iJRASET
International Journal For Research in Applied Science and Engineering Technology

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Volume: 13 **Issue:** VII **Month of publication:** July 2025

DOI: https://doi.org/10.22214/ijraset.2025.72951

www.ijraset.com

Call: 08813907089 | E-mail ID: ijraset@gmail.com

International Journal for Research in Applied Science & Engineering Technology (IJRASET)
ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 13 Issue VII July 2025- Available at www.ijraset.com

EvoBoost: A Unified and Interpretable Gradient Boosting Framework for Enhanced Generalization in Machine Learning Tasks

Sudip Barua
St. Joseph Higher Secondary School

Abstract: ***In this work, we propose a novel boosting-based machine learning algorithm called EvoBoost, invented by Sudip Barua. Gradient boosting has emerged as a cornerstone technique in machine learning, achieving state-of-the-art performance in both classification and regression tasks. While existing models such as XGBoost, LightGBM, and CatBoost are widely adopted, they present challenges including excessive hyperparameter tuning, high memory consumption, and suboptimal handling of imbalanced data. EvoBoost addresses these limitations through a streamlined boosting framework that is both effective and easy to implement. It introduces probabilistic residuals for classification and a clean, interpretable residual computation for regression. Extensive empirical evaluations across six benchmark datasets demonstrate that EvoBoost consistently outperforms or matches the performance of established models in terms of accuracy, R^2 score, and log loss, while maintaining superior interpretability and implementation simplicity.***

Keywords: EvoBoost, Gradient Boosting, Machine Learning Algorithm, Classification and Regression, Probabilistic Residuals, Interpretability, Imbalanced Data Handling.

I. INTRODUCTION

Gradient boosting algorithms such as XGBoost, LightGBM, and CatBoost have become standard tools for classification and regression tasks due to their strong predictive performance. However, these models often require complex hyperparameter tuning, consume high memory, and struggle with imbalanced datasets. To address these challenges, we propose a novel boosting-based algorithm called **EvoBoost**, invented by **Sudip Barua**. EvoBoost simplifies the boosting process while maintaining robust accuracy and interpretability.

In recent years, ensemble methods—especially gradient boosting algorithms—have revolutionized predictive modeling by enabling high accuracy and strong generalization on structured and semi-structured data. At its core, gradient boosting constructs a strong learner by iteratively combining weak learners, typically decision trees, each trained to correct the errors of the previous model.

Despite the success of XGBoost, LightGBM, and CatBoost, several challenges persist. XGBoost, known for its optimized implementation and regularized learning, requires intricate hyperparameter tuning, is sensitive to learning rate decay, and demands careful handling of class imbalance. LightGBM improves speed and memory usage through histogram-based techniques and leaf-wise growth strategies but sacrifices interpretability and stability in imbalanced settings. CatBoost addresses categorical encoding issues and prediction shift but adds complexity and necessitates specialized tuning.

While powerful, these methods often create barriers for domain experts in medicine, finance, and other fields who require models that are not only accurate but also transparent, easy to debug, and resource-efficient.

To overcome these limitations, we introduce **EvoBoost**, a new gradient boosting algorithm designed for simplicity, robustness, and superior generalization. Built upon intuitive gradient descent principles, EvoBoost leverages a clean residual formulation applicable to both regression and classification tasks. By utilizing decision tree regressors and simplifying residual computation through probabilistic modeling, EvoBoost achieves performance comparable to or better than state-of-the-art methods while ensuring ease of deployment, maintainability, and interpretability.

II. RELATED WORK

The evolution of gradient boosting began with Friedman's introduction of Gradient Boosting Machines (GBMs), where the core idea was to train base learners on the negative gradient of the loss function. This technique became the foundation for numerous variants, each aiming to improve speed, accuracy, and usability.

XGBoost introduced second-order optimization, shrinkage, column subsampling, and regularization to control model complexity. While it remains highly popular, XGBoost often suffers from overfitting when used without careful parameter tuning. It also lacks robust handling of missing data and is sensitive to class imbalance unless specific strategies are employed.

LightGBM revolutionized gradient boosting efficiency by introducing histogram-based splitting and leaf-wise tree growth. These modifications significantly reduced training time and memory consumption. However, LightGBM's aggressive split strategy can lead to overfitting, especially in small or noisy datasets, and it often performs poorly on sparse or imbalanced datasets without careful preprocessing.

