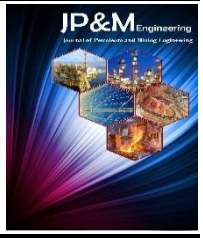
JP&M Engineering
Journal of Petroleum and Mining Engineering



Enhancing Lithology Classification From Well Logs Using Weighted Class Training and Bayesian Optimization in Voting Classifiers

Obika, F. Chidera[*a] , Unegbu Chukwudi[b] and Okereke Ndubuisi [a]
[a]Department of Petroleum Engineering, Federal University of Technology, Owerri P.M.B. 1526, Imo State, Nigeria.
[b]Shell Nigeria Exploration and Production Company, 21/22 Marina P.M.B. 2418, Lagos State, Nigeria.
*Corresponding author e-mail: obikafranklin.20191174703@futo.edu.ng

Article Info

Received 23 Dec. 2024
Revised 4 Jun. 2025
Accepted 10 Jun. 2025

Keywords

Voting Classifier; Imbalanced Data; Ensemble methods; Lithology classification; Bayesian Optimization

Abstract

Lithology classification is crucial for understanding subsurface geology and enhancing petroleum resource exploration. This study proposes a voting classifier that combines two base models, namely Weighted Class Random Forest (WCRF) and Bayesian Optimized Extreme Gradient Boosting (XGBoost-BO), to improve lithology classification from well log data. The dataset comprised imbalanced well log data from 10 Norwegian wells in the Utsira formation, totaling 4,327 samples with 7 well logs, and 7 lithology classes. WCRF handled class imbalance by assigning weights to each class based on the reciprocal of class frequencies in the training data, while XGBoost-BO used a balanced training dataset created with the Synthetic Minority Oversampling Technique (SMOTE). The models' performance was assessed using metrics like the F1-Score, the area under the receiver operating characteristic curve (AUC), and confusion matrix. The average AUC values for WCRF and XGBoost-BO were 0.982 and 0.985, respectively, showing high generalization performance. The voting classifier achieved the highest performance, with an average F1-Score of 0.874, surpassing WCRF and XGBoost-BO with F1 scores of 0.866 and 0.861, respectively. This voting classifier enhances the accuracy and efficiency of identifying subsurface rock types, ultimately reducing costs and risks associated with drilling by leveraging data-driven insights.

Introduction

Lithology refers to the physical and mineralogical characteristics of rock layers, such as composition, grain size, texture, and color, which are used to identify and classify different rock types within the subsurface. Lithology is relevant for understanding reservoir quality, predicting fluid flow, and making informed drilling decisions. Traditional lithology classification methods rely heavily on manual interpretation of well logs, core samples, and seismic data by geologists and petrophysicists [1-3]. This process involves analysing physical measurements, such as gamma ray, resistivity, density, and sonic logs, to identify different rock types and their properties. Visual examination and interpretation are essential in this approach, often requiring extensive experience and geological knowledge to accurately infer lithological characteristics. While effective, traditional methods are time-consuming, subjective, and can lead to inconsistencies due to human bias or variability in geological formations [4-6]. Conversely, machine learning provides an automated, data-driven method capable of processing large amounts of well log and seismic data quickly and consistently [7,8].

Machine learning (ML)-based lithology classification using well log data has become a transformative approach in the oil and gas industry, allowing for faster and more accurate subsurface characterization [9,10]. Well logs, offering measurements like gamma ray, resistivity, density, and neutron porosity, deliver detailed subsurface data that can be used to deduce rock types and properties [11]. By training ML models on labelled well log data, algorithms can learn to recognize patterns that indicate different lithologies, automating a process that traditionally required expert interpretation. Supervised learning algorithms such as decision trees, ensemble methods, support vector machines, or neural networks, are commonly applied to classify lithology by correlating specific well log readings with known rock types. These algorithms have demonstrated superior performance compared to unsupervised learning methods like K-Means, Hierarchical Clustering, Principal Component Analysis (PCA), and t-SNE (t-Distributed Stochastic Neighbor Embedding) [12}.

Wang et al. [13] employed artificial neural networks (ANN) and support vector machine (SVM) in classifying shale lithofacies from well conventional

logs. They identified SVM as the preferred model. Xie et al. [14] evaluated the performance of five ML algorithms, namely the Naïve Bayes, Support Vector Machine (SVM), Artificial Neural Network (ANN), Random Forest (RF) and Gradient Tree Boosting (GTB), for formation lithology classification using well log data from the Daniudui gas field and the Hangjinqi gas field. They inferred that the ensemble methods (RF and GTB) had better performance than the other ML methods. Sun et al. [15] investigated the performance of three ML methods in classifying lithology using well log data from Yan'an Gas Field. The results showed that RF performed best. Saporetti et al. [16] showed that Gradient Tree Boosting with a Differential Evolution (GTB-DE) performed better in classifying lithology than standard GTB, using well log data from the Daniudui gas field and the Hangjinqi gas field. Sun et al. [12] applied XGBoost with Bayesian Optimization (XGBoost-BO) in lithology classification using well log data from the Daniudui gas field and the Hangjinqi gas field. They compared the performance of the proposed model with that of GTB and Gradient Tree Boosting with a Differential Evolution (GTB-DE). The results revealed that XGBoost-BO performed better than GTB and GTB_DE. Zhang et al. [17] examined the performance of linear and non-linear ML models and concluded that the non-linear models, especially RF and XGBoost, had the best performance.

In this study, a voting classifier, incorporating Weighted Class Random Forest (WCRF) and Bayesian Optimized Extreme Gradient Boosting (XGBoost-BO) as base models, was used to improve lithology classification using well log data. The performance of the proposed classifier was evaluated based on classification metrics such as F1-Score, area under the receiver operating characteristic curve (AUC), and Confusion Matrix. The performance of the proposed voting classifier was compared with that of the base models.

Materials and Methods

Study Area and Dataset

The study made use of the train data from the FORCE Machine Learning competition with well logs and seismic 2020 [18]. This train data contained 18 well logs from 98 wells, 70 formations and 15 groups in the Norwegian Sea. The Norwegian Sea is a significant area for oil and gas exploration, located between the North Sea and Barents Sea along Norway's continental shelf as shown in Figure 1. This region is characterized by deep waters, often ranging from 200 to over 3,000 meters, and hosts several prolific hydrocarbon basins, such as the Haltenbanken, More, and Voring basins. Reservoirs in the Norwegian Sea are typically composed of Jurassic sandstones, with additional potential in Cretaceous and Triassic formations. One of the key fields, the Ormen Lange gas field, is among Europe's largest, providing natural gas primarily to the UK. How-ever, the Norwegian Sea poses considerable challenges due to its complex subsurface geology, including thick shale sequences and faulted structures, which necessitate advanced drilling technologies and seismic imaging. Additionally, the cold climate, deepwater conditions, and strict environmental regulations of Norway require opera-tors to employ sustainable practices, including minimizing flaring, managing emissions, and protecting the marine ecosystem. The Norwegian Sea thus represents a blend of vast resource potential and high operational standards, reflective of Norway's strategic approach to offshore energy development.

Figure 1 Location of the Norwegian Sea in the North Atlantic Ocean [19].

