Pakistan Journal of Scientific Research, PJOSR
ISSN (p): 0552-9050
Volume: 3, Number: 1, Pages: 26- 31, Year: 2023

XGBoost and Random Forest Algorithms: An In-Depth Analysis

Sana Fatima, Ayan Hussain, Sohaib Bin Amir, Syed Haseeb Ahmed, Syed Muhammad Huzaifa Aslam

Software Engineering Department, NED University of Engineering & Technology, Karachi, Pakistan.
Corresponding author: Sana Fatima (e-mail: sanafatima@cloud.neduet.edu.pk)

Received: 15/01/2023, Revised: 20/05/2023, Accepted: 30/06/2023

Abstract- **Machine learning is increasingly important in many facets of our lives as technology develops, including forecasting weather, figuring out social media trends, and predicting prices on the world market. This significance invoked the demand for efficient predicting models that can easily handle complex data and provide maximum accurate results. XGBoost and Random Forest are upgradable ensemble techniques used to solve regression and classification problems that have evolved and proved to be dependable machine learning challenge solvers. In this research paper, we comprehensively analyze and compare these two prominent machine learning algorithms. The first half of the research includes a relevant overview of both technique's significance and the evolution of both algorithms. The latter part of this study involves a meticulous comparative analysis between Random Forest and XGBoost, scrutinizing facets such as time complexity, precision, and reliability. We examine their distinctive approaches to handling regression and classification problems while closely examining their subtle handling of training and testing datasets. A thorough quantitative evaluation using a variety of performance metrics, such as the F1-score, Recall, Precision, Mean Squared Error, and others, concludes this discussion.**

Index Terms-- **Classification, Ensemble learning, Machine Learning, Random Forest, Regression, XGBoost.**

I. INTRODUCTION

With technological advancement, machine learning has also gained significance in many applications like healthcare, sports analysis, weather prediction, health insurance, social media analytics, global market price prediction, etc. This significance invoked the demand for efficient predicting models that can easily handle complex data and provide maximum accurate results. Ensemble learning is one of the convenient methods for supervised and unsupervised learning, which predominantly works on the principle of randomization [1-2]. The common thing in the aforementioned applications is that they all use ensemble learning to achieve their required outputs. XGBoost and Random Forest are the two advanced ensemble methods that provide efficient results. Random Forest is a paradigm-shifting invention in ensemble learning, particularly in the context of bagging. In this clever method, the final prediction judgment results from a synthesis[1] of the outputs produced by a large number of individual weak learners. Random Forest strategically uses a subset of data for each decision-making iteration instead of considering every feature, which is what sets it apart from traditional bagging techniques. Random Forest is given the ability to overcome the limitations of conventional bagging techniques and open up fresh possibilities for improved prediction accuracy and robustness [3]. Extreme Gradient Boosting, or XGBoost, is a sophisticated development in gradient boosting, reinforced with various supplemental characteristics. This particular iteration stands out for its exceptional execution speed, improved model performance, and a wide range of characteristics, including parallelization, Core Computing, and Cache Optimization. Combining these features creates an ensemble of unmatched precision, resulting in forecasts that ring true with the highest degree of accuracy. Notably, XGBoost achieves superior prediction performance and masters the world of loss function reduction, utilizing its ability to identify the best strategies for reducing prediction errors [4-6].

This study expands on the foundation established by earlier solitary research projects on machine learning techniques. The article goes even further by conducting a thorough analysis and comparison and presenting a theoretical evaluation of various factors and their effectiveness. The study uses the same real-world test and training datasets for both algorithms to highlight the practical relevance and applicability of these techniques, and it includes a range of performance metrics designed for regression and classification predictive modelling problems.