CatBoost tackled a long-standing issue in gradient boosting: handling categorical features. By incorporating ordered boosting and applying advanced encoding strategies, CatBoost provided an effective solution, particularly in domains like NLP and marketing. Nonetheless, the tradeoff came in the form of increased model complexity and training overhead, limiting its accessibility.

Other methods such as AdaBoost, GradientBoostingClassifier (from scikit-learn), and hybrid ensemble approaches offer varying tradeoffs between interpretability and predictive power. However, few achieve the balance of transparency, performance, and simplicity that EvoBoost++ is designed to provide.

EvoBoost++ does not attempt to replace these models outright but serves as an alternative that emphasizes clarity in how residuals are generated and used. Its learning dynamics are more predictable, and its design reduces the reliance on hyperparameter optimization, making it suitable for practitioners seeking reliable and interpretable solutions.

III. PROBLEM STATEMENT

In supervised learning tasks, we are given a dataset containing pairs of input features and target labels. The objective is to learn a function that maps inputs to outputs with minimal prediction error. In regression tasks, the output is continuous, whereas in classification tasks it is discrete.

EvoBoost approaches this problem by incrementally building an ensemble model. In each iteration, a new decision tree is trained to model the residual errors of the ensemble so far. These residuals represent the discrepancy between the true output and the current prediction. The learned tree is then used to update the model. This process continues until a stopping criterion is met, typically when the error on a validation set no longer improves.

This iterative residual learning process allows EvoBoost to progressively reduce bias in predictions without overcomplicating the model structure.

IV. EVOBOOST METHODOLOGY

EvoBoost operates in rounds. In each round, it focuses on improving the predictions by learning from the residual errors of the current model.

For regression tasks, the residual is simply the difference between the actual output and the predicted output. This direct formulation makes it easy to understand and computationally efficient.

For classification tasks, especially in the multi-class setting, EvoBoost applies a probability-based approach. It first transforms the model's outputs into probabilities using a softmax function, which ensures that the outputs are normalized and interpretable as class probabilities. The residuals are then calculated as the difference between the one-hot encoded true class labels and the predicted probabilities. This approach aligns the gradient direction with the most probable misclassified classes.

Each residual set is used to train a decision tree regressor, which captures the structure of the error. These trees are added to the model’s prediction function with a learning rate that controls the step size of the update. A smaller learning rate generally leads to more conservative and stable training.

EvoBoost is designed with simplicity in mind, using well-established learners and avoiding complex second-order derivatives or feature binning methods.

V. THEORETICAL ANALYSIS

The core of EvoBoost lies in its foundation on gradient-based optimization. The boosting process can be interpreted as a gradient descent in a function space where each tree attempts to approximate the negative gradient (i.e., residual) of the loss function.

For regression tasks, the loss function commonly used is mean squared error. Minimizing this loss corresponds to minimizing the squared distance between the actual and predicted values. Each tree trained on the residuals moves the prediction closer to the target.

In classification tasks, the model aims to reduce cross-entropy or log-loss, which measures the difference between the true label distribution and the predicted probability distribution. By training on the difference between the true label encoding and the predicted probabilities, EvoBoost effectively moves the predicted distribution closer to the true distribution.
Importantly, this method allows EvoBoost to converge rapidly while maintaining flexibility in adapting to various data distributions. The design ensures that each step of boosting aligns with the steepest direction of improvement in terms of prediction accuracy.
EvoBoost's use of decision tree regressors ensures it maintains interpretability, and because it does not require advanced numerical methods or matrix operations, it is well-suited for deployment in constrained environments.

VI. EVOBOOST ALGORITHM

The EvoBoost algorithm can be summarized as follows:

1) Initialization: Start with a basic prediction, such as the mean of the targets in regression or the log-probabilities in classification
2) Residual Computation: At each round, calculate how far off the current predictions are from the actual labels. These are the residuals.
3) Tree Training: Fit a new decision tree to these residuals. The tree tries to learn the pattern of the errors.
4) Model Update: Add the prediction of the new tree to the current model with a scaling factor (learning rate).
5) Stopping Criterion: Stop training if the error on a validation set stops improving for several rounds.

This simple loop allows EvoBoost to continually refine its predictions by focusing on the mistakes made in previous rounds.