This study made use of ten wells from the Utsira Formation namely, 16/10-3, 16/8-1, 25/5-1, 30/3-3, 30/6-5, 31/4-5, 34/10-19, 34/10-21, 34/2-4, and 34/8-3. Data pre-processing techniques were employed to handle missing values, redundant features, and outliers. Spectral Gamma Ray log (SGR), Neutron-Porosity log (NPHI), Shear wave sonic log (DTS, us/ft), Photo Electric Factor log (PEF) were dropped due to a large percentage of missing values (above 50%) in the features as shown in Figure 2. The missing values in other features were filled with the median of each feature. This proved more effective than filling with the mean since outliers sway the mean. In addition, the Medium Resistivity Log (RMED) was dropped due to its high Pearson's correlation coefficient of 0.72 with the Deep Resistivity Log (RDEP) as shown in Figure 3. This mitigated the effect of multi-collinearity on the models' performance [20]. Moreover, dropping RMED instead of RDEP showed better model performance.

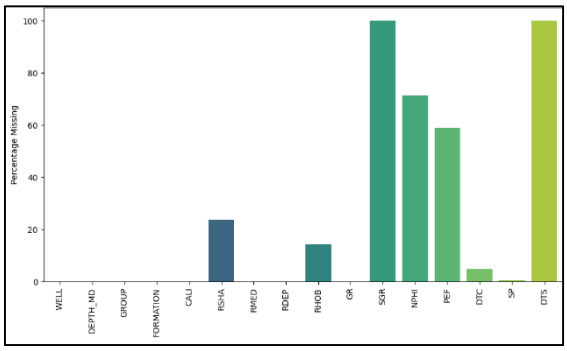


Figure 2 A bar plot showing the percentage of missing values in the dataset.

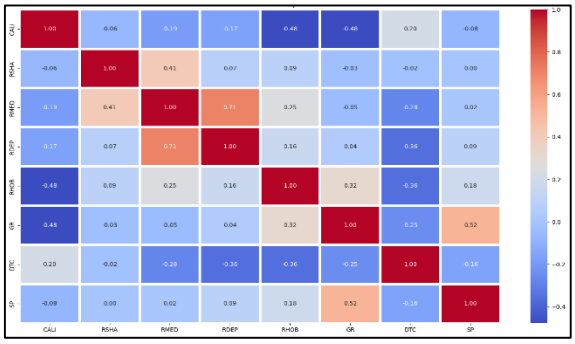
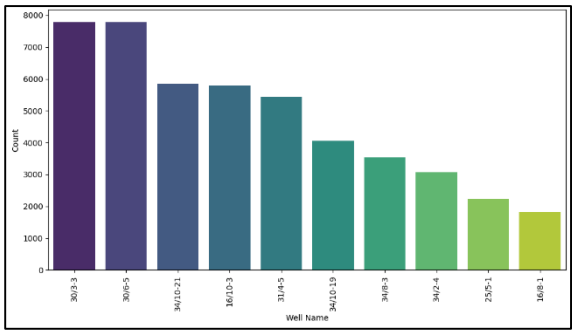
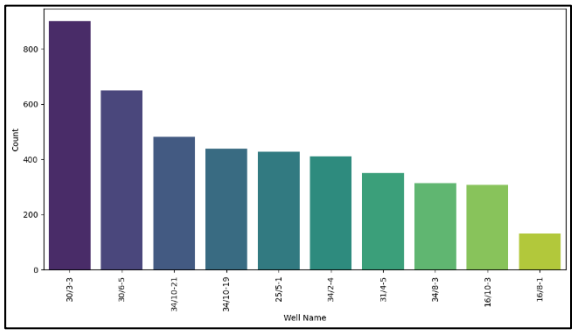


Figure 3 Heat map of the correlation matrix on the training dataset (before dropping RMED).

The data from the ten wells contained 47,378 samples, which would require large computational resources for model training. Hence, the samples of the majority classes (shale, sandstone, and sandstone/Shale) were under-sampled to 1000 samples each to minimize computation. The total number of samples in the data became 4,412 samples after under-sampling the majority classes as shown in Figure 4.



(a)



(b)

Figure 4 Distribution of data points across the selected ten wells in the Utsira Formation. (a) before under-sampling the majority classes to 1000 data points; (b) after under-sampling the majority classes to 1000 data points.

There were only 7 samples of the marl class in the data, so it was dropped as the number of samples was very small for model training. Hence, the data was left with seven lithology classes with the minority classes being tuff, dolomite, and coal. Figure 5 shows that coal was situated within a depth range of about 700m to 850m, and tuff could be found at greater depths of about 1700m to 1850m. Figure 6 shows the distribution of lithology classes across depth in well 16/10-3, well 34/10-19, and well 30/3-3, while Figure 7 shows the count of lithology classes across the ten wells.

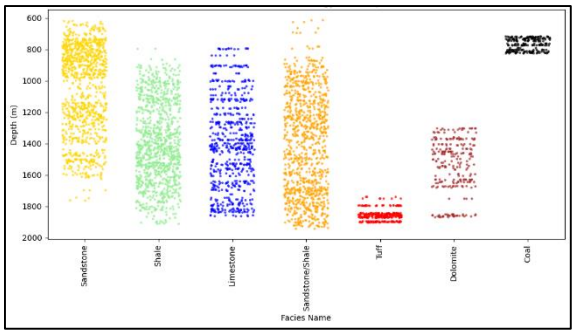
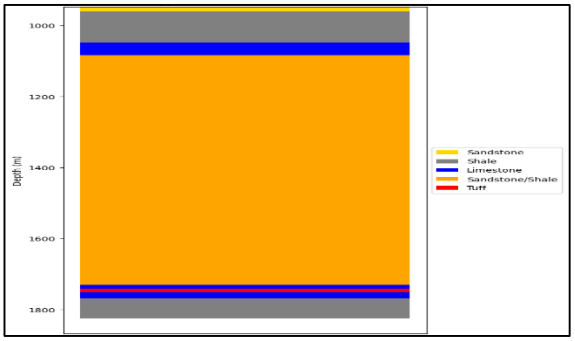
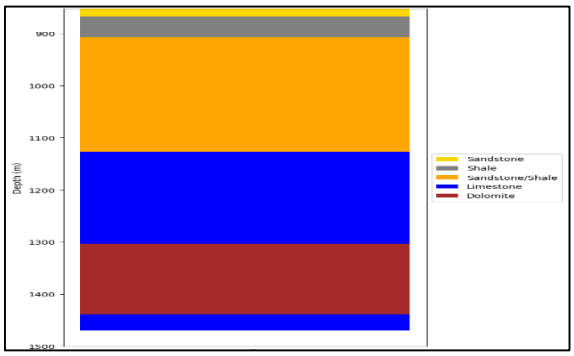


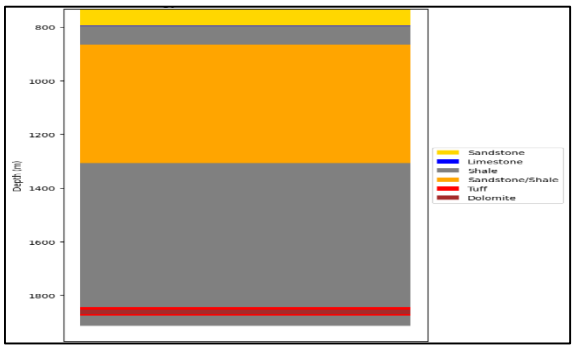
Figure 5 The distribution of lithology classes across depth.