II. LITERATURE REVIEW

The two major types of ensemble learning are bagging and boosting. Michael Kearns (1988) stated the goal of boosting ensemble learning as “An efficient algorithm for converting

relatively poor hypotheses into very good hypotheses". Hence by following this approach, the first boosting algorithm, Adaptive Boosting (also known as AdaBoost) came into being, this is considered as the first successful boosting algorithm. AdaBoost is a particular sort of self-adaptive Boosting technique that improves the performance of weak classifiers by creating a set of numerous classifiers [7-8]. An extensive variety of concerns have been raised due to the fact that it automatically adjusts to the fundamental algorithm's error rate during training by dynamically regulating the weight of each sample. The theoretical examination of the PAC (Probably Approximately Correct) learning model served as the foundation for the boosting approach. The ideas of strong learning and weak learning were first put out by Kearns and Valiant. In the PAC learning model, a group of concepts is considered to have strong learning if a polynomial learning algorithm exists to identify them and their recognition accuracy is very high; however, if their rate of correct identification is only marginally higher than that of random guessing, they are considered to have weak learning [8]. The weak learners used in this algorithm have only one split and are known as Decision Stumps. Training of weak learners is done sequentially and more priority is given by the weak learners to the features which have many flaws, before passing through the next stump [9]. Leo Breiman, in 1998, introduced the idea of the loss function, a function that directs the algorithm's iterative process of assembling a group of weak learners into a strong learner, in AdaBoost, and in the year 1999, Jerome Friedman fabricated Gradient Boosting, by generalizing the entire boosting algorithm which differs from traditional boosting in adjusting weights, handling errors, and optimizing the model.

During this time span (from 1988 to 1999), research was being done to overcome the major drawback of unexpected complexity made by classifiers of traditional methods as the classifiers were not providing accurate results at such conditions. The method of random decision forests, pioneered by Ho in 1995, involves growing a collection of trees with oblique hyperplane splits that can increase accuracy without overfitting by limiting sensitivity to select feature dimensions. Other splitting techniques produced similar results, contradicting the theory that increasing classifier complexity causes overfitting. Kleinberg's theory explains this resistance to overtraining. Random subset decision-making for single tree growth proposed by Amit and Geman, as well as Ho's notion of random subspace selection had an impact on Breiman's invention of random forests. This involves growing a forest by projecting data into random subspaces and introducing variation. Randomized node optimization was another key concept from Dietterich. With the use of CART-like techniques, randomized node optimization, and bagging, Breiman's article formally established random forests [10-11].

In parallel to the advancements in boosting and random forests, another significant development in ensemble learning was the emergence of stacking, a technique introduced by David Wolpert in 1992. Stacking takes a different approach by combining the outputs of multiple diverse base models through a meta-learner to enhance overall predictive performance. Unlike bagging and boosting, stacking involves a two-level architecture where the first level consists of the individual base models that make predictions on the data. These predictions are then used as inputs for the second-level meta-learner, which learns to combine the base models' outputs into a final prediction. Stacking aims to exploit the strengths of different models and compensate for their weaknesses, effectively creating a hybrid model that can capture more complex relationships within the data [12].

III. MATHODOLOGY

A. DATA COLLECTION

For Classification, The Titanic dataset was taken, it contains information about passengers aboard the RMS Titanic, including details such as their age, gender, class, and survival status. Two subsets of the data—one for training and the other for testing—were taken from the Kaggle platform. The testing fraction lacks survival labels whereas the training subset contains labeled data with survival outcomes, making it difficult for the models to generate reliable predictions. The dataset provides a wide range of attributes that can be used as classification algorithm inputs. This dataset was chosen because of its historical importance and ability to serve as an example when testing classification models.

Table I (below) represents the Titanic data set and its description.

TABLE I
TITANIC DATASET FEATURES

Features	Description
PassengerId	Unique identifier for each passenger
Pclass	Passenger class (1st, 2nd, 3rd)
Name	Passenger's name
Sex	Passenger's gender
Age	Passenger's age
SibSp	Number of siblings/spouses aboard
Parch	Number of parents/children aboard
Ticket	Ticket number
Fare	Fare paid for the ticket
Cabin	Cabin number
Embarked	Port of embarkation (C, Q, S)
Survived	Survival status (0 = No, 1 = Yes)

Furthermore, the California Housing dataset was chosen for regression analysis. It includes housing data for several Californian regions, including characteristics like median income, average rooms, and more. The dataset is essential for testing regression methods because it was originally derived from the 1990 U.S. Census. Its special qualities offer a wide range of property price-influencing elements. Accessibility of the dataset within the scikit-learn framework facilitates data retrieval and permits immediate integration into the pipeline for comparative analysis.