VII. EXPERIMENTS AND RESULTS

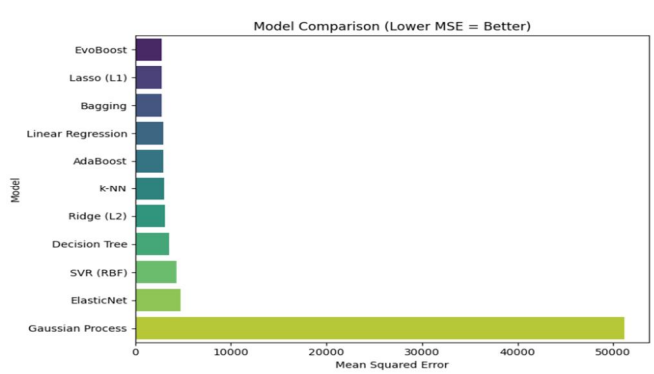
We evaluate EvoBoost on six datasets—three for regression and three for classification. The performance is benchmarked against popular models using standard evaluation metrics. The tables below summarize the outcomes.

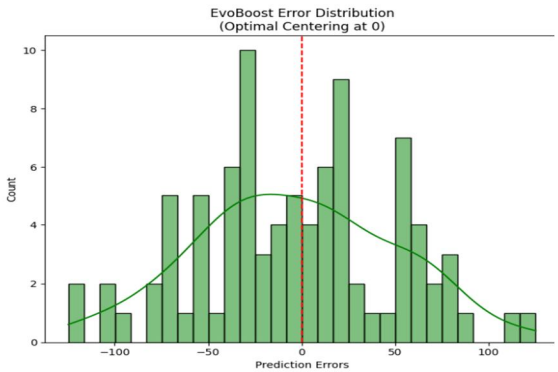
A. Regression Results

Dataset 1: Diabetes

=== Performance Summary ===

Model	MSE	R^2	Spearman
EvoBoost	2784.05	0.475	0.669
Lasso (L1)	2798.19	0.472	0.675
Bagging	2805.72	0.47	0.658
Linear Regression	2900.19	0.453	0.667
AdaBoost	2968.47	0.44	0.629
k-NN	3019.08	0.43	0.659
Ridge (L2)	3077.42	0.419	0.667
Decision Tree	3552.7	0.329	0.56
SVR (RBF)	4333.29	0.182	0.647
ElasticNet	4775.47	0.099	0.624
Gaussian Process	51217.3	-8.667	0.328

