A Simplified Replication Study: Comparing Supervised Learning Algorithms Using GridSearchCV

Ethan Chuang
UC San Diego
9500 Gilman Dr
La