Table II (below) represents California Housing Dataset Features.

TABLE II
CALIFORNIA HOUSING DATASET FEATURES

Features	Description
MedInc	Median income in the district
HouseAge	Median age of houses in the district
AveRooms	Average number of rooms in houses
AveBedrms	Average number of bedrooms in houses
Population	Population in the district
AveOccup	Average occupancy per household
Latitude	Latitude coordinate of the district
Longitude	Longitude coordinate of the district
MedHouseVal	Median house value in the district

B. PERFORMANCE METRICS

The evaluation of model performance in our comparison of XGBoost and Random Forest takes into account a variety of indicators appropriate for both classification and regression tasks. For classification, important metrics like accuracy, precision, recall, and F1-score offer a thorough grasp of the models' capacity to categorize instances accurately, strike a balance between positive and negative predictions, and reduce false positives and false negatives. In contrast, measures such as Mean Squared Error (MSE) and R-squared are crucial in evaluating the precision and goodness-of-fit of the model predictions in regression, allowing for the evaluation of the model's capacity to identify underlying relationships in continuous data. The selection of the most appropriate algorithm for certain use cases and datasets is made easier with the help of these performance measures, which jointly provide insights into the advantages and disadvantages of XGBoost and Random Forest in handling various task types.

Figure 1 (below) shows the work flow of entire process from data collection till performance indicators by using anyone above mentioned algorisms.

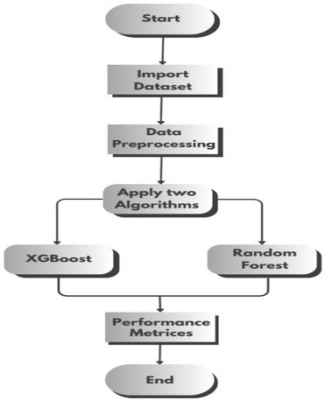

FIGURE1. Work-flow of Analysis OF XGBOOSTAND RANDOM FOREST ALGORITHMS.

IV. COMPARATIVE ANALYSIS

A. UNBALANCED DATASET

For unbalanced datasets, XGBoost is an excellent alternative, but we can't trust random forest in these situations. The classes will most likely be unbalanced in applications such as counterfeiting and fraud detection have many legitimate transactions compared to malicious transactions. When the XGBoost model is not able to predict accurately for the first time then it is given more priority and weight in subsequent iterations, enhancing its capability for predicting low-participation classes. Still, we cannot guarantee that Random Forest will properly deal with class imbalances.

XGBoost adopts a proactive stance by customizing its learning process to give priority to underrepresented classes, making it a viable option for applications with a class imbalance. Random Forest, on the other hand, has an inherent mechanism to effectively manage class imbalance, which may result in less accurate predictions in situations when there is a major skew in the classes. This analysis highlights the value of matching algorithmic capabilities with the properties of the available dataset, enabling sensible decisions in practical applications.

B. SIMILARITY SCORE

In machine learning, the domain similarity score is the dimension representing the object features. If the similarity score is low, it means that the features have a small distance. XGBoosting trims off the decision tree with a Similarity Score, before the real modeling purposes. XGBoost scans the information gain of a node in a decision tree to find the difference between a node's similarity score and a child's similarity score. If the information gained from a node is the minimum then it ceases constructing the decision tree to a larger depth which can control the testing error problems, whereas if the Random Forest decision trees are provided with the same dataset the model will show a high testing error. When the test data is introduced, the model will collapse if the trees are fully developed. As a result, substantial emphasis is devoted to distributing all of the sample's elementary units to all trees with roughly equal participation. Utilizing the "Similarity Score" as a pre-pruning criterion by XGBoost shows a proactive approach by limiting tree development when the gain is limited.

On the other hand, the focus on evenly distributed data among trees in Random Forest emphasizes a preventative strategy against overfitting, which can be brought on by uneven data distribution. It helps to make an informed decision about which ensemble techniques to use and how to use them based on the unique properties of the dataset at hand when you are aware of the subtle mechanics that underlie these algorithms.