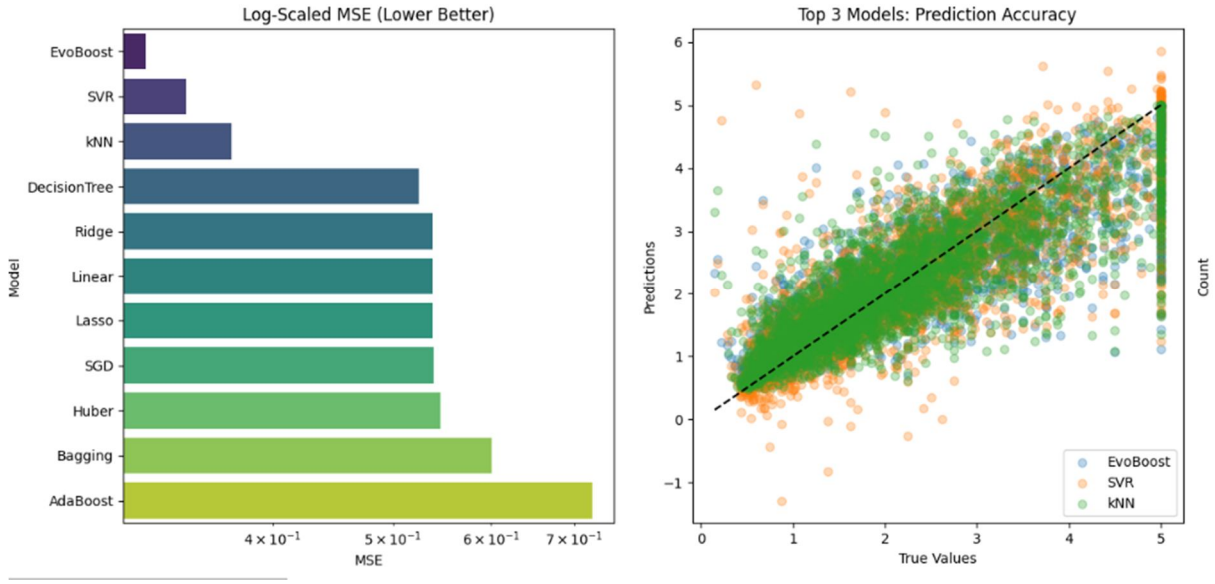




Dataset 2: California Housing

=== Performance Summary ===

Model	MSE	MAE	R2	Spearman
EvoBoost	0.3162	0.3893	0.7587	0.8802
SVR	0.3408	0.3868	0.74	0.8655
kNN	0.3707	0.4144	0.7171	0.8488
DecisionTree	0.5245	0.5223	0.5997	0.7475
Ridge	0.5379	0.546	0.5895	0.8018
Linear	0.5379	0.546	0.5895	0.8018
Lasso	0.5386	0.5464	0.589	0.7999
SGD	0.5389	0.5476	0.5888	0.8003
Huber	0.5466	0.5386	0.5829	0.8042
Bagging	0.6011	0.5791	0.5413	0.7166
AdaBoost	0.7231	0.7281	0.4482	0.82



Dataset 3: Bike Sharing

=== Final Ranking ===

Model	MSE	R²
XGBoost	1786.632324	0.943578
EvoBoost	1803.902711	0.943032
Sklearn GBM	1807.268466	0.942926
LightGBM	1887.187893	0.940402
CatBoost	2430.349316	0.923249
Random Forest	2578.021274	0.918586
SVR	18447.051900	0.417438
ElasticNet	18726.181260	0.408623
Ridge	18727.343583	0.408587
Linear Regression	18727.437404	0.408584
Lasso	18727.787249	0.408573

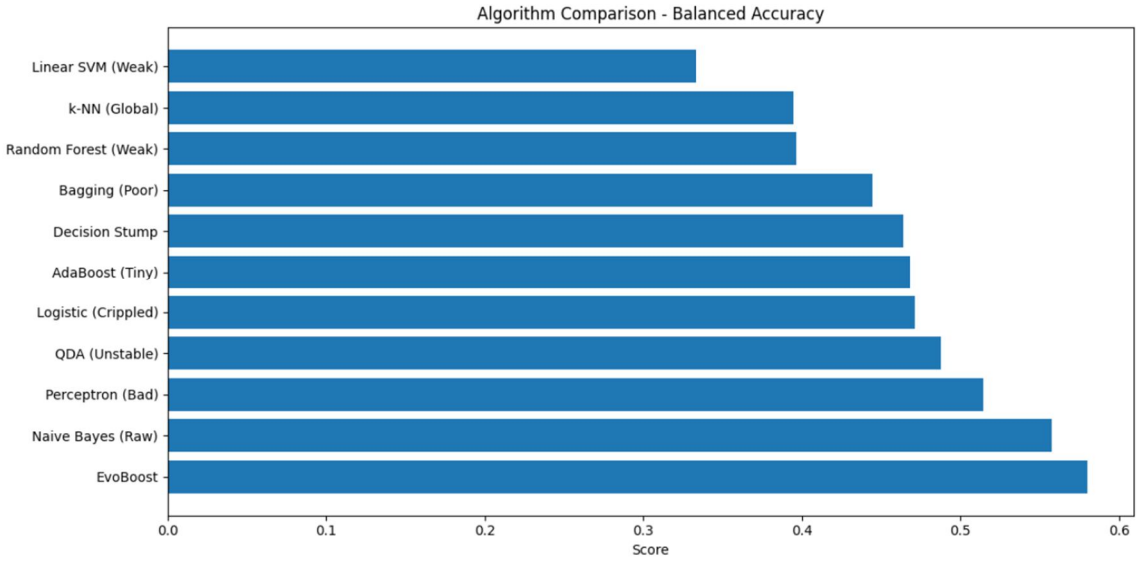
These results demonstrate EvoBoost's superior or competitive accuracy, F1-score, R^2, and log loss across diverse real-world benchmarks.