(a)

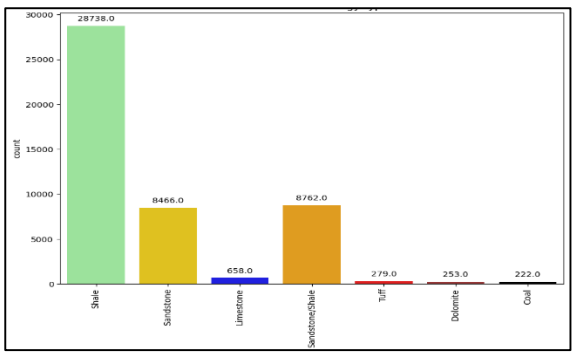


(b)



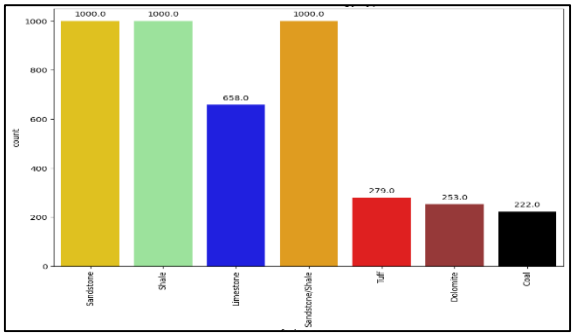
(c)

Figure 6 The distribution of lithology classes across depth in three wells. (a) well 16/10-3; (b) well 34/10-19; (c) well 30/3-3.



(a)

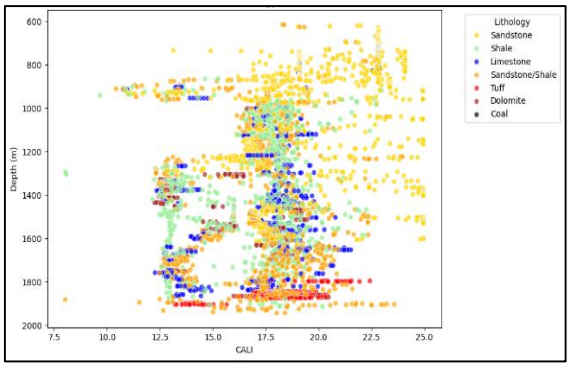
 DOI: 10.21608/jpme.2025.344501.1221



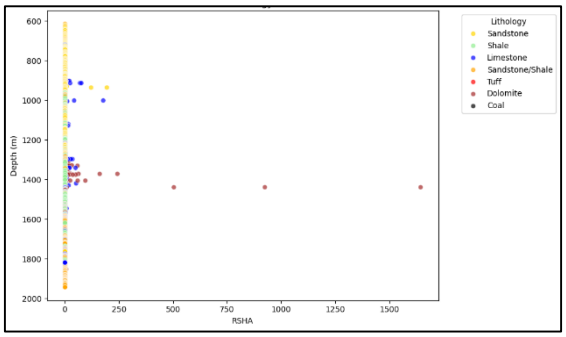
(b)

Figure 7 The count of lithology classes across the ten wells: (a) before under-sampling the majority classes to 1000 data points; (b) after under-sampling the majority classes to 1000 data points.

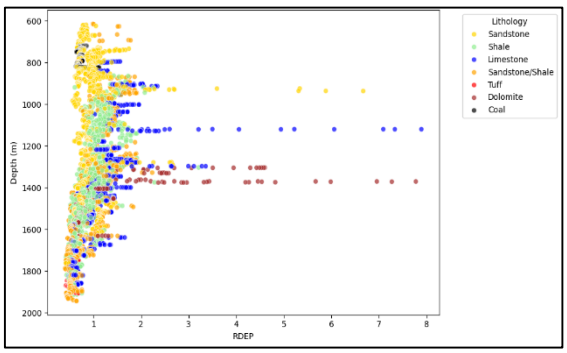
We used Tukey's Fences for outliers' detection and removal based on the interquartile range (IQR) [21]. Still, it did not improve the performance of the models as relevant data points were regarded as outliers and removed. Hence, the detection and removal of outliers were based on the distribution of the lithology classes over depth for each well log as shown in Figure 8. For instance, in the SP and RSHA logs, values beyond 150 and 250 were considered outliers, respectively.



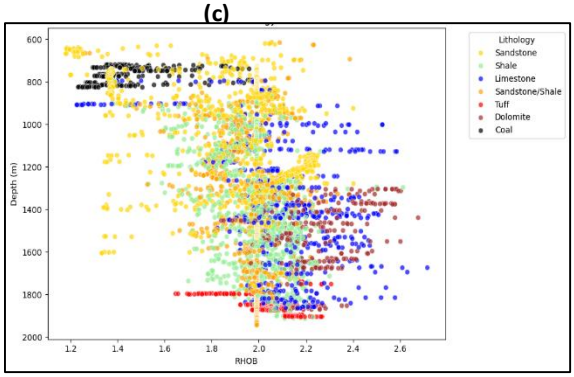
(a)



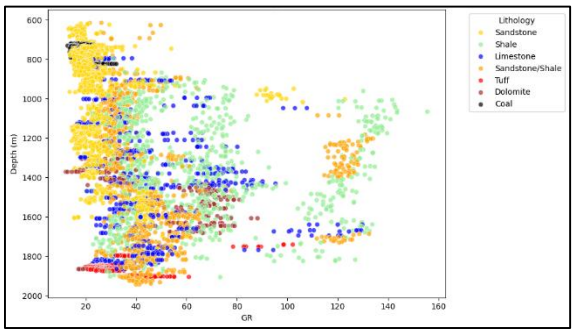
(b)



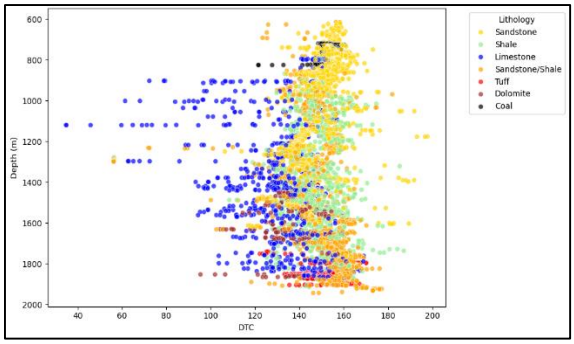
(c)



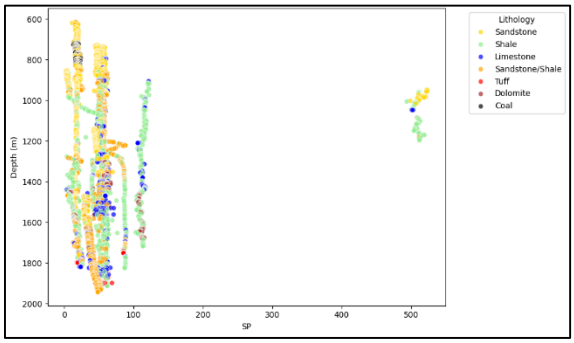
(d)



(e)



(f)



(g)

Figure 8 Distribution of the lithology classes over depth for each well log. (a) CALI; (b) RSHA; (c) RDEP; (d) RHOB; (e) GR; (f) DTC; (g) SP

After carrying out exploratory data analysis (EDA), features selection, and outliers' removal, the final data used for the study contained 4,329 samples, 7 well logs, and 7 lithology classes. Table 1 shows the summary statistics of the cleaned data used in the study for both model training and evaluation.

Table 1 Summary statistics of well logs in the cleaned data

Well Logs	Mean	Min	Q1	Median	Q3	Max
CALI	17.835340	10.424725	16.316009	17.908951	19.214706	24.967262
RSHA	1.563277	0.135231	0.787387	0.910921	1.139654	243.076035
RDEP	0.967922	0.401266	0.661833	0.877946	1.134758	7.773187
RHOB	1.962430	1.179810	1.893999	1.991170	2.087290	2.716326
GR	42.057914	12.119803	24.796065	35.961948	49.410852	138.245102
DTC	145.624878	45.839245	139.869774	148.311600	154.382927	197.798416
SP	47.953064	2.490359	20.529184	49.375645	60.270714	122.954636

Methods

This work was carried out using Google Colaboratory cloud resources with a CPU, 12 Gigabytes of RAM, and 2 CPU cores for training, and evaluation. The models in this work were built using Python programming language and pre-built libraries such as Numpy, Pandas, Matplotlib, Seaborn, and Scikit-learn.

Random Forest (RF)

The Random Forest algorithm is an ensemble learning technique that enhances accuracy and minimizes overfitting by aggregating predictions from multiple decision trees. As shown in Figure 9, Random Forest operates by building a collection of decision trees, each trained on a randomly selected subset of the training data and features. This randomness ensures that each tree captures unique patterns in the data. During the training of each decision tree, only a subset of features is used to determine splits at each node, reducing correlations between trees and fostering diversity in predictions. This approach, called bagging (bootstrap aggregating), leverages the

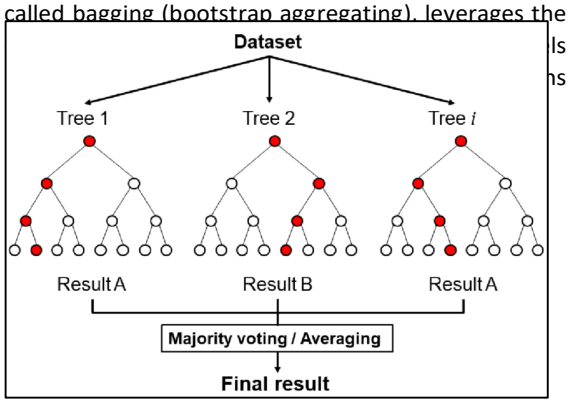


Figure 9 Schematic Diagram of Random Forest Classifier [23]

Once all the trees are trained, the Random Forest predicts by combining the out-puts of the individual trees. For classification tasks, it employs majority voting, where each tree votes for a class label, and the class with the most votes becomes the final prediction. For regression tasks, it calculates the average of the predictions from all the trees. This ensemble method lowers the risk of overfitting, as the final decision is based on multiple diverse models rather than a single complex tree. Random Forests are versatile and perform well across various tasks, demonstrating resilience to outliers and noisy data [24], making them a popular choice in machine learning for both classification and regression problems [25]. In this study, RF was designed to handle the class imbalance by assigning weights to each class based on the reciprocal of class frequencies in the training data, leading to the formation of a weighted class RF (WCRF) [26]. In another approach, RF was trained using a balanced data created with SMOTE leading to the formation of RF-SMOTE.

Extreme Gradient Boosting (XGBoost)

The XGBoost (Extreme Gradient Boosting) algorithm is an advanced implementation of gradient boosting that is highly efficient and optimized for speed and performance. As shown in Figure 10, It works by creating an ensemble of decision trees sequentially, where each new tree aims to correct errors made by the previous trees. It uses a regularized objective function that balances model complexity and accuracy, helping to avoid overfitting, and supports both L1 and L2 regularization [27,28]. A unique feature of XGBoost is its ability to perform gradient boosting with an optimized tree-growing technique called exact greedy algorithm, which makes the splitting of data points more precise and efficient [29]. XGBoost also includes features such as shrinkage (a learning rate that moderates the addition of new trees), column subsampling (selecting a subset of features to construct each tree, similar to Random Forests), and parallel processing, which enhances both the speed and accuracy of training [27]. These techniques allow XGBoost to efficiently process large datasets with exceptional performance, making it a widely favored algorithm in machine learning for both classification and regression tasks [30].

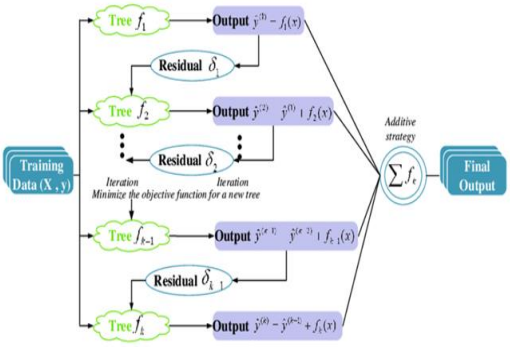


Figure 10 XGBoost Architecture [31]

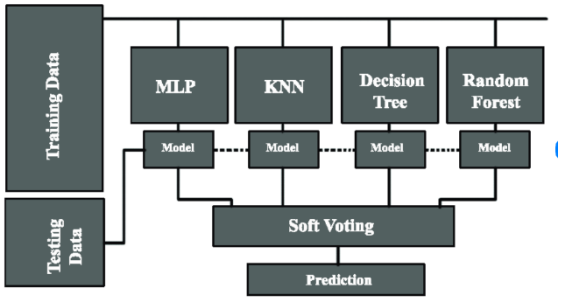


Figure 11 An example of a voting classifier architecture [36].

XGBoost and Random Forest are both ensemble learning methods, but they differ in how they combine trees to generate predictions. XGBoost employs boosting, where trees are built sequentially, with each tree correcting the errors of the previous one. This approach focuses on challenging examples, making XGBoost highly accurate but susceptible to overfitting without proper tuning. In contrast, Random Forest uses bagging (bootstrap aggregating), where multiple trees are built independently and in parallel on random subsets of data and features. It then averages the predictions from all trees, reducing variance and improving robustness. XGBoost also integrates regularization and optimization techniques to enhance accuracy and speed, while Random Forest relies more on randomness to ensure diversity among trees. These distinctions make XGBoost better suited for tasks that demand high precision, while Random Forest is generally simpler and less prone to overfitting.

Voting Classifier

A voting classifier is an ensemble learning method that combines the predictions from several individual models to enhance classification performance [32] as shown in Figure 11. The core concept is that each model generates its own prediction for a given input, and the voting classifier combines these predictions to make a final decision. There are two primary types of voting: majority voting (hard voting) and weighted voting (soft voting). In majority voting, each model votes for a class, and the class with the most votes becomes the final prediction. In weighted voting, the classifier considers the probability estimates from each model, assigning weights based on model confidence or performance, and then selects the class with the highest average probability. This method harnesses the strengths of each model, enabling the ensemble to produce more accurate and robust predictions than any individual model [33-35].

Synthetic Minority Oversampling Technique (SMOTE)

SMOTE employs a more advanced technique for oversampling by generating entirely new synthetic data points for the minority class. It identifies the k-nearest neighbors (similar data points) for each sample in the minority class, then randomly selects one of these neighbors and creates a new data point along the line connecting them in feature space [37]. The new point retains characteristics of the original minority class sample but with slight differences, which helps introduce more variation and prevent overfitting. This process contributes to a more balanced dataset by augmenting the minority class with synthetic samples that are similar to existing ones, yet include some variation. This can enhance the performance of classification algorithms by offering a more balanced training dataset [38-40].

Bayesian Optimization

Bayesian optimization is an effective method for optimizing objective functions that are costly to evaluate. It probabilistically models the objective function, often using a Gaussian process (GP) as a surrogate model to approximate the actual function [41,42]. Based on prior evaluations of the function, the Gaussian process creates a probability distribution over potential functions that fit the observed data, enabling predictions for unobserved points. Bayesian optimization iteratively chooses the most promising points to evaluate next by balancing exploration (searching uncertain areas) and exploitation (focusing on areas with high estimated values) [43].