C. HYPERPARAMETER TUNING

One of the most significant differences between Random Forest and XGBoost algorithms is that in XGBoost more priority is given to the functional space as far as the reduction of model cost is concerned, on the other hand, hyperparameters are given more priority in Random Forest. In Random Forest all trees get affected by small variations in hyperparameter which can lead to inappropriate prediction. So when test data is expected with a large number of changes with a preconceived intention of

hyperparameter for the whole forest, this approach is not a good option. While in XGBoost only the tree initially works through hyperparameter and logically adapts at the beginning of the iteration. Also, XGBoost takes a small number of initial parameters when it is compared the Random Forest [13-15].

The dynamic allocation of hyperparameters to specific trees in XGBoost promotes adaptability, enabling robust performance in response to changing test data conditions. Contrarily, Random Forest faces difficulties when incorporating a variety of real-time inputs due to its interrelated hyperparameter influence. This investigation highlights the complex interaction between algorithmic design and hyperparameter tuning and emphasizes the significance of matching these elements to the particular requirements of the application area.

D. EFFECT OF LEAF NODE

In Random Forest, the developers are made to add more features to data to ensure how the algorithm works to that given data because there are many decision trees with equal leaf nodes to obtain efficient accuracy with available data. While in XGBoost the number of leaf nodes doesn't matter. If the predictability of the model is not up to the mark, then the algorithm adds more leaf sequentially in the decision tree which discards the biasness to the large extent and the result completely supports the given data [16].

In order to maximize accuracy, Random Forest uses many trees with consistent leaf nodes, which calls for in-depth feature engineering. While this is going on, XGBoost takes a more flexible approach, closely matching the intrinsic properties of the data while iteratively changing leaf nodes to improve model prediction. This subtle investigation highlights the complex interactions between algorithmic tactics, leaf-node manipulation, and the quest for precise predictions.

E. OVERFITTING

In XGBoost the overfitting is avoided by automatically selecting a flex point which decreases the performance of the dataset and increases the performance of the training set continuously as the overfit starts. The loss function which is used in training the model is the measure performance. Few trees in a Random Forest can lead to overfitting which can easily be indicated because a Random Forest implemented with one tree is the same as a single tree [17 - 18]. The overfitting decreases as more trees are added to the Random Forest but this overfitting can never be approached to zero. Random Forest handles error minimization by reducing variance. To decrease the variance the trees are made independent, but biasedness cannot be reduced by the algorithm.

By dynamically adjusting flex points, XGBoost adds adaptability while balancing training efficiency with dataset size. On the other side, Random Forest uses the power of ensemble to minimize variation and reduce overfitting, however complete eradication is still impossible. The challenge of controlling overfitting in the context of several algorithmic approaches is highlighted by this investigation.

F. HANDLING MISSING VALUES

XGBoost demonstrates robustness in handling missing data by adding built-in techniques to handle such situations during model training. XGBoost eliminates the requirement for explicit imputation steps by managing missing values and having the capacity to learn from missingness [19-20]. The management of missing data, on the other hand, necessitates preprocessing steps in Random Forest, which frequently calls for methods like mean imputation or surrogate splits to account for missing values during tree construction. This difference demonstrates XGBoost's competitive edge in its ability to smoothly incorporate missing values and learn from them, hence reducing the potential negative effects of data gaps on model performance. In contrast, Random Forest relies on additional imputation methods, which raises the bar for preprocessing complexity. Understanding these various methods for handling missing values ultimately helps in choosing the best algorithm for datasets with data gaps, which results in more accurate and trustworthy predictive modeling.

G. HANDLING NON-LINEARITY

XGBoost, known for its ensemble capabilities, skillfully handles non-linearity by combining decision trees and boosting approaches. Multiple weak learners are combined using XGBoost, which naturally detects complex non-linear patterns in the data [21-22]. Because boosting is an iterative process, XGBoost may effectively represent non-linear interactions by improving predictions via repeated iterations. This allows XGBoost to focus increasing emphasis on situations with challenging outcomes. Furthermore, XGBoost's feature transformations, such as discretization and quantile-based binning, increase its ability to recognize complicated relationships, which improves its ability to move through non-linear domains.