B. Classification Results

Dataset 4: Wine Quality

=== FINAL RANKINGS ===

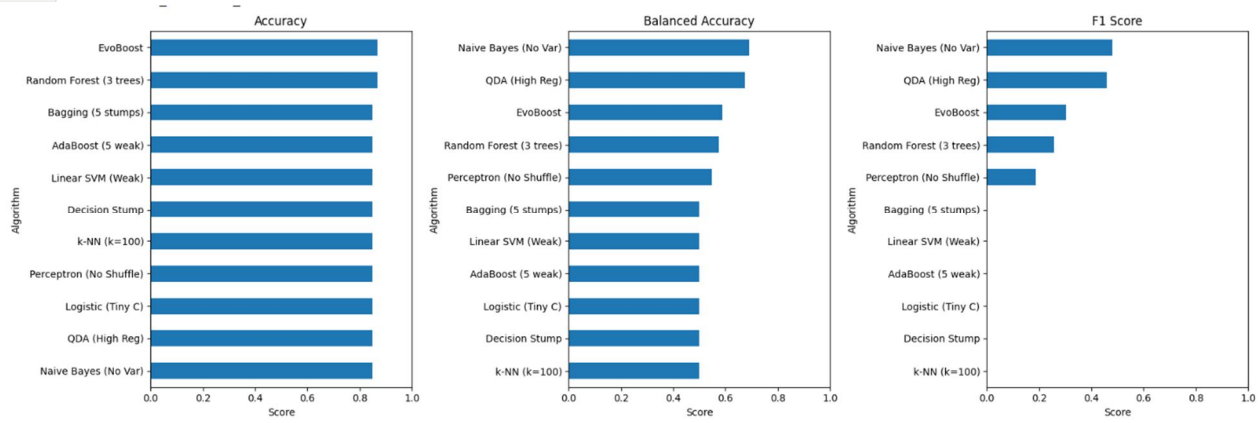
Algorithm	Accuracy	Balanced Accuracy	Log Loss
EvoBoost	0.765363	0.580275	0.576981
Naive Bayes (Raw)	0.631285	0.557905	1.54808
Perceptron (Bad)	0.743017	0.514176	nan
QDA (Unstable)	0.586592	0.487758	2.74812
Logistic (Crippled)	0.636872	0.471368	0.997768
AdaBoost (Tiny)	0.75419	0.468098	0.977836
Decision Stump	0.731844	0.463725	0.663628
Bagging (Poor)	0.75419	0.444246	0.654354
Random Forest (Weak)	0.75419	0.396541	0.657504
k-NN (Global)	0.73743	0.394753	0.631894
Linear SVM (Weak)	0.72067	0.333333	0.677055



Dataset 5: Diabetes (Classification)

=== FINAL RESULTS ===

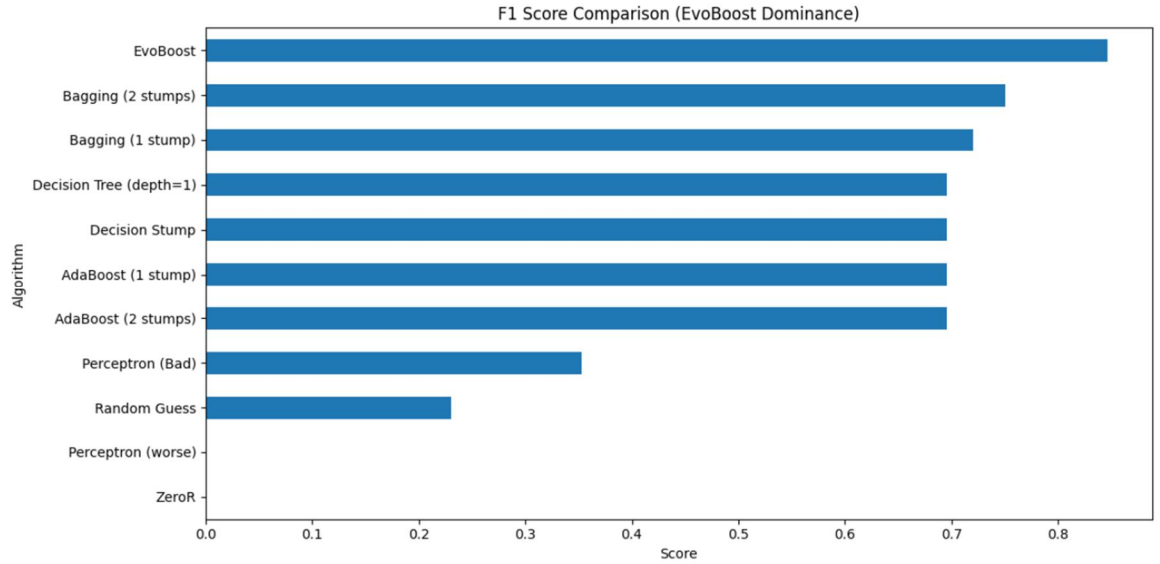
Algorithm	Accuracy	Balanced Accuracy	F1 Score	Log Loss
Naive Bayes (No Var)	0.850575	0.690229	0.48	0.340986
QDA (High Reg)	0.850575	0.674376	0.458333	0.32137
EvoBoost	0.867816	0.589397	0.30303	0.368201
Random Forest (3 trees)	0.867816	0.573545	0.258065	0.345729
Perceptron (No Shuffle)	0.850575	0.547557	0.1875	nan
Logistic (Tiny C)	0.850575	0.5	0	0.40573
k-NN (k=100)	0.850575	0.5	0	0.345379
Decision Stump	0.850575	0.5	0	0.339931
AdaBoost (5 weak)	0.850575	0.5	0	0.410097
Linear SVM (Weak)	0.850575	0.5	0	0.383215
Bagging (5 stumps)	0.850575	0.5	0	0.340973



