oil gravity drainage mechanism in fractured oil reservoirs: Surrogate model development and sensitivity analysis. *Comput. Geosci.* **26**(5), 1323–1343 (2022).

79. Khan, J. A. & Chen, Y. *Mechanism and Oil-Water Pressure Drop of Unique Autonomous Inflow Control Device Under Different Water Cut: Water Control Performance of AICD in Large Bottom Water Reservoir in South Sudan*. IPTC.
80. Zhang, Y., et al. *Well Production Prediction Method Based on Multi-Factor Fusion Time Series Model*. IPTC.
81. Dasuki, N. A., et al. *Extending the Lifespan of Marginal Field Through in-Situ Gas Lift in Sarawak Offshore*. IPTC.
82. Segaran, T. C., et al. *An Innovative Breakthrough in Gas Lift Optimization Analysis That Improves Upon the Current Best Practices Established in The Industry–An Effort to Know Your Well Better from The Surface in One Glance*. IPTC.
83. Franco, C. A. et al. Enhancing heavy crude oil mobility at reservoir conditions by nanofluid injection in wells with previous steam stimulation cycles: Experimental evaluation and field trial implementation. *J. Mol. Liq.* **6**, 127024 (2025).
84. Gallego, J. F. et al. Demulsification of water-in-oil emulsion with carbon quantum dot (CQD)-enhanced demulsifier. *Processes* **13**(2), 575 (2025).
85. Qiao, M., Zhang, F. & Li, W. Rheological properties of crude oil and produced emulsion from CO2 flooding. *Energies* **18**(3), 739 (2025).

Acknowledgements

The authors extend their appreciation to the Deanship of Research and Graduate Studies at King Khalid University for funding this work through Large Research Project under grant number RGP2/215/46.

Author contributions

All authors contributed equally to this research paper.

Declarations

Competing interests

The authors declare no competing interests.

Additional information

Correspondence and requests for materials should be addressed to Z.X. or H.A.

Reprints and permissions information is available at www.nature.com/reprints.

Publisher's note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.