In contrast, Random Forest's strategy for dealing with non-linearity is based on the diversity of its trees and its ensemble structure. Random Forest takes advantage of the collective by building many decision trees on bootstrapped data.

V. EXPERIMENTAL RESULTS

Both XGBoost and Random Forest can be used for predictive regression and classification modeling. Two distinct datasets have been used for detailed analysis of XGBoost and Random Forest in order to check which algorithm is better for regression and classification techniques.

A. CLASSIFICATION

Due to its iterative structure, which enables it to adaptively improve accuracy over iterations, XGBoost emerges as a front-runner for classification. In contrast, Random Forest has a tendency to handle class imbalances efficiently, but it may have issues with computing performance in real-time predictions, especially when there are a lot of trees. Table III (below) represents the result derived from XGBoost and Random Forest algorithms for classification.

TABLE III
TABULAR COMPARISON OF CLASSIFICATION MODELS

Metrics	XGBoost	Random Forest
Accuracy	0.82	0.80
Precision	0.83	0.83
Recall	0.72	0.66
F1-score	0.77	0.74

a) ACCURACY RATE

The accuracy rate of XGBoost was 0.82, which was marginally better than random forest's accuracy of 0.80. This shows that XGBoost's predictions in this particular dataset were more accurate, demonstrating its capacity to recognize underlying patterns and relationships.

b) PRECISION

Both XGBoost and Random Forest showed remarkable precision levels of 0.83. This implies that both algorithms were equally proficient at accurately recognizing positive situations, highlighting their competence in limiting false positive predictions.

c) RECALL EVALUATION

Recall, which is sometimes referred to as sensitivity or the true positive rate, measures how well the model can find all pertinent instances in the dataset. With a recall of 0.72, XGBoost outperformed Random Forest, which had a recall rate of 0.66. This suggests that XGBoost was more adept at minimizing false negatives because it had a higher capacity to identify and appropriately label positive events.

These results demonstrate that XGBoost performed better in terms of recall, indicating that it can discover relevant examples thoroughly even when there are more instances to classify. According to this result, XGBoost may be better appropriate for applications where thorough identification of positive cases is important.

d) F1- SCORE

The F1-score, a harmonic mean of precision and recall, offers a balanced perspective on the model's overall performance. Random Forest received an F1 score of 0.74, whereas XGBoost received a score of 0.77. This lends greater credence to the idea that XGBoost excelled at striking the right balance between precision and recall, resulting in a more thorough and precise model evaluation.

On the basis of results, we can say that, the analysis of the metrics for accuracy, precision, recall, and F1-score show that XGBoost performed marginally better than Random Forest in the context of the studied dataset. The subtle variations in these metrics highlight the algorithms' advantages and provide information about how well-suited they are to various use cases. Figure 2 (below) shows the evaluation metrics analysis.

FIGURE 2.Comparative analysis of evaluation matrix.

B. REGRESSION

Regression analysis reveals the superior performance of XGBoost, which is especially clear in its reduced Mean Squared Error and higher R-squared values. This suggests that it has a remarkable capacity for capturing intricate correlations in continuous data. Even if it is capable, Random Forest displays lower R-squared values and a relatively greater Mean Squared Error, which indicates a substantially larger prediction error and less variance explained in regression scenarios. Table IV (below) shows the result derived from XGBoost and Random Forest algorithms for Regression.