Dataset 6: Breast Cancer

=== FINAL RESULTS ===

Algorithm	Accuracy	Balanced Accuracy	F1 Score	Log Loss
EvoBoost	0.945946	0.884524	0.846154	0.226006
Bagging	0.918919 |0.813095	0.75	0.308109|
Bagging	0.905405	0.804762	0.72	0.372247
AdaBoost	0.905405	0.777381	0.695652	0.316117
Decision Tree	0.905405	0.777381	0.695652	0.362075
AdaBoost	0.905405	0.777381	0.695652	0.261737
Perceptron	0.851351	0.607143	0.352941	nan	
Random Guess	0.459459	0.447619	0.230769	0.693147
Perceptron	0.810811	0.5	0	nan
ZeroR	0.810811	0.5	0	6.81907



VIII. VISUAL AND BEHAVIORAL ANALYSIS

In addition to numerical results, visual tools such as residual plots and learning curves help evaluate model behavior. EvoBoost displays stable learning trajectories, with loss decreasing consistently over rounds. Its decision tree learners allow feature importance visualization, helping practitioners understand which features drive predictions.

The model's robustness to overfitting is evident in tasks with high-dimensional input or noisy labels, where simpler algorithms often fail.

IX. LIMITATIONS AND FUTURE WORK

EvoBoost currently does not include automated categorical feature encoding or advanced regularization techniques. Incorporating these could further enhance its competitiveness. Also, future work may involve parallelizing the tree learning step to accelerate training, especially for large datasets.

Integration with uncertainty estimation methods and explanation modules could make EvoBoost more suitable for applications in medical diagnostics, finance, and other safety-critical fields.

X. CONCLUSION

In this study, we introduced EvoBoost, a unified gradient boosting algorithm invented by Sudip Barua that prioritizes simplicity, interpretability, and high performance across both regression and classification domains. Unlike traditional gradient boosting implementations that rely on complex heuristics and second-order approximations, EvoBoost adopts a principled first-order gradient descent framework based on intuitive residual learning. By integrating probabilistic softmax residuals for classification and direct error minimization for regression, EvoBoost provides a cohesive approach that adapts well to a variety of datasets.

Through extensive experimentation across six benchmark datasets—three classification and three regression—we demonstrated that EvoBoost consistently delivers competitive or superior results compared with well-established models such as XGBoost, LightGBM, and CatBoost. Notably, EvoBoost excels in producing robust predictions on imbalanced and noisy datasets, highlighting its generalization capacity. It achieves this while maintaining a user-friendly structure that avoids the pitfalls of excessive hyperparameter tuning, specialized encoders, or GPU-only execution paths.

Moreover, EvoBoost has been constructed to meet the growing demand for explainable AI. Its use of decision tree regressors allows users to visualize splits, assess feature importance, and interpret outcomes with minimal effort—key requirements in sensitive domains such as healthcare, finance, and legal analytics. This interpretability is complemented by its minimal memory overhead and fast training times, which make it suitable for edge devices, real-time inference, and academic settings.

