A comparative study of boosting algorithms: Concepts, Algorithms, Applications, and Prospects

Mrs.D.Malarvizhi[1], Mrs. KiruthikaS[2]

[1]*Assistant Professor, Department of Computer Science Dr. N.G.P. Arts and Science College Coimbatore*

[2]*Assistant Professor, Department of Information Technology, Sri Ramakrishna College of Arts & Science, Coimbatore*

Abstract: **Recently, a number of intriguing ideas that prioritize accuracy and speed—such as XGBoost, LightGBM, and CatBoost—have added to the family of gradient boosting algorithms. Scalable ensemble method XGBoost has proven to be a dependable and effective machine learning problem solution. LightGBM is a precise model that uses selective selection of high gradient cases to provide incredibly quick training performance. To increase the model's accuracy, CatBoost alters the gradient computation to prevent the prediction shift. This paper offers a useful examination of the performance of these unique gradient boosting variations in terms of training speed, generalization, and hyper-parameter configuration. Furthermore, a thorough comparison of gradient boosting, random forests, LightGBM, XGBoost, and random forests has been carried out utilizing both the models' default settings and highly calibrated models. Despite the few discrepancies, the comparison's findings show that CatBoost outperforms the other algorithms in terms of both AUC and generalization accuracy throughout the datasets under study. While LightGBM is the quickest approach available, it isn't the most precise. In terms of training speed and accuracy, XGBoost comes in second. Lastly, two new techniques are suggested and used to do a thorough examination of the impact of hyper-parameter adjustment in XGBoost, LightGBM, and CatBoost.**

Keywords: **XGBoost, LightGBM, CatBoost, AdaBoost, Gradient boosting**

I INTRODUCTION

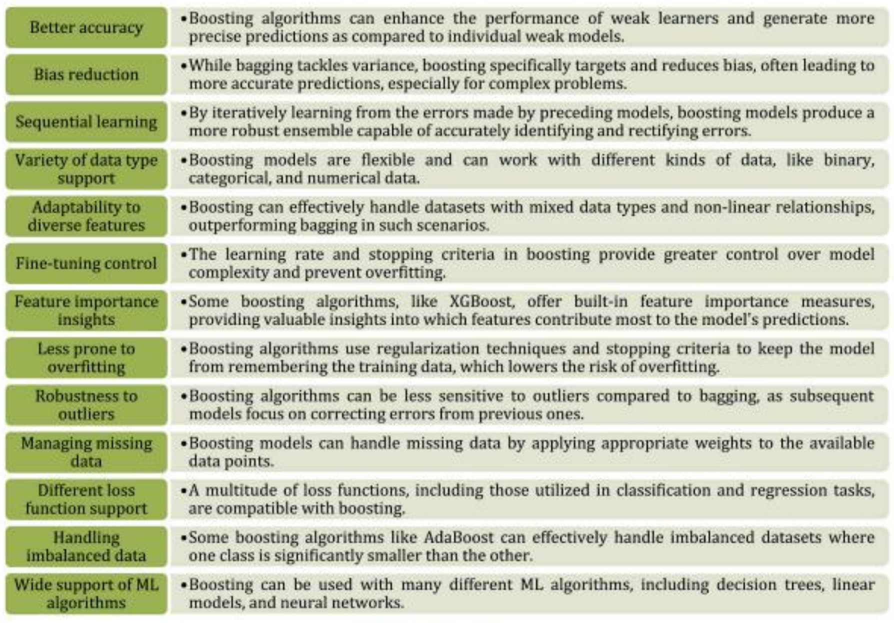
The recent advances in computing power and machine learning (ML) have given rise to several innovations and developments in many areas of research and human lives. In the last few years, advancements in machine learning, a subset of artificial intelligence (AI), have transformed and inherently changed almost every area of our lives [1], [2]. Particularly, ML has been applied in disease diagnosis [3], fraud detection [4], text classification [5], and image recognition [6], among others. Unlike humans, ML algorithms consider several factors when making decisions, and they are not prone to fatigue or prejudice. Meanwhile, learning can sometimes be challenging, especially when learning from high-dimensional and imbalanced datasets [7], [8], [9]. Research has shown that conventional ML algorithms tend to underperform when trained with imbalanced datasets [10]. Therefore, researchers have frequently resorted to new and improved learning approaches, such as ensemble and deep learning.

Ensemble learning and deep learning are two approaches that have dominated the machine learning domain [11], [12], [13]. Ensemble learning methods train multiple base learners and combine their predictions to obtain improved performance and better generalization ability than the individual base learners [14]. The fundamental idea behind ensemble learning is the recognition that machine learning models have limitations and can make errors. Hence, ensemble learning aims to improve classification performance by harnessing the strengths of multiple base models. Meanwhile, some limitations of ML algorithms include: that they result in models with high variance, high bias, and low accuracy [15], [16]. However, several studies have shown that ensemble models often achieve higher accuracy than single ML models [17]. Ensemble methods can limit the variance and bias errors associated with single ML models; for example, bagging reduces variance without increasing the bias, while boosting reduces bias [18], [19], [20]. Overall, ensemble classifiers are more robust and perform better than the individual ensemble learners.

Ensemble learning methods are broadly categorized into boosting, bagging, and stacking [21]. Gradient boosting, XGBoost, and AdaBoost are examples of boosting algorithms, while random forest and Extra Trees classifier are well-known bagging algorithms. Meanwhile, examples of the stacking framework include super ensemble and blending techniques [22], [23], [24]. These ensemble algorithms have achieved excellent performance in various ML applications [25], [26], [27]. Their utilization in many real-world applications is also well-known [28]. Due to their high performance, ensemble methods are the go-to algorithms in many machine learning competitions. When implementing ensemble classifiers, the accuracy and diversity of the base learners are two essential factors that need to be considered [29]. Most ensemble algorithms ensure diversity through data resampling or by altering the structure of the individual learners [30]. Furthermore, base learners that ensure higher accuracy than random guessing are desirable. The individual learners need to have separate knowledge of the task being learned and are also expected to have errors different from the others.