TABLE IV
TABULAR COMPARISON OF REGRESSION MODELS

Models	MEAN SQUARED ERROR	R-squared
XGBoost	0.29	0.77
Random Forest	0.60	0.54

a) MEAN SQAURED ERROR

XGBoost demonstrated its skill in minimizing the squared disparities between anticipated and actual values by producing a noticeably reduced MSE of 0.29. However, Random Forest had a significantly higher MSE of 0.60, indicating a far higher level of prediction inaccuracy. These results highlight XGBoost's strong regression skills and demonstrate its capacity to produce predictions that are less variable from real values

b) R-SQUARED EVALUATION

The R-squared statistic, also known as the coefficient of determination, is a useful indicator of how effectively the regression model captures the variance of the dependent variable. In the research that was given, XGBoost showed an impressive R-squared value of 0.77, indicating that the model can account for around 77% of the variance in the dependent variable. In contrast, Random Forest produced an R-squared value of 0.54 and indicated that it explained about 54% of the data variance. This demonstrates XGBoost's better performance in regression modeling and further supports its ability to identify and explain the correlations between variables.

To summarize the above results we can say that, the comparison of regression measures demonstrates that XGBoost

is more adept at producing precise predictions than Random Forest, with a lower Mean Squared Error and a higher R-squared value. These outcomes highlight the capability of XGBoost for accurate regression modeling, highlighting its benefits in reducing prediction errors and identifying underlying patterns in the data.

VII. CONCLUSION

This paper compares and contrasts the two machine learning algorithms, XGBoost and Random Forest, by comparing their valuable features such as overfitting, hyperparameter tuning, and the impact of leaf nodes, handling missing values, classification, and regression. By rigorously examining their performance characteristics, this study has revealed each algorithm's nuanced strengths and weaknesses. The detailed analysis of these two machine learning algorithms concludes that XGBoost has the upper hand over Random Forest in multiple dimensions. Notably, its iterative approach and enhanced accuracy make it a formidable choice in predictive modelling tasks. On the other hand, Random Forest's main limitation lies in its computational efficiency for real-time predictions, particularly with many trees. Furthermore, its usefulness may be constrained by the bias toward characteristics with higher levels in the presence of categorical variables with different levels.

To conclude the research, the thorough comparison of XGBoost and Random Forest reveals that XGBoost contains most of the admirable attributes due to its accurate iterative process. The knowledge gained from this research gives programmers a better grasp of the characteristics that make XGBoost a desirable choice in various situations, enabling wise algorithm selection based on particular needs.

ACKNOWLEDGMENT

We are deeply grateful to thank our primary supervisor, [Ms. Sana Fatima], for their guidance, support, and encouragement throughout the entire process. Her mentorship and expertise were invaluable in helping us to shape the direction of our research and to bring our ideas to fruition.

FUNDING STATEMENT

The author(s) received no specific funding for this study.

CONFLICTS OF INTEREST

The authors declare they have no conflicts of interest to report regarding the present study.

REFERENCES

[1] Bentéjac, C., & Csörgő, A. (2019). A Comparative Analysis of XGBoost. Available: https://doi.org/10.48550/arXiv.1911.01914

[2] Ren, Qiong & Cheng, Hui & Han, Hai. (2017). Research on machine learning framework based on random forest algorithm. AIP Conference Proceedings. 1820. 080020. 10.1063/1.4977376.

[3] Couronné, R., Probst, P. & Boulesteix, AL. Random forest versus logistic regression: a large-scale benchmark experiment. BMC Bioinformatics 19, 270 (2018). Available: https://doi.org/10.1186/s12859-018-2264-5

[4] Chen, T., & Guestrin, C. (2016). XGBoost: A Scalable Tree Boosting System. Available: https://doi.org/10.1145/2939672.2939785

[5] Cutler, A., Cutler, D.R., Stevens, J.R. (2012). Random Forests. In: Zhang, C., Ma, Y. (eds) Ensemble Machine Learning. Springer, New York, NY. Available: https://doi.org/10.1007/978-1-4419-9326-7_5

[6] Bentéjac, C., Csörgő, A. & Martínez-Muñoz, G. A comparative analysis of gradient boosting algorithms. Artif Intell Rev 54, 1937–1967 (2021). Available: https://doi.org/10.1007/s10462-020-09896-5

[7] Mayr, A., Binder, H., Gefeller, O., & Schmid, M. (2014). The Evolution of Boosting Algorithms - From Machine Learning to Statistical Modelling. Available: https://doi.org/10.3414/ME13-01-0122

[8] Wu, P., Zhao, H. (2011). Some Analysis and Research of the AdaBoost Algorithm. In: Chen, R. (eds) Intelligent Computing and Information Science. ICICIS 2011. Communications in Computer and Information Science, vol 134. Springer, Berlin, Heidelberg. Available: https://doi.org/10.1007/978-3-642-18129-0_1

[9] Bakırlı, Gözde & Birant, Derya. (2017). DTreeSim: A new approach to compute decision tree similarity using re-mining. TURKISH JOURNAL OF ELECTRICAL ENGINEERING & COMPUTER SCIENCES. 25. 108-125. 10.3906/elk-1504-234.

[10] Fawagreh, Khaled & Gaber, Mohamed & Elyan, Eyad. (2014). Random Forests: From Early Developments to Recent Advancements. Systems Science & Control Engineering. 2. Available: https://doi.org/10.1080/21642583.2014.956265

[11] Joharestani, Mehdi & Cao, Chunxiang & Ni, Xiliang & Bashir, Barjeece & Talebiesfandarani, Somayeh. (2019). PM2.5 Prediction Based on Random Forest, XGBoost, and Deep Learning Using Multisource Remote Sensing Data Available: https://doi.org/10.3390/atmos10070373

[12] R. Vijaya Kumar Reddy, Dr. U. Ravi Babu, "A Review on Classification Techniques in Machine Learning", India, 2018.

[13] Probst, P., Wright, M., & Boulesteix, A. (2018).Hyperparameters and Tuning Strategies for Random Forest. Available: https://doi.org/10.1002/widm.1301

[14] R. García Leiva, A. Fernández Anta, V. Mancuso and P. Casari, "A Novel Hyperparameter-Free Approach to Decision Tree Construction That Avoids Overfitting by Design," in IEEE Access, vol. 7, pp. 99978-99987, 2019, Available: https://ieeexplore.ieee.org/document/8767915

[15] Casper Bøgeskov Hansen, Ahmed Magdy Mahmoud Ali, Mads Zeuch Ethelberg, Martin Moltke Wozniak, "Comparing Random Forest, XGBoost and Neural Networks With Hyperparameter Optimization by Nested Cross-Validation", ROSKILDE UNIVERSITy, 6TH SEMESTER, SPRING 2019

[16] Santhanam, Ramraj & Uzir, Nishant & Raman, Sunil & Banerjee, Shatadeep. (2017). Experimenting XGBoost Algorithm for Prediction and Classification of Different Datasets.

[17] Ying, Xue. (2019). An Overview of Overfitting and its Solutions. Journal of Physics: Conference Series. 1168. 022022. Available: 10.1088/1742-6596/1168/2/022022

[18] Maulud, Dastan & Mohsin Abdulazeez, Adnan. (2020). A Review on Linear Regression Comprehensive in Machine Learning. Journal of Applied Science and Technology Trends. 1. 140-147. Available: 10.38094/jastt1457

[19] Rusdah, D.A., Murfi, H. XGBoost in handling missing values for life insurance risk prediction. SN Appl. Sci. 2, 1336 (2020). Available: https://doi.org/10.1007/s42452-020-3128-y

[20] S. Kabiraj et al., "Breast Cancer Risk Prediction using XGBoost and Random Forest Algorithm," 2020 11th International Conference on Computing, Communication and Networking Technologies (ICCCNT), Kharagpur, India, 2020, pp. 1-4, Available: https://ieeexplore.ieee.org/document/9225451

21] S. Jhaveri, I. Khedkar, Y. Kantharia and S. Jaswal, "Success Prediction using Random Forest, CatBoost, XGBoost and AdaBoost for Kickstarter Campaigns," 2019 3rd International Conference on Computing Methodologies and Communication (ICCMC), Erode, India, 2019, pp. 1170-1173, Available: https://ieeexplore.ieee.org/abstract/document/8819828

[22] D. Zhang, L. Qian, B. Mao, C. Huang, B. Huang and Y. Si, "A Data-Driven Design for Fault Detection of Wind Turbines Using Random Forests and XGboost," in IEEE Access, vol. 6, pp. 21020-21031, 2018, Available: https://ieeexplore.ieee.org/document/8329419