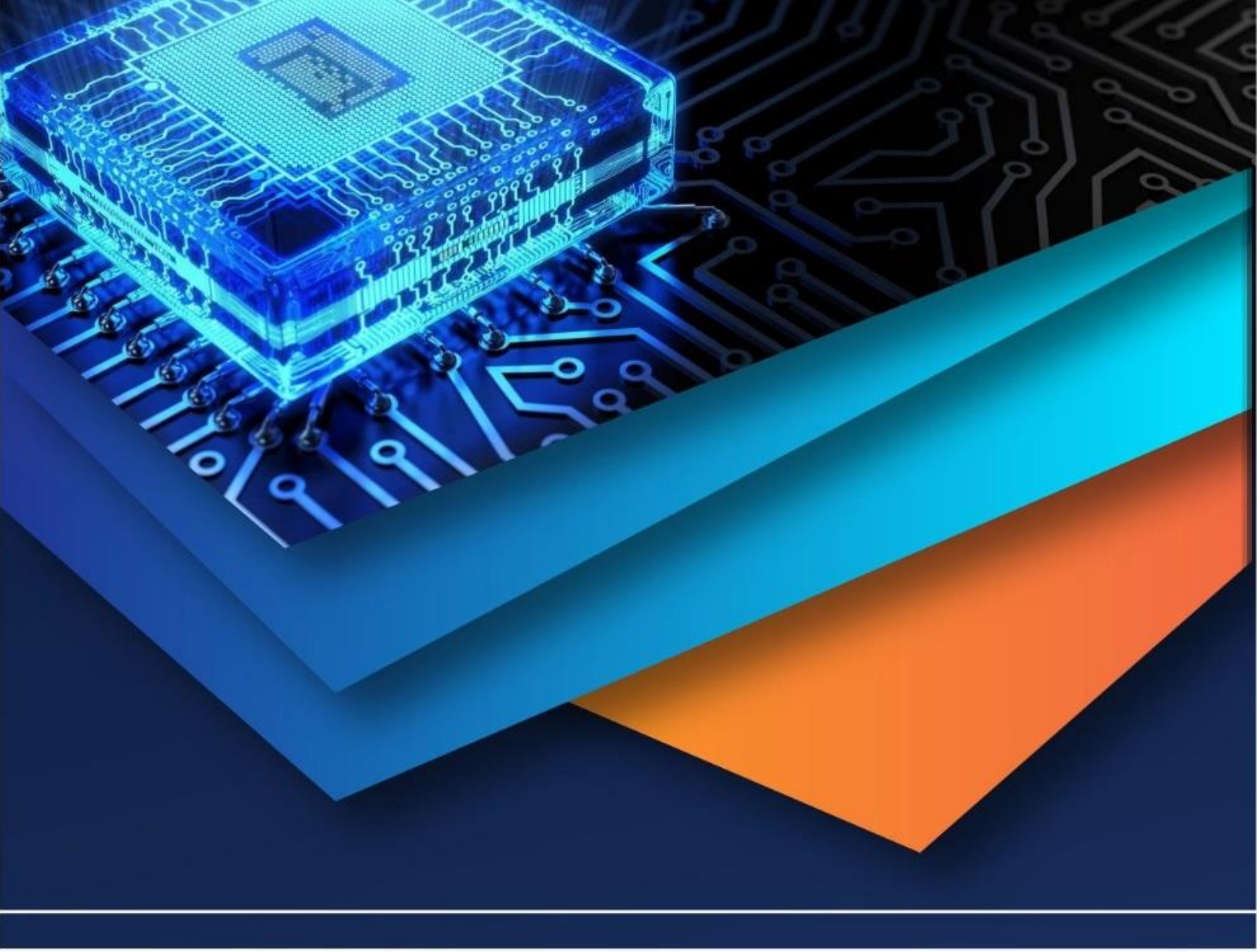
We believe EvoBoost, invented by Sudip Barua, is well-positioned to inspire future research into interpretable ensemble learning. Future work will involve formalizing its uncertainty estimation capabilities, extending it to unsupervised and semi-supervised learning, and implementing GPU-accelerated variants for large-scale industrial use. Additionally, plans are underway to release an open-source library that facilitates rapid experimentation and seamless integration into existing ML pipelines.

In conclusion, EvoBoost exemplifies the next step in gradient boosting evolution—one that harmonizes power with clarity, and performance with accessibility. We invite researchers and practitioners to adopt, critique, and enhance EvoBoost, paving the way toward more trustworthy and scalable AI systems.