Bayesian optimization is especially beneficial when function evaluations are ex-pensive, such as in hyperparameter tuning for machine learning models. Unlike other search algorithms like grid search or random search, Bayesian optimization is more efficient because it learns from prior evaluations and minimizes the number of function evaluations needed [44]. It also outperforms gradient-based methods when the objective function is non-differentiable, discontinuous, or noisy. Moreover, by incorporating uncertainty into its surrogate model, Bayesian optimization effectively balances exploration and exploitation, making it particularly well-suited for

black-box optimization where the function's analytical form is unknown [45,46].

Data Normalization and Partitioning

The data contained features on vastly different scales, which can significantly impact machine learning models. Data Normalization was implemented to address this issue by transforming the data into a consistent range to ensure that all the features contributed equally to the model training [47]. This study employed the Min-Max Scaler technique, scaling all features between 0 and 1. This technique proved more effective than the Robust Scaler and Standard Scaler techniques. The data was then partitioned into two datasets for training and testing using 80% by 20% stratified split for an improved sampling of the general data [48,49]. The models were trained using the training data set and then evaluated using the testing dataset.

Performance Metrics

The performance of the models was evaluated using the F1 score, area under the receiver operating characteristic curve (AUC), and confusion matrix. The F1 score provides a balanced measure of both precision and recall, serving as the harmonic mean of these two metrics to offer a single value for overall model effectiveness. Precision assesses the accuracy of the model's positive predictions, calculated by dividing the number of correctly identified positive cases (True Positives) by the total number of predicted positive cases (True Positives + False Positives). Recall evaluates the model's ability to correctly identify actual positive cases, calculated as the ratio of correctly identified positives (True Positives) to the total number of actual positives (True Positives + False Negatives) [50,51]. The metrics are mathematically expressed as seen below;

$$\text{Precision} = TP / (TP + FP) \quad (1)$$

$$\text{Recall} = TP / (TP + FN) \quad (2)$$

$$\text{F1 Score} = 2[(\text{Precision} * \text{Recall}) / (\text{Precision} + \text{Recall})] \quad (3)$$

Where TP, FP, and FN are the True Positives, False Positives and False Negatives respectively.

Area under the receiver operating curve (AUC) is a metric used to evaluate the performance of a binary or multi-class classification model. It represents the area un-der the Receiver Operating Characteristic (ROC) curve, which plots the True Positive Rate (TPR) (recall) against the False Positive Rate (FPR) across various threshold set-tings for the classifier [52]. FPR measures the proportion of actual negatives that are incorrectly classified as positive by the model. The TPR and FPR can be calculated as;

$$TPR = \text{Recall} = TP / (TP + FN) \quad (4)$$

$$FPR = FP / (FP + TN) \quad (5)$$

Where TN is True Negative, the number of correctly predicted negative cases.

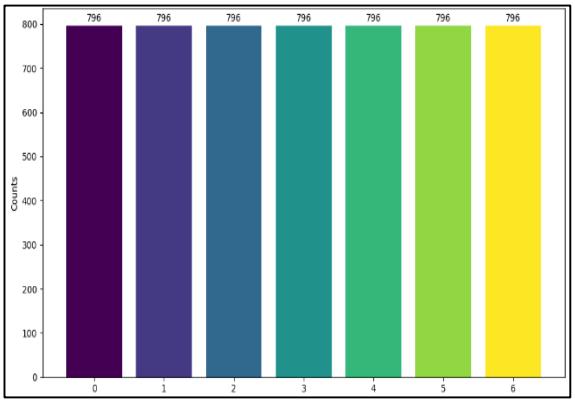
The AUC score ranges from 0 to 1. AUC = 1.0 indicates a perfect model that correctly classifies all positive and negative cases with no overlap. AUC = 0.5 represents a model that performs no better than random guessing, essentially following the diagonal line on the ROC plot. 0.5 < AUC < 1.0 reflects the model's ability to balance TPR and FPR effectively, with higher values showing stronger predictive accuracy [50,51]. This metric is particularly beneficial when the classes are imbalanced, as it evaluates the model's performance across all classification thresholds rather than a single accuracy score.

Results

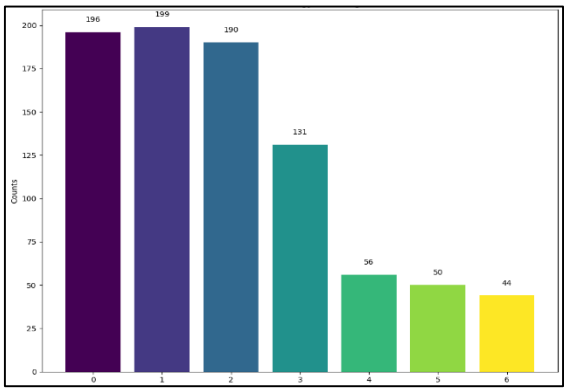
This section presents the results obtained from the machine learning algorithms dis-cussed in this study for lithology classification. First, the results of SMOTE are presented. Next, the outcomes of the XGBoost model tuning using Bayesian optimization are discussed. Finally, the performance of the machine learning models is evaluated using classification metrics, including F1 Score, area under the receiver operating characteristic curve (AUC), and the Confusion Matrix.

Handling Imbalanced Class

RF was designed to handle class imbalance by assigning weights to each class based on the reciprocal of class frequencies in the training data. In another approach, RF, alongside XGBoost-BO, was trained using a balanced data created using SMOTE. Only the training dataset (the testing dataset not included) was balanced using SMOTE. This allowed the model to learn from a balanced representation of classes while being evaluated on data that reflects the actual class distribution. Figure 12 shows the class distribution in the balanced training dataset and the imbalanced testing dataset.



(a)



(b)

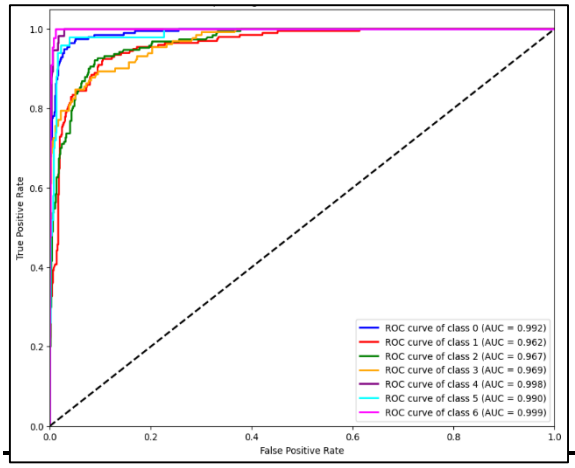
Figure 12 Lithology class distribution, where 0=Sandstone, 1=Sandstone/Shale, 2=Shale, 3=Limestone, 4=Tuff, 5=Dolomite, 6=Clay. (a) Balanced (SMOTE) training dataset; (b) imbalanced testing dataset.

Hyperparameter tuning

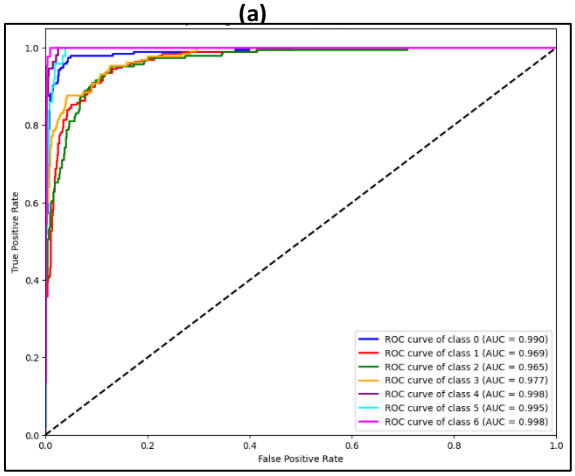
Table 2 presents the optimal hyperparameter settings tuned for the XGBoost model. The objective function was accuracy, and the number of iterations was set to 200. The number of boosted trees was randomly chosen in the interval [10, 1000]. The maximum depth of the tree was randomly chosen in the interval [1, 20]. The boosting learning rate was chosen from a uniform distribution ranging from 0.01 to 0.5. The subsample ratio was randomly chosen in the interval [0.1, 1]. The min child weight was randomly chosen in the interval [1, 5]. The gamma was randomly chosen in the interval [0.1, 0.5]. The colsample by tree was randomly chosen in the interval [0.5, 1]. The L2 regularization was randomly chosen in the interval [0.001, 0.01].

Model Evaluation

This study focused on the F1 Score for each lithology class instead of the precision and recall because F1 Score provides a balanced view of both Precision and Recall. Higher F1 Scores mean better classification accuracy, precision, and recall for that class. Table 3 shows the F1 scores and AUC values obtained from RF-SMOTE, WCRF, and XGBoost-BO. Figure 13 shows the ROC curves for each model, displaying the generalization performance of the models on the testing dataset.



(a)



(b)

Figure 13 ROC curves of the models, where 0=Sandstone, 1=Sandstone/Shale, 2=Shale, 3=Limestone, 4=Tuff, 5=Dolomite, 6=Clay. (a) WCRF; (b) XGBoost-BO.

The average F1 Score and AUC indicate that all models performed well, with RF-SMOTE achieving an average AUC of 0.983, WCRF - 0.982, and XGBoost-BO - 0.985, suggesting a high overall classification capability. However, differences arise in handling specific lithologies, especially minority classes like tuff, dolomite, and clay. In terms of the minority classes, WCRF and XGBoost-BO show relatively high F1 Scores (for instance, tuff: 0.864 for WCRF and 0.866 for XGBoost-BO), indicating these models manage imbalanced data effectively. WCRF's use of class weights helped boost performance on these classes, as seen in dolomite, where it achieved a notable F1 Score of 0.871 compared to RF-SMOTE's 0.800. XGBoost-BO also performed robustly, leveraging optimized hyperparameters to improve generalization, as shown by its high AUCs for each class. These results are significant for lithology classification as they demonstrate that both WCRF and XGBoost-BO can handle imbalanced data well, improving classification for underrepresented lithologies.

Combining WCRF and XGBoost-BO in a voting classifier leverages the strengths of both models, creating a more robust classifier for lithology classification. WCRF's weighted class approach allows it to handle imbalanced data effectively by giving more emphasis to minority classes such as tuff, dolomite, and clay, which are underrepresented in the dataset. This weighting mechanism helps WCRF boost its performance on less frequent lithologies, providing a balanced view of different rock types. Meanwhile, XGBoost-BO, optimized through Bayesian methods, brings strong predictive power and flexibility, with the ability to handle complex relationships in well log data due to its boosting nature and fine-tuned hyperparameters.

By combining these models in a voting classifier, the system benefits from WCRF's ability to account for class imbalance alongside XGBoost's high accuracy and generalization capabilities. This method enhances

the overall classification accuracy, ensuring that the classifier is not biased toward majority classes while still maintaining high precision across all lithologies. The combined classifier effectively captures diverse patterns and improves robustness, making it highly relevant for geological applications where accurate classification of all lithologies, including rare ones, is crucial for decision-making in exploration and drilling activities. Table 4 shows the F1 scores of the voting classifier (both hard and soft) with different weights adjustments.

Table 2 Tuned hyperparameters for the XGBoost model

Hyperparameter	Symbol	Search Range	Optimum Value
Number of boosted trees	n_estimators	10-1000	1000
Maximum depth of a tree	max_depth	1-20	20
Boosting learning rate	learning_rate	0.01-0.5	0.01
Subsample ratio of the training instances	subsample	0.1-1	0.791
Min Child weight	min_child_weight	1-5	1
Gamma	gamma	0.1-0.5	0.1
Colsample by tree	colsample_bytree	0.5-1	1
L2 regularization term on weights	reg_alpha	0.001-0.01	0.001

Table 3 Performance results of the machine learning models for lithology classification

	RF-SMOTE		WCRF		XGBoost-BO	
Class	**F1 Score**	**AUC**	**F1 Score**	**AUC**	**F1 Score**	**AUC**
Sandstone	0.929	0.990	0.929	0.992	0.919	0.990
Sandstone/Shale	0.837	0.966	0.848	0.962	0.838	0.969
Shale	0.811	0.964	0.823	0.967	0.803	0.965
Limestone	0.836	0.970	0.814	0.969	0.832	0.977
Tuff	0.853	0.998	0.864	0.998	0.866	0.998
Dolomite	0.800	0.993	0.871	0.990	0.847	0.995
Clay	0.926	0.999	0.911	0.999	0.925	0.998
Avg	**0.856**	**0.983**	**0.866**	**0.982**	**0.861**	**0.985**

Table 4 Performance of the voting classifier (WCRF and XGBoost-BO) with weight adjustments

	Voting Classifier [1,1]		Voting Classifier [2,1]		Voting Classifier [1,2]	
Class	**F1 Score (Hard)**	**F1 Score (Soft)**	**F1 Score (Hard)**	**F1 Score (Soft)**	**F1 Score (Hard)**	**F1 Score (Soft)**
Sandstone	0.923	0.927	0.929	0.929	0.921	0.927
Sandstone/Shale	0.834	0.841	0.849	0.843	0.830	0.839
Shale	0.820	0.835	0.824	0.833	0.824	0.832
Limestone	0.817	0.831	0.815	0.819	0.822	0.831
Tuff	0.893	0.889	0.864	0.880	0.894	0.887
Dolomite	0.878	0.885	0.871	0.885	0.893	0.885
Clay	0.921	0.945	0.911	0.933	0.935	0.945
Avg	**0.869**	**0.879**	**0.866**	**0.875**	**0.874**	**0.878**

Table 4 reveals that the voting classifiers generally yield higher average F1-Scores, suggesting improved performance through ensemble learning. Each column represents a different weighting of WCRF and XGBoost-BO in the voting classifier. For instance, [1,1] uses equal weight for both models, while [2,1] and [1,2] give different weights to the models. Across configurations, soft voting generally outperforms hard voting, with the highest average F1-Score of 0.879 observed in the [1,1] soft voting con-figuration. Examining minority classes—Tuff, Dolomite, and Clay—the voting classifiers exhibit enhanced performance compared to the base models. Tuff and Dolomite benefit from the ensemble approach, showing F1-Scores as high as 0.889 and 0.885, respectively, under the [1,1] soft voting. This improvement in minority class recognition is crucial for geological accuracy, as these classes are less frequently encountered but hold importance in lithology classification. Furthermore, Clay consistently achieves high F1-Scores (e.g., 0.945 in [1,1] soft voting), reflecting the ensemble's strong ability to identify this minority class. The [1,1] soft voting classifier improves the F1-Score by approximately

1.5% compared to WCRF and 2.1% compared to XGBoost-BO.

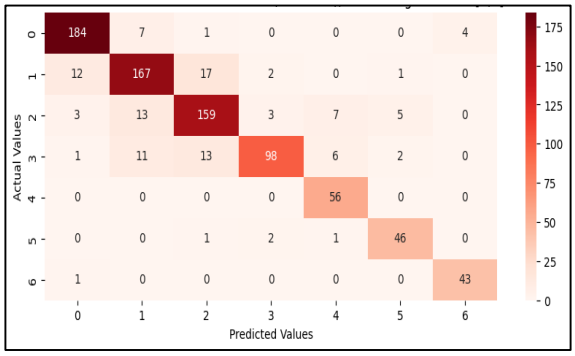


Figure 14 Confusion matrix of [1,1] soft voting classifier, where 0=Sandstone, 1=Sandstone/Shale, 2=Shale, 3=Limestone, 4=Tuff, 5=Dolomite, 6=Clay.

Figure 14 shows that there is notable confusion among closely related lithologies, especially between Sandstone and Sandstone/Shale, Shale and Limestone, and Lime-stone and Tuff. This confusion may stem from overlapping physical or chemical properties that make these classes harder to distinguish. The [1,1] soft voting approach improves overall classification by balancing the individual strengths of its component models, which is especially beneficial for minority classes like Tuff and Clay. This improvement indicates that the ensemble method marginally but effectively enhances model performance.

Conclusions

This study introduced an ensemble approach combining Weighted Class Random Forest (WCRF) and Bayesian Optimized Extreme Gradient Boosting (XGBoost-BO) to enhance lithology classification from well log data. Individually, the WCRF model achieved an average AUC of 0.982 and an F1-Score of 0.866, while XGBoost-BO attained a slightly higher AUC of 0.985 and an F1-Score of 0.861. By combining WCRF and XGBoost-BO in a voting classifier, we were able to leverage the strengths of both models, creating a more robust classifier for lithology classification. The [1,1] soft voting configuration in the ensemble model improved the F1-Score by approximately 1.5% over WCRF and 2.1% over XGBoost-BO, confirming the effectiveness of this ensemble approach.

This method provides a robust, efficient solution for subsurface lithology classification, enhancing accuracy across diverse rock types. The improved prediction accuracy, especially for underrepresented classes, can lead to better-informed decisions in drilling and resource assessment, reducing exploration costs and risks. These findings underscore the value of ensemble machine learning models in the petroleum industry for reliable lithology classification.

Funding Sources

This research received no external funding.

Conflicts of Interest

There are no conflicts to declare.

Acknowledgements

The authors would like to express their gratitude to Peter Bormann, Peder Aursand, Fahad Dilib, Surrender Manral, and Peter Dischington for their contributions to the FORCE 2020 Well Log and Lithofacies Dataset and the associated resources. Their work in creating and sharing this dataset and the accompanying GitHub repository [18] has greatly facilitated advancements in machine learning applications in geosciences.

Notes and References

[1] Harris, J. R., & Grunsky, E. C. (2015). Predictive lithological mapping of Canada's north using random forest classification applied to geophysical and geochemical data. Computers & Geosciences, 80, 9–25. https://doi.org/10.1016/j.cageo.2015.03.013

[2] Logging, C. (2015). Reservoir characteristics of oil sands and logging evaluation methods: A case study from Ganchaigou area, Qaidam Basin. Lithology and Reservoirs, 27, 119–124.

[3] Saporetti, C. M., da Fonseca, L. G., Pereira, E., & de Oliveira, L. C. (2018). Machine learning approaches for petrographic classification of carbonate-siliciclastic rocks using well logs and textural information. Journal of Applied Geophysics, 155, 217–225. https://doi.org/10.1016/j.jappgeo.2018.06.012

[4] Lu, X., Sun, D., Xie, X., Chen, X., Zhang, S., Zhang, S., & Sun, G. (2019). Microfacies characteristics and reservoir potential of Triassic Baikouquan formation, northern Mahu Sag, Junggar Basin, NW China. Journal of Natural Gas Geoscience, 4(1), 47–62. https://doi.org/10.1016/j.jnggs.2019.03.001

[5] Zhao, Z., He, Y., Huang, X., & Zhan, S. (2021). Study on fracture characteristics and controlling factors of tight sandstone reservoir: A case study on the Huagang formation in the Xihu Depression, East China Sea Shelf Basin, China. Lithosphere. https://doi.org/10.2113/2021/3310886

[6] Liu, H., Wu, Y., Cao, Y., Lv, W., Han, H., Li, Z., & Chang, J. (2020). Well logging based lithology identification model establishment under data drift: A transfer learning method. Sensors, 20(13), 3643. https://doi.org/10.3390/s20133643

[7] Anthonio, U., Okereke, N., Vincent, N., & Blessing, E. (2024). Ensemble machine learning for real-time sonic log prediction in geothermal exploration. In Proceedings of the SPE Nigeria Annual International Conference and Exhibition (SPE NAICE 2024), Lagos, Nigeria, 5–7 August. https://doi.org/10.2118/221632-MS

[8] Ekeopara, P., & Nekekpemi, P. (2024). Generative AI: Prospects and applications in geothermal energy. In Proceedings of the 49th Workshop on Geothermal Reservoir Engineering, Stanford University, Stanford, California, USA, 12–14 February.

[9] Ekeopara, P., Odo, J., Obah, B., & Nwankwo, V. (2022). Hybridized probabilistic machine learning ranking system for lithological identification in geothermal resources. In Proceedings of the SPE Nigeria Annual International Conference and Exhibition (SPE NAICE 2022), Lagos, Nigeria, 1–3 August. https://doi.org/10.2118/212015-MS

[10] Obika, F. C., Okereke, N. U., Eze, F. M., & Ekeh, B. C. (2024). Unveiling the potential of random undersampling in geothermal lithology classification for improved geothermal resource exploration. In Proceedings of the SPE Nigeria Annual International Conference and Exhibition (SPE NAICE 2024), Lagos, Nigeria, 5–7 August. https://doi.org/10.2118/221656-MS

[11] Xie, Y., Zhu, C., Lu, Y., & Zhu, Z. (2019). Towards optimization of boosting models for formation lithology identification. Mathematical Problems in Engineering, 2019(1), 5309852. https://doi.org/10.1155/2019/5309852

[12] Sun, Z. X., Jiang, B. S., Li, X. L., Li, J., & Xiao, K. (2020). A data-driven approach for lithology identification based on parameter-optimized ensemble learning. Energies, 13(15), 3903. https://doi.org/10.3390/en13153903

[13] Wang, G., Carr, T. R., Ju, Y., & Li, C. (2014). Identifying organic-rich Marcellus shale lithofacies by support vector machine classifier in the Appalachian Basin. Computers & Geosciences, 64, 52–60. https://doi.org/10.1016/j.cageo.2013.12.002

[14] Xie, Y., Zhu, C., Zhou, W., Li, Z., Liu, X., & Tu, M. (2018). Evaluation of machine learning methods for formation lithology identification: A comparison of tuning processes and model performances. Journal of Petroleum Science and Engineering, 160, 182–193. https://doi.org/10.1016/j.petrol.2017.10.028

[15] Sun, J., Li, Q., Chen, M., Ren, L., Huang, G., Li, C., & Zhang, Z. (2019). Optimization of models for a rapid identification of lithology while drilling: A win-win strategy based on machine learning. Journal of Petroleum Science and Engineering, 176, 321–341. https://doi.org/10.1016/j.petrol.2019.01.006

[16] Saporetti, C. M., & da Fonseca, L. G. (2019). A lithology identification approach based on machine learning with evolutionary parameter tuning. IEEE Geoscience and Remote Sensing Letters, 16(12), 1819–1823. https://doi.org/10.1109/LGRS.2019.2911473

[17] Zhang, J., He, Y., Zhang, Y., Li, W., & Zhang, J. (2022). Well-logging-based lithology classification using machine learning methods for high-quality reservoir identification: A case study of Baikouquan Formation in Mahu Area of Junggar Basin, NW China. Energies, 15(10), 3675. https://doi.org/10.3390/en15103675

[18] Bormann, P., Aursand, P., Dilib, F., Manral, S., & Dischington, P. (2020). FORCE 2020 well log and lithofacies dataset for machine learning competition. Zenodo. https://doi.org/10.5281/zenodo.4351156

[19] File:Norwegian Sea map.png. Available online: https://commons.wikimedia.org/wiki/File:Norwegian_Sea_map.png (accessed on 26 October 2024).

[20] Chan, J. Y.-L., Leow, S. M. H., Bea, K. T., Cheng, W. K., Phoong, S. W., Hong, Z.-W., & Chen, Y.-L. (2022). Mitigating the multicollinearity problem and its machine learning approach: A review. Mathematics, 10(8), 1283. https://doi.org/10.3390/math10081283

[21] Tukey, J. W. (1975). Mathematics and the picturing of data. In Proceedings of the International Congress of Mathematicians, Vancouver, BC, Canada, pp. 523–531.

[22] Breiman, L. (2001). Random forests. Machine Learning, 45(1), 5–32. https://doi.org/10.1023/A:1010933404324

[23] Christiansen, S. D. (2021). Ischemic stroke thrombus characterization through quantitative magnetic resonance imaging. *Doctoral Dissertation*, The University of Western Ontario, London, ON, Canada. http://dx.doi.org/10.13140/RG.2.2.12667.64806

[24] Díaz-Uriarte, R., & Alvarez De Andrés, S. (2006). Gene selection and classification of microarray data using random forest. BMC Bioinformatics, 7(1), 3. https://doi.org/10.1186/1471-2105-7-3

[25] Caruana, R., & Niculescu-Mizil, A. (2006). An empirical comparison of supervised learning algorithms. Journal of Machine Learning Research, 23(1), 161–168. https://doi.org/10.1145/1143844.1143865

[26] Tewari, S., Dwivedi, U. D., & Biswas, S. (2021). A novel application of ensemble methods with data resampling techniques for drill bit selection in the oil and gas industry. Energies, 14(2), 432. https://doi.org/10.3390/en14020432

[27] Friedman, J. H. (2001). Greedy function approximation: A gradient boosting machine. Annals of Statistics, 29, 1189–1232.

[28] Chen, T., & Guestrin, C. (2016). XGBoost: A scalable tree boosting system. In Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, ACM, San Francisco, California, USA, pp. 785–794. https://doi.org/10.1145/2939672.2939785

[29] Reddy, D. K. K., Behera, H. S., Nayak, J., Naik, B., Ghosh, U., & Sharma, P. K. (2021). Exact greedy algorithm-based split finding approach for intrusion detection in fog-enabled IoT environment. Journal of Information Security Applications, 60, 102866. https://doi.org/10.1016/j.jisa.2021.102866

[30] Ramraj, S., Uzir, N., Sunil, R., & Banerjee, S. (2016). Experimenting XGBoost algorithm for prediction and classification of different datasets. International Journal of Control Theory and Applications, 9(40), 651–662.

[31] Cheng, F., Yang, C., Zhou, C., Lan, L., Zhu, H., & Li, Y. (2020). Simultaneous determination of metal ions in zinc sulfate solution using UV–Vis spectrometry and SPSE-XGBoost method. Sensors, 20(17), 4936. https://doi.org/10.3390/s20174936

[32] Dietterich, T. G. (2000). Ensemble methods in machine learning. In Multiple Classifier Systems (pp. 1–15). Springer Berlin Heidelberg. https://doi.org/10.1007/3-540-45014-9_1

[33] Kuncheva, L. I. (2014). Combining pattern classifiers: Methods and algorithms. John Wiley & Sons.
[34] Polikar, R. (2006). Ensemble-based systems in decision making. IEEE Circuits and Systems Magazine, 6(3), 21–45.
[35] Zhou, Z.-H. (2012). Ensemble methods: Foundations and algorithms. CRC Press.
[36] Awan, F. M., Saleem, Y., Minerva, R., & Crespi, N. (2020). A comparative analysis of machine/deep learning models for parking space availability prediction. Sensors, 20(1), 322. https://doi.org/10.3390/s20010322
[37] Chawla, N. V., Bowyer, K. W., Hall, L. O., & Kegelmeyer, W. P. (2002). SMOTE: Synthetic minority over-sampling technique. Journal of Artificial Intelligence Research, 16, 321–357. https://doi.org/10.1613/jair.953
[38] Blagus, R., & Lusa, L. (2013). SMOTE for high-dimensional class-imbalanced data. BMC Bioinformatics, 14(1), 106. https://doi.org/10.1186/1471-2105-14-106
[39] Wang, J., Xu, M., Wang, H., & Zhang, J. (2006). Classification of imbalanced data by using the SMOTE algorithm and locally linear embedding. In Proceedings of the 2006 8th International Conference on Signal Processing, Guilin, China, 16–20 November 2006; Vol. 3. https://doi.org/10.1109/ICOSP.2006.345752
[40] Fernández, A., García, S., Galar, M., Prati, R. C., Krawczyk, B., & Herrera, F. (2018). *Learning from imbalanced data sets*. Springer International Publishing: Cham. https://doi.org/10.1007/978-3-319-98074-4
[41] Rasmussen, C. E., & Nickisch, H. (2010). Gaussian processes for machine learning (GPML) toolbox. Journal of Machine Learning Research, 11, 3011–3015.
[42] Snoek, J., Larochelle, H., & Adams, R. P. (2012). Practical Bayesian optimization of machine learning algorithms. arXiv. https://doi.org/10.48550/arXiv.1206.2944
[43] Muhuri, P. K., & Biswas, S. K. (2020). Bayesian optimization algorithm for multi-objective scheduling of time and precedence constrained tasks in heterogeneous multiprocessor systems. Applied Soft Computing, 92, 106274. https://doi.org/10.1016/j.asoc.2020.106274
[44] Shahriari, B., Swersky, K., Wang, Z., Adams, R. P., & De Freitas, N. (2015). Taking the human out of the loop: A review of Bayesian optimization. Proceedings of the IEEE, 104(1), 148–175. https://doi.org/10.1109/JPROC.2015.2494218
[45] Adams, R. P. (2014). A tutorial on Bayesian optimization for machine learning. Harvard University.
[46] Frazier, P. I. (2018). A tutorial on Bayesian optimization. arXiv. https://doi.org/10.48550/arXiv.1807.02811
[47] Mining, W. I. D. (2006). Data mining: Concepts and techniques. Morgan Kaufmann, 10(559–569), 4.
[48] Kamiran, F., & Calders, T. (2012). Data preprocessing techniques for classification without discrimination. Knowledge and Information Systems, 33(1), 1–33. https://doi.org/10.1007/s10115-011-0463-8
[49] Iglewicz, B., & Hoaglin, D. C. (1993). How to detect and handle outliers (Vol. 16). Quality Press.
[50] Hossin, M., & Sulaiman, M. N. (2015). A review on evaluation metrics for data classification evaluations. International Journal of Data Mining and Knowledge Management Process, 5(2), 1. https://dx.doi.org/10.5121/ijdkp.2015.5201
[51] Vujović, Ž. (2021). Classification model evaluation metrics. International Journal of Advanced Computer Science and Applications, 12(6), 599–606. https://dx.doi.org/10.14569/IJACSA.2021.0120670
[52] Wu, S., & Flach, P. (2005). A scored AUC metric for classifier evaluation and selection. In Second Workshop on ROC Analysis in ML, Bonn, Germany. Citeseer.