In the past, researchers have conducted some general ensemble learning reviews [31], [32], [33], reviews that focus on ensemble learning for feature selection [34], reviews on ensemble learning in healthcare applications [35], and a review of recent developments and applications of ensemble learning [36]. Other ensemble learning reviews in the literature include those that focus on intrusion detection systems [37], sentiment analysis [38], and big data [39]. Taxonomy for characterizing ensemble methods was presented in [40]. Most published reviews and surveys provide a general overview of ensemble learning and focus on its applications in specific tasks. However, what is lacking in the literature is a succinct explanation of ensemble learning algorithms, their early developments, and concise mathematical and algorithmic representations in one peer-reviewed article. We think it is relevant and timely to have a survey that fills this gap, considering the rise in the popularity of ensemble methods, their application in diverse problems, and the introduction of more recent variants such as XGBoost, LighGBM, and CatBoost.

Therefore, this paper provides a concise yet deep insight into ensemble learning methods and algorithms. The paper aims to differentiate between the three main ensemble methods: bagging, boosting, and stacking. Particular focus is placed on the mathematical and algorithmic representations of the widely used ensemble algorithms, including random forest, gradient boosting, AdaBoost, XGBoost, LightGBM, and CatBoost. This review is for ML researchers and practitioners who intend to understand ensemble learning and the various ensemble algorithms.



Better accuracy	•Boosting algorithms can enhance the performance of weak learners and generate more precise predictions as compared to individual weak models.
Bias reduction	•While bagging tackles variance, boosting specifically targets and reduces bias, often leading to more accurate predictions, especially for complex problems.
Sequential learning	•By iteratively learning from the errors made by preceding models, boosting models produce a more robust ensemble capable of accurately identifying and rectifying errors.
Variety of data type support	•Boosting models are flexible and can work with different kinds of data, like binary, categorical, and numerical data.
Adaptability to diverse features	•Boosting can effectively handle datasets with mixed data types and non-linear relationships, outperforming bagging in such scenarios.
Fine-tuning control	•The learning rate and stopping criteria in boosting provide greater control over model complexity and prevent overfitting.
Feature importance insights	•Some boosting algorithms, like XGBoost, offer built-in feature importance measures, providing valuable insights into which features contribute most to the model's predictions.
Less prone to overfitting	•Boosting algorithms use regularization techniques and stopping criteria to keep the model from remembering the training data, which lowers the risk of overfitting.
Robustness to outliers	•Boosting algorithms can be less sensitive to outliers compared to bagging, as subsequent models focus on correcting errors from previous ones.
Managing missing data	•Boosting models can handle missing data by applying appropriate weights to the available data points.
Different loss function support	•A multitude of loss functions, including those utilized in classification and regression tasks, are compatible with boosting.
Handling imbalanced data	•Some boosting algorithms like AdaBoost can effectively handle imbalanced datasets where one class is significantly smaller than the other.
Wide support of ML algorithms	•Boosting can be used with many different ML algorithms, including decision trees, linear models, and neural networks.

Fig. 1. Advantages of boosting.

II BOOSTING ALGORITHMS

Boosting algorithms Boosting methods use an iterative training process to train base models or weak learners, with an increasing emphasis on misclassified examples in each iteration. Doing it this way, the focus is on fixing the errors made by earlier models, which improves prediction performance, including better accuracy. Many boosting methods have been proposed [41]. Each enhances classification performance by manipulating specific stages of the general boosting scheme. In this study, we considered the following seven boosting algorithms.

A. Boosting

Boosting is a machine learning technique capable of converting weak learners into a strong classifier. It is a type of ensemble meta-algorithm used for reducing bias and variance. Meanwhile, a weak learner is a classifier that performs a bit better than random guessing, while strong learners are those that attain good accuracy [42], and they are the core of which the boosting ensemble algorithms are built. The boosting algorithm was first discussed in a 1990 paper by Schapire [43] in response to a question asked by Kearns and Valiant [44] if a set of weak learners could produce a single strong learner. The work by Schapire [43] had a significant impact on machine learning and statistics, which led to the development of several boosting algorithms, including AdaBoost [69] and XGBoost [45].

The main idea behind boosting involves iteratively applying the base learning algorithm to adjusted versions of the input data [46]. In particular, boosting techniques use the input data to train a weak learner, compute the predictions from the weak learner, select misclassified training samples, and train the subsequent weak learner with an adjusted training set comprising the misclassified instances from the previous training round [47]. The iterative learning process continues until a predefined number of base learners is obtained, and the base learners are weighted together [48]. Boosting focuses more on reducing bias than variance [49]. Therefore, it enhances base learners with a high bias and low variance, such as decision stumps (a decision tree with one internal node). Misclassified samples get more weight, causing the base learner to focus on such samples. Hence, if the base classifier is biased against specific samples, those samples are given more weight; hence, the algorithm corrects the bias. However, this iterative learning approach makes boosting unsuitable for learning noisy data because the weight given to noisy samples is usually much greater than the weights given to the other samples, thereby forcing the algorithm to focus excessively on the noisy samples, resulting in overfitting. Despite that, boosting-based ensemble methods are among the most successful algorithms in applied machine learning [50], [51].

1) AdaBoost

Adaboost functions by dynamically adjusting the weights of weak learners without prior knowledge. During the training process, the error rate of the estimator is used to assess each base learner's weakness. The AB algorithm often uses decision tree stumps to solve classification and regression problems. [53]

The AdaBoost algorithm is a type of boosting algorithm capable of using weak learners to obtain a robust classifier. It was developed in 1995 by Freund and Schapire [54] and is among the most robust ML algorithms. AdaBoost was the first successful boosting algorithm, and the base learners are decision trees having a single split. Because the decision trees are short, they are usually called decision stumps. The most successful AdaBoost implementation is the AdaBoost M1, used for binary classification tasks [55].

The AdaBoost learning process involves training a base classifier using a base algorithm, usually a decision tree. The sample weights are adjusted with respect to the classifier's predictions, and the adjusted samples are employed for training the subsequent classifier. Therefore, the misclassified samples are assigned larger weights and correctly classified instances are assigned lesser weights, ensuring that subsequent classifiers give more attention to the misclassified samples.

The different base learners are added sequentially and weighed to obtain the strong classifier [56]. At every iteration, the AdaBoost algorithm assigns weights to each instance in the training set [57]. Given m labelled training instances S={(x1,y1),…,(xi,yi),…,(xm,ym)} , where yi is the target label of sample xi , and yi∈Y={−1,+1} , the weight D1 of the sample xi and the weight update Dt+1 are computed as:

$$D_1(i) = \frac{1}{n}, i = 1, 2, \ldots, m \tag{3}$$

$$D_{t+1}(i) = \frac{D_t(i)}{z_t} \exp(-\alpha_t y_i h_t(x_i)), \quad i = 1, 2, \ldots, \tag{4}$$

where ht(x) is the base classifier, t=1,…,T is the number of iterations, Zt denotes a normalization factor, while αt is the weight of the classifier ht(x) .

The weight αt measures the importance of the classifier ht(x) when obtaining the final classifier prediction. The instances that are wrongly predicted in ht(x) are assigned larger weights in the t+1 training round. Furthermore, Zt is selected such that Dt+1 will be a distribution. Zt and αt are obtained using:

$$Z_t = \sum_{t+1}^{n} D_t(i) \exp\left(-\alpha_t y_i h_t(x_i)\right) \tag{5}$$

$$\alpha_t = \frac{1}{2} In \left(\frac{1-\epsilon_t}{\epsilon_t}\right) \tag{6}$$

where ϵt represents the error rate of the classifier, and it is obtained using:

$$\epsilon_t = P[h_t(x_i) \neq y_i] = \sum_{i=n}^{n} D_i(i) I[h_t(x_i) \neq y_i] \tag{7}$$

When the given number of iterations have been completed, the final strong classifier is computed using:

$$H(x) = sign\left(\sum_{t=1}^{T} \alpha_t h_t(x)\right) \tag{8}$$

Algorithm 1 AdaBoost.M1 Algorithm

```
Input: training data S=(x1,y1),…,(x2,y2),…,(xm,ym)
The base algorithm L
The number of iterations T .
Procedure:
Initialize the weight D1 of sample xi using (3)
for t=1,…,T :
1.    Train the base classifier ht(x) by minimizing ϵt .
2.    Calculate the weight αt of the classifier using (6).
3.    Update the sample weights using (4)

end for

Output:

Apply (8) to combine the predictions of the base classifiers to obtain the final strong classifier H(x)
```

2) Gradient Boosting

Gradient boosting is a machine learning algorithm that uses the boosting technique to create strong ensembles. It mainly uses decision trees as the base learner to produce a robust ensemble classifier, and it is also called gradient boosted decision tree (GBDT). The gradient boosting technique was first introduced by Breiman [58], who noted that boosting can be represented as an optimization technique on an appropriate loss function.

Subsequently, an extended version of the gradient boosting algorithm was developed by Friedman [59]. The learning process of this algorithm involves sequentially training new models to obtain a robust classifier. It is developed in a step-by-step manner similar to other boosting techniques, but its core idea is to develop base learners that are highly correlated with the negative gradient of the loss function related to the entire ensemble [60].

Gradient boosting (GB) [64] is an ensemble method that builds a predictive model by combining weak learners sequentially, while the weights of each estimator are adjusted individually before being added. The GB algorithm reduces the discrepancy between expected and actual values by fixing residual errors of previous estimators.

Given a training set S={xi,yi}N1 , the gradient boosting technique aims to find an approximation, F^(x) , of a function F∗(x) that maps the predictor variables x to their response variables y , by minimizing the specified loss function L(y,F(x)) . The GBDT creates an additive approximation of F(x) through a weighted sum of functions:

Fm(x)=Fm−1(x)+ρmhm(x) (9)

where ρm represents the weight of the mth function, hm(x) . These functions are the decision tree models in the ensemble. The algorithm performs the approximation iteratively. Meanwhile, a constant approximation of F∗(x) is achieved using:

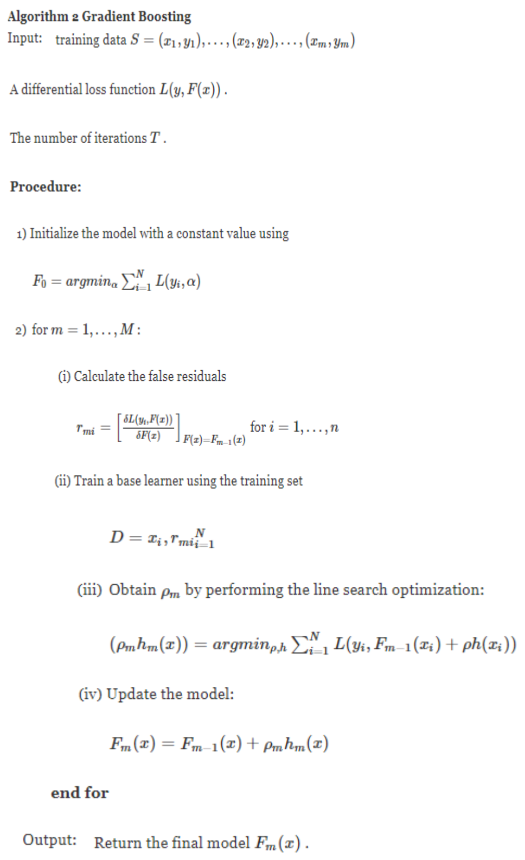
$$F_0(x) = argmin_{\alpha} \sum_{i=1}^{N} L(y_i, \alpha) \tag{10}$$

Successive base learners aim to minimize

$$(\rho_m h_m(x)) = argmin_{\rho,h} \sum_{i=1}^{N} L(y_i, F_{m-1}(x_i) + \rho h(x_i)) \tag{11}$$

Meanwhile, rather than performing the optimization task directly, every hm can be considered a greedy step in a gradient descent optimization for F∗ [61]. Therefore, every hm is trained with a new training set D={xi,rmi}Ni=1 , where rmi represents the false residuals, and it is the difference between the output of an individual base classifier and the actual label [62]. The false residual is also called pseudo residuals, and it is computed as:

$$r_{mi} = \left[\frac{\delta L(y_i, F(x))}{\delta F(x)}\right]_{F(x)=F_{m-1}(x)} \tag{12}$$



```
Algorithm 2 Gradient Boosting
Input:   training data $S = (x_1, y_1), \dots, (x_2, y_2), \dots, (x_m, y_m)$

A differential loss function $L(y, F(x))$ .

The number of iterations $T$ .

Procedure:

 1) Initialize the model with a constant value using

    $F_0 = argmin_\alpha \sum_{i=1}^{N} L(y_i, \alpha)$

 2) for $m = 1, \dots, M$ :

        (i) Calculate the false residuals

            $r_{mi} = \left[\frac{\delta L(y_i, F(x))}{\delta F(x)}\right]_{F(x)=F_{m-1}(x)}$ for $i = 1, \dots, n$

        (ii) Train a base learner using the training set

            $D = x_i, r_{mi}{}_{i=1}^{N}$

        (iii) Obtain $\rho_m$ by performing the line search optimization:

            $(\rho_m h_m(x)) = argmin_{\rho,h} \sum_{i=1}^{N} L(y_i, F_{m-1}(x_i) + \rho h(x_i))$

        (iv) Update the model:

            $F_m(x) = F_{m-1}(x) + \rho_m h_m(x)$

    end for

 Output:   Return the final model $F_m(x)$ .
```

3) XGBoost

The XGBoost algorithm is a decision tree-based ensemble that employs the gradient boosting framework. It is a scalable and highly accurate algorithm used for classification and regression applications. The XGBoost has recently dominated the applied machine learning domain and has won several Kaggle competitions. It was developed in 2016 by Chen and Guestrin [65], having several advancements compared to the conventional gradient boosting algorithm. Unlike the gradient boosting, the XGBoost loss function contains a regularization term that prevents overfitting [66]:

$$L_M(F(x_i)) = \sum_{i=1}^{n} L(y_i, F(x_i)) + \sum_{m=1}^{M} \Omega(h_m) \quad (13)$$

where F(xi) represents the prediction on the i−th instance at the M−th iteration, L(∗) represents a loss function that computes the differences between the predicted class and the actual class of the target variable. Meanwhile, Ω(hm) denotes the regularization term, and it is formulated as:

$$\Omega(h) = \gamma T + \frac{1}{2}\lambda||\omega||^2 \quad (14)$$

where Y represents the complexity parameter, and it controls the minimum loss reduction gain required for splitting an internal node. Assigning a high value to Y leads to simpler trees. Meanwhile, T represents the number of leaves in the tree, λ is a penalty parameter, and ω denotes the output of the leaf nodes. Meanwhile, unlike the first-order derivative in GBDT, a second-order Taylor approximation of the objective function is employed in the XGBoost. Therefore, Equation 13 is transformed thus:

$$L_M \approx \sum_{i=1}^{n}\left[g_i f_m(x_i) + \frac{1}{2} h_i f_m^2(x_i)\right] + \Omega(h_m) \quad (15)$$

where gi and hi denote the first and second derivatives of the loss function. Assuming Ij represents the samples in leaf node j , then the final loss value is calculated by summing the loss values of the various leaf nodes. Hence, the objection function is represented as:

$$L_M = \sum_{j=1}^{T}\left[\left(\sum_{i\in I_j} g_i\right)\omega_j + \frac{1}{2}\left(\sum_{i\in I_j} h_i + \lambda\right)\omega_j^2\right] + \gamma T \quad (16)$$

4) LightGBM

The light gradient boosting machine (LightGBM) is an efficient implementation of the gradient boosting algorithm, and it was developed in 2017 by researchers at Microsoft [67]. It can be used for classification, ranking, and other ML problems. The LightGBM algorithm uses two novel methods, Gradient-based One-Sided Sampling (GOSS) and Exclusive Feature Bundling (EFB), ensuring the algorithm trains faster and achieves high accuracy. The GOSS technique is a modification of the gradient boosting technique, which takes into account the training instances that lead to a larger gradient, thereby making the learning process fast and reducing the computational complexity of the model.

Specifically, the GOSS technique involves excluding a considerable number of training examples with small gradients and uses only the remaining examples to compute the information gain [68]. The reason behind excluding samples with small gradients is that instances with large gradients are more useful in calculating the information gain (IG). Hence, the GOSS technique achieves excellent estimation of the IG with a reduced sample size [67]. Meanwhile, the EFB method performs a feature selection task by bundling sparse mutually exclusive attributes, thereby reducing the number of attributes [69]. Several attributes are almost exclusive in sparse feature space, i.e. they hardly take nonzero values simultaneously [70]. A typical example of exclusive features is One-hot encoded features. Furthermore, the EFB technique bundles such features to reduce the dimension of the feature matrix [71].

The main benefit of the LightGBM is that it is fast and mostly leads to a very efficient model. Secondly, it has a low memory consumption since it converts continuous values to discrete bins. Thirdly, it achieves much higher accuracy than most boosting methods, resulting from the introduction of GOSS and EFB techniques. Lastly, the LightGBM algorithm performs well when trained with large datasets, with a faster training time than the XGBoost algorithm [72]. In terms of disadvantages, the LightGBM can overfit small training datasets easily as it performs better with large datasets. Also, splitting the tree leaf-wise could result in overfitting because more complex trees are produced.

Meanwhile, the LightGBM has been applied for different classification problems, achieving excellent results [73], [74], [75], [76], and its procedure is presented in Algorithm 3. A detailed explanation of the LightGBM technique can be found in [67]. Also, a comprehensive mathematical overview of the LightGBM algorithm is presented in [77].

```
Algorithm 3 LightGBM
Input:   training data $S = (x_1, y_1), \dots, (x_2, y_2), \dots, (x_n, y_n)$

The loss function $L(y, \Theta(x))$

The number of iterations $T$ .

The sampling ratio of large gradient data $a$ , and the sampling ratio of small gradient data $b$ .

Procedure:

 1) Merge mutually exclusive features of $x_i$ , $i = \{1, \dots, N\}$ using the EFB technique.

 2) Initialize $\Theta_0\ (x) = argmin_c \sum_i^N L(y_i, c)$

 3) for t = 1,…,T :

        (i) Compute the absolute values of gradients:

            $r_i = \left[\frac{\delta L(y_i, \Theta(x_i))}{\delta \Theta(x_i)}\right]_{\Theta(x) = \Theta_{t-1}(x)}, i = \{1, \dots, N\}$

        (ii) Resample the dataset using the GOSS technique

            $topN = a \times len(D); randN = b \times len(D); sorted = GetSortedIndices(abs(r));\ A = sor$

             D’ = A + B;

        (iii) Calculate information gain values

            $V_j(d) = \frac{1}{n}\left(\frac{\left(\sum_{x_i \in A_i} r_i + \frac{1-a}{b} \sum_{x_i \in B_i} r_i\right)^2}{n_i^j(d)} + \frac{\left(\sum_{x_i \in A_i} r_i + \frac{1-a}{b} \sum_{x_i \in B_i} r_i\right)^2}{n_i^j(d)}\right)$

        (iv) Obtain a new decision tree $\Theta_t(x)'$ on set $D'$

        (v) Update $\Theta_t\ (x) = \Theta_{t-1}\ (x) + \Theta_t(x)'$

end for

Output:   $\hat{\theta}(x) = \Theta_T(x)$
```

5) CatBoost

The CatBoost algorithm is an implementation of gradient boosting proposed in 2017 by Prokhorenkova *et al.* [78]. The algorithm effectively handles categorical features during the training phase. A notable improvement in CatBoost is its ability to perform unbiased gradient estimation that reduces overfitting. Therefore, in order to estimate the gradient of each example at every boosting iteration, the CatBoost algorithm omits that example from being used to train the current model [79].

Another notable improvement in the CatBoost algorithm is how it automatically transforms categorical features into numerical ones. Categorical features contain a discrete set of values termed categories that are mostly not comparable. Hence, these features are not suitable for building decision trees in their present state. The categorical features are often converted to numerical features at the preprocessing stage, where they are replaced with numerical values. One-hot encoding is the most common method applied to low-cardinality categorical attributes, in which the original feature is replaced with a binary variable.

However, CatBoost applies an improved and more robust approach that does not lead to overfitting and ensures all the examples in the training set are used for training the model. This method involves performing a random permutation of the training set, and for every sample, the algorithm calculates the average label value for the sample with the same category value located before the given one in the permutation [102]. If σ=(σ1,…,σn) is the permutation, then xσp,k is replaced with

$$\frac{\sum_{j=1}^{p-1}[x_{\sigma_j,k} = x_{\sigma_p,k}]Y_{\sigma_j} + a.p}{\sum_{j=1}^{p-1}[x_{\sigma_j,k} = x_{\sigma_p,k}]Y_{\sigma_j} + a} \quad (17)$$

III CONCLUSION

Ensemble learning algorithms have been widely used in numerous classification and regression tasks across a wide range of disciplines, including medical diagnosis, fraud detection, sentiment analysis, and anomaly detection, because of their strong learning capabilities. This paper provides a succinct but thorough introduction to ensemble learning, covering everything from its early inception to the most recent advancements in algorithm technology. One of the primary types of ensemble methods covered in this study is boosting. The popular ensemble algorithms, including random forest, AdaBoost, XGBoost, LightGBM, and CatBoost, are highlighted. For machine learning practitioners and researchers who want to comprehend ensemble learning and the widely used ensemble algorithms, this paper will be essential reading.

REFERENCES

[1] B. van Giffen, D. Herhausen and T. Fahse, "Overcoming the pitfalls and perils of algorithms: A classification of machine learning biases and mitigation methods", J. Bus. Res., vol. 144, pp. 93-106, May 2022.

[2] C. Janiesch, P. Zschech and K. Heinrich, "Machine learning and deep learning", Electron. Markets, vol. 31, no. 3, pp. 685-695, Sep. 2021.

[3] R. T. Aruleba, T. A. Adekiya, N. Ayawei, G. Obaido, K. Aruleba, I. D. Mienye, et al., "COVID-19 diagnosis: A review of rapid antigen RT-PCR and artificial intelligence methods", Bioengineering, vol. 9, no. 4, pp. 153, Apr. 2022.

[4] A. T. Khan, X. Cao, S. Li, V. N. Katsikis, I. Brajevic and P. S. Stanimirovic, "Fraud detection in publicly traded U.S firms using beetle antennae search: A machine learning approach", Expert Syst. Appl., vol. 191, Apr. 2022.

[5] H. Chen, L. Wu, J. Chen, W. Lu and J. Ding, "A comparative study of automated legal text classification using random forests and deep learning", Inf. Process. Manage., vol. 59, no. 2, Mar. 2022.

[6] C. A. Graham, H. Shamkhalichenar, V. E. Browning, V. J. Byrd, Y. Liu, M. T. Gutierrez-Wing, et al., "A practical evaluation of machine learning for classification of ultrasound images of ovarian development in channel catfish (Ictalurus punctatus)", Aquaculture, vol. 552, Apr. 2022.

[7] A. Malekloo, E. Ozer, M. AlHamaydeh and M. Girolami, "Machine learning and structural health monitoring overview with emerging technology and high-dimensional data source highlights", Struct. Health Monitor., vol. 21, no. 4, pp. 1906-1955, Jul. 2022.

[8] X.-F. Song, Y. Zhang, D.-W. Gong and X.-Z. Gao, "A fast hybrid feature selection based on correlation-guided clustering and particle swarm optimization for high-dimensional data", IEEE Trans. Cybern., vol. 52, no. 9, pp. 9573-9586, Sep. 2022.

[9] W. Wang and D. Sun, "The improved AdaBoost algorithms for imbalanced data classification", Inf. Sci., vol. 563, pp. 358-374, Jul. 2021.

[10] I. D. Mienye and Y. Sun, "Performance analysis of cost-sensitive learning methods with application to imbalanced medical data", Informat. Med. Unlocked, vol. 25, Jan. 2021.

[11] A. Mohammed and R. Kora, "An effective ensemble deep learning framework for text classification", J. King Saud Univ. Comput. Inf. Sci., Nov. 2021.

[12] P. Goel, R. Jain, A. Nayyar, S. Singhal and M. Srivastava, "Sarcasm detection using deep learning and ensemble learning", Multimedia Tools Appl., May 2022.

[13] J. Zhou, Z. Jiang, F.-L. Chung and S. Wang, "Formulating ensemble learning of SVMs into a single SVM formulation by negative agreement learning", IEEE Trans. Syst. Man Cybern. Syst., vol. 51, no. 10, pp. 6015-6028, Oct. 2021.

[14] I. D. Mienye, Y. Sun and Z. Wang, "Improved predictive sparse decomposition method with densenet for prediction of lung cancer", Int. J. Comput., vol. 19, no. 4, pp. 533-541, Dec. 2020.

[15] S. Mishra, K. Shaw, D. Mishra, S. Patil, K. Kotecha, S. Kumar, et al., "Improving the accuracy of ensemble machine learning classification models using a novel bit-fusion algorithm for healthcare AI systems", Frontiers Public Health, vol. 10, May 2022.

[16] Y. Sun, Z. Li, X. Li and J. Zhang, "Classifier selection and ensemble model for multi-class imbalance learning in education grants prediction", Appl. Artif. Intell., vol. 35, no. 4, pp. 290-303, Mar. 2021.

[17] G. Brown, "Ensemble learning" in Encyclopedia of Machine Learning, Boston, MA, USA:Springer, pp. 312-320, 2010.

[18] S. Doroudi, "The bias-variance tradeoff: How data science can inform educational debates", AERA Open, vol. 6, no. 4, Oct. 2020.

[19] L. Breiman, "Arcing classifiers", Ann. Stat., vol. 26, no. 3, pp. 801-824, 1998.

[20] S. Alelyani, "Stable bagging feature selection on medical data", J. Big Data, vol. 8, no. 1, pp. 11, Jan. 2021.

[21] K. A. Nguyen, W. Chen, B.-S. Lin and U. Seeboonruang, "Comparison of ensemble machine learning methods for soil erosion pin

measurements", ISPRS Int. J. Geo-Inf., vol. 10, no. 1, pp. 42, Jan. 2021.

[22] M. H. D. M. Ribeiro and L. dos Santos Coelho, "Ensemble approach based on bagging boosting and stacking for short-term prediction in agribusiness time series", Appl. Soft Comput., vol. 86, Jan. 2020.

[23] I. K. Nti, A. F. Adekoya and B. A. Weyori, "A comprehensive evaluation of ensemble learning for stock-market prediction", J. Big Data, vol. 7, no. 1, pp. 20, Mar. 2020.

[24] K. Fawagreh, M. M. Gaber and E. Elyan, "Random forests: From early developments to recent advancements", Syst. Sci. Control Eng., vol. 2, no. 1, pp. 602-609, 2014.

[25] S. Mohammadi, Z. Narimani, M. Ashouri, R. Firouzi and M. H. Karimi-Jafari, "Ensemble learning from ensemble docking: Revisiting the optimum ensemble size problem", Sci. Rep., vol. 12, no. 1, pp. 1-15, Jan. 2022.

[26] E. Esenogho, I. D. Mienye, T. G. Swart, K. Aruleba and G. Obaido, "A neural network ensemble with feature engineering for improved credit card fraud detection", IEEE Access, vol. 10, pp. 16400-16407, 2022.

[27] Q.-F. Li and Z.-M. Song, "High-performance concrete strength prediction based on ensemble learning", Construct. Building Mater., vol. 324, Mar. 2022.

[28] N. C. Oza and K. Tumer, "Classifier ensembles: Select real-world applications", Inf. Fusion, vol. 9, pp. 4-20, Jan. 2008.

[29] C. Perales-González, F. Fernández-Navarro, M. Carbonero-Ruz and J. Pérez-Rodríguez, "Global negative correlation learning: A unified framework for global optimization of ensemble models", IEEE Trans. Neural Netw. Learn. Syst., vol. 33, no. 8, pp. 4031-4042, Aug. 2022.

[30] W. Li, Z. Wang, W. Sun and S. Bahrami, "An ensemble clustering framework based on hierarchical clustering ensemble selection and clusters clustering", Cybern. Syst., pp. 1-26, May 2022.

[31] O. Sagi and L. Rokach, "Ensemble learning: A survey", WIREs Data Mining Knowl. Discovery, vol. 8, no. 4, Jul. 2018.

[32] X. Dong, Z. Yu, W. Cao, Y. Shi and Q. Ma, "A survey on ensemble learning", Frontiers Comput. Sci., vol. 14, no. 2, pp. 241-258, 2020.

[33] R. Polikar, "Ensemble learning" in Ensemble Machine Learning: Methods and Applications, Boston, MA, USA:Springer, pp. 1-34, 2012.

[34] D. Guan, W. Yuan, Y.-K. Lee, K. Najeebullah and M. K. Rasel, "A review of ensemble learning based feature selection", IETE Tech. Rev., vol. 31, no. 3, pp. 190-198, May 2014.

[35] D. Che, Q. Liu, K. Rasheed and X. Tao, "Decision tree and ensemble learning algorithms with their applications in bioinformatics" in Software Tools and Algorithms for Biological Systems, New York, NY, USA:Springer, pp. 191-199, 2011.

[36] Y. Ren, L. Zhang and P. N. Suganthan, "Ensemble classification and regression-recent developments applications and future directions [review article]", IEEE Comput. Intell. Mag., vol. 11, no. 1, pp. 41-53, Feb. 2016.

[37] B. A. Tama and S. Lim, "Ensemble learning for intrusion detection systems: A systematic mapping study and cross-benchmark evaluation", Comput. Sci. Rev., vol. 39, Feb. 2021.

[38] Y. Su, Y. Zhang, D. Ji, Y. Wang and H. Wu, "Ensemble learning for sentiment classification" in Chinese Lexical Semantics, Berlin, Germany:Springer, pp. 84-93, 2013.

[39] B. Krawczyk, L. L. Minkub, J. Gama, J. Stefanowski and M. Woźniak, "Ensemble learning for data stream analysis: A survey", Inf. Fusion, vol. 37, pp. 132-156, Sep. 2017.

[40] L. Rokach, "Taxonomy for characterizing ensemble methods in classification tasks: A review and annotated bibliography", Comput. Statist. Data Anal., vol. 53, no. 12, pp. 4046-4072, 2009.

[41] A.J. Ferreira, M.A.T. Figueiredo, Boosting algorithms: a review of methods, theory, and applications, in: C. Zhang, Y. Ma (Eds.), Ensemble Machine Learning, Springer, Boston, MA, 2012, pp. 35–85.

[42] Y. Sun, Z. Li, X. Li and J. Zhang, "Classifier selection and ensemble model for multi-class imbalance learning in education grants prediction", Appl. Artif. Intell., vol. 35, no. 4, pp. 290-303, Mar. 2021.

[43] R. E. Schapire, "The strength of weak learnability", Mach. Learn., vol. 5, no. 2, pp. 197-227, 1990.

[44] M. Kearns and L. G. Valiant, "Crytographic limitations on learning Boolean formulae and finite automata", Proc. 21st Annu. ACM Symp. Theory Comput. (STOC), pp. 433-444, 1989.

[45] Y. Freund and R. E. Schapire, "A short introduction to boosting", Proc. 16th Int. Joint Conf. Artif. Intell., pp. 1401-1406, 1999.

[46] T. Chen and C. Guestrin, "XGBoost: A scalable tree boosting system", Proc. 22nd ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining, pp. 785-794, Aug. 2016.

[47] I. D. Mienye, Y. Sun and Z. Wang, "Improved predictive sparse decomposition method with densenet for prediction of lung cancer", Int. J. Comput., vol. 19, no. 4, pp. 533-541, Dec. 2020.

[48] I. D. Mienye, G. Obaido, K. Aruleba and O. A. Dada, "Enhanced prediction of chronic kidney disease using feature selection and boosted classifiers" in Intelligent Systems Design and Applications, Cham, Switzerland, pp. 527-537, 2022.

[49] Q.-F. Li and Z.-M. Song, "High-performance concrete strength prediction based on ensemble learning", Construct. Building Mater., vol. 324, Mar. 2022.

[50] N. C. Oza and K. Tumer, "Classifier ensembles: Select real-world applications", Inf. Fusion, vol. 9, pp. 4-20, Jan. 2008.

[51] H. Jafarzadeh, M. Mahdianpari, E. Gill, F. Mohammadimanesh and S. Homayouni, "Bagging and boosting ensemble classifiers for classification of multispectral hyperspectral and PolSAR data: A comparative evaluation", Remote Sens., vol. 13, no. 21, pp. 4405, Nov. 2021.

[52] J. R. Bertini and M. D. C. Nicoletti, "An iterative boosting-based ensemble for streaming data classification", Inf. Fusion, vol. 45, pp. 66-78, Jan. 2019.

[53] Y. Freund, R.E. Schapire, A decision-theoretic generalisation of on-line learning and an application to boosting, J. Comput. Syst. Sci. 55 (1) (1997) 119–139.

[54] Y. Freund and R. E. Schapire, "A short introduction to boosting", Proc. 16th Int. Joint Conf. Artif. Intell., pp. 1401-1406, 1999.

[55] M. Kuhn and K. Johnson, Applied Predictive Modeling, New York, NY, USA:Springer, 2013.

[56] F. Wang, Z. Li, F. He, R. Wang, W. Yu and F. Nie, "Feature learning viewpoint of AdaBoost and a new algorithm", IEEE Access, vol. 7, pp. 149890-149899, 2019.

[57] Y. Freund and R. E. Schapire, "A decision-theoretic generalization of on-line learning and an application to boosting", J. Comput. Syst. Sci., vol. 55, no. 1, pp. 119-139, Aug. 1997.

[58] L. Breiman, "Arcing classifiers", Ann. Stat., vol. 26, no. 3, pp. 801-824, 1998.

[59] J. H. Friedman, "Greedy function approximation: A gradient boosting machine", Ann. Statist., vol. 29, no. 5, pp. 1189-1232, Oct. 2001.

[60] A. Natekin and A. Knoll, "Gradient boosting machines a tutorial", Frontiers Neurorobot., vol. 7, pp. 21, Dec. 2013.

[61] C. Bentéjac, A. Csörgő and G. Martínez-Muñoz, "A comparative analysis of gradient boosting algorithms", Artif. Intell. Rev., vol. 54, no. 3, pp. 1937-1967, Mar. 2021.

[62] R. Caruana, A. Niculescu-Mizil, G. Crew and A. Ksikes, "Ensemble selection from libraries of models", Proc. 21st Int. Conf. Mach. Learn. (ICML), pp. 18, 2004.

[63] J.H. Friedman, Greedy function approximation: a gradient boosting machine, Ann. Stat. 29 (5) (2001) 1189–1232.

[64] Y. Li and W. Chen, "A comparative performance assessment of ensemble learning for credit scoring", Mathematics, vol. 8, no. 10, pp. 1756, Oct. 2020.

[65] T. Chen and C. Guestrin, "XGBoost: A scalable tree boosting system", Proc. 22nd ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining, pp. 785-794, Aug. 2016.

[66] G. Ke, "LightGBM: A highly efficient gradient boosting decision tree", Proc. Adv. Neural Inf. Process. Syst., vol. 30, pp. 3149-3157, 2017, May 9, 2022, [online] Available:

[67] R. Wang, Y. Liu, X. Ye, Q. Tang, J. Gou, M. Huang, et al., "Power system transient stability assessment based on Bayesian optimized LightGBM", Proc. IEEE 3rd Conf. Energy Internet Energy Syst. Integr. (EI2), pp. 263-268, Nov. 2019.

[68] K. Neshatian and L. Varn, "Feature bundles and their effect on the performance of tree-based evolutionary classification and feature selection algorithms", Proc. IEEE Congr. Evol. Comput. (CEC), pp. 1612-1619, Jun. 2019.

[69] Z. Huang and Z. Chen, "Comparison of different machine learning algorithms for predicting the SAGD production performance", J. Petroleum Sci. Eng., vol. 202, Jul. 2021.

[70] Y. Chen, K. Liu, Y. Xie and M. Hu, "Financial trading strategy system based on machine

learning", Math. Problems Eng., vol. 2020, pp. 1-13, Jul. 2020.

[71] D. A. McCarty, H. W. Kim and H. K. Lee, "Evaluation of light gradient boosted machine learning technique in large scale land use and land cover classification", Environments, vol. 7, no. 10, pp. 84, Oct. 2020.

[72] L. Deng, J. Pan, X. Xu, W. Yang, C. Liu and H. Liu, "PDRLGB: Precise DNA-binding residue prediction using a light gradient boosting machine", BMC Bioinf., vol. 19, no. S19, pp. 522, Dec. 2018.

[73] A. A. Taha and S. J. Malebary, "An intelligent approach to credit card fraud detection using an optimized light gradient boosting machine", IEEE Access, vol. 8, pp. 25579-25587, 2020.

[74] Y. Ju, G. Sun, Q. Chen, M. Zhang, H. Zhu and M. U. Rehman, "A model combining convolutional neural network and LightGBM algorithm for ultra-short-term wind power forecasting", IEEE Access, vol. 7, pp. 28309-28318, 2019.

[75] J. Song, G. Liu, J. Jiang, P. Zhang and Y. Liang, "Prediction of protein–ATP binding residues based on ensemble of deep convolutional neural networks and LightGBM algorithm", Int. J. Mol. Sci., vol. 22, no. 2, pp. 939, Jan. 2021.

[76] Y. Wang and T. Wang, "Application of improved LightGBM model in blood glucose prediction", Appl. Sci., vol. 10, no. 9, pp. 3227, May 2020.

[77] L. Prokhorenkova, G. Gusev, A. Vorobev, A. V. Dorogush and A. Gulin, "CatBoost: Unbiased boosting with categorical features", Proc. Adv. Neural Inf. Process. Syst., vol. 31, pp. 6639-6649, 2018, May 22, 2022, [online]

[78] J. Tanha, Y. Abdi, N. Samadi, N. Razzaghi and M. Asadpour, "Boosting methods for multi-class imbalanced data classification: An experimental review", J. Big Data, vol. 7, no. 1, pp. 70, Sep. 2020.