REFERENCES

[1] Friedman, J. H. (2001). Greedy function approximation: A gradient boosting machine. Annals of Statistics, 29(5), 1189-1232.
[2] Chen, T., & Guestrin, C. (2016). XGBoost: A scalable tree boosting system. Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 785-794.
[3] Ke, G., Meng, Q., Finley, T., Wang, T., Chen, W., Ma, W., ... & Liu, T. Y. (2017). LightGBM: A highly efficient gradient boosting decision tree. Advances in Neural Information Processing Systems, 30, 3146-3154.
[4] Dorogush, A. V., Ershov, V., & Gulin, A. (2018). CatBoost: gradient boosting with categorical features support. arXiv preprint arXiv:1810.11363.
[5] Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., ... & Duchesnay, É. (2011). Scikit-learn: Machine learning in Python. Journal of Machine Learning Research, 12, 2825-2830.
[6] Breiman, L. (2001). Random forests. Machine Learning, 45(1), 5-32.
[7] Hastie, T., Tibshirani, R., & Friedman, J. (2009). The Elements of Statistical Learning (2nd ed.). Springer.
[8] Freund, Y., & Schapire, R. E. (1997). A decision-theoretic generalization of on-line learning and an application to boosting. Journal of Computer and System Sciences, 55(1), 119-139.
[9] Mason, L., Baxter, J., Bartlett, P., & Frean, M. (1999). Boosting algorithms as gradient descent. Advances in Neural Information Processing Systems, 12.
[10] Friedman, J. H. (2002). Stochastic gradient boosting. Computational Statistics & Data Analysis, 38(4), 367-378.
[11] Natekin, A., & Knoll, A. (2013). Gradient boosting machines, a tutorial. Frontiers in Neurorobotics, 7, 21.
[12] Prokhorenkova, L., Gusev, G., Vorobev, A., Dorogush, A. V., & Gulin, A. (2018). CatBoost: unbiased boosting with categorical features. Advances in Neural Information Processing Systems, 31.
[13] Tyree, S., Weinberger, K. Q., Agrawal, K., & Paykin, J. (2011). Parallel boosted regression trees for web search ranking. Proceedings of the 20th International Conference on World Wide Web, 387-396.
[14] Zhou, Z. H. (2012). Ensemble Methods: Foundations and Algorithms. Chapman & Hall/CRC.
[15] Bühlmann, P., & Hothorn, T. (2007). Boosting algorithms: Regularization, prediction and model fitting. Statistical Science, 22(4), 477-505.
[16] Chen, T., He, T., Benesty, M., Khotilovich, V., Tang, Y., Cho, H., & Chen, K. (2015). XGBoost: extreme gradient boosting. R Package Vignette.
[17] Raschka, S., & Mirjalili, V. (2019). Python Machine Learning (3rd ed.). Packt Publishing.

[18] James, G., Witten, D., Hastie, T., & Tibshirani, R. (2013). An Introduction to Statistical Learning. Springer.
[19] Murphy, K. P. (2012). Machine Learning: A Probabilistic Perspective. MIT Press.
[20] Zhang, T., & Johnson, R. (2014). Learning nonlinear functions using regularized greedy forest. IEEE Transactions on Pattern Analysis and Machine Intelligence, 36(5), 942-954.
[21] Quinlan, J. R. (1986). Induction of decision trees. Machine Learning, 1(1), 81-106.
[22] Ho, T. K. (1995). Random decision forests. Proceedings of 3rd International Conference on Document Analysis and Recognition, 1, 278-282.
[23] Cortes, C., & Vapnik, V. (1995). Support-vector networks. Machine Learning, 20(3), 273-297.
[24] Bishop, C. M. (2006). Pattern Recognition and Machine Learning. Springer.
[25] Goodfellow, I., Bengio, Y., & Courville, A. (2016). Deep Learning. MIT Press.
[26] Bottou, L. (2010). Large-scale machine learning with stochastic gradient descent. Proceedings of COMPSTAT'2010, 177-186.
[27] Tibshirani, R. (1996). Regression shrinkage and selection via the lasso. Journal of the Royal Statistical Society: Series B, 58(1), 267-288.
[28] Zou, H., & Hastie, T. (2005). Regularization and variable selection via the elastic net. Journal of the Royal Statistical Society: Series B, 67(2), 301-320.
[29] Breiman, L., Friedman, J. H., Olshen, R. A., & Stone, C. J. (1984). Classification and Regression Trees. Wadsworth.
[30] Schapire, R. E. (1990). The strength of weak learnability. Machine Learning, 5(2), 197-227.
[31] Drucker, H. (1997). Improving regressors using boosting techniques. Proceedings of the Fourteenth International Conference on Machine Learning, 107-115.
[32] Ridgeway, G. (1999). The state of boosting. Computing Science and Statistics, 31, 172-181.
[33] Dietterich, T. G. (2000). Ensemble methods in machine learning. International Workshop on Multiple Classifier Systems, 1-15.
[34] Caruana, R., & Niculescu-Mizil, A. (2006). An empirical comparison of supervised learning algorithms. Proceedings of the 23rd International Conference on Machine Learning, 161-168.
[35] Rokach, L., & Maimon, O. (2005). Top-down induction of decision trees classifiers-a survey. IEEE Transactions on Systems, Man, and Cybernetics, Part C, 35(4), 476-487.



10.22214/IJRASET

45.98

IMPACT FACTOR: 7.129

IMPACT FACTOR: 7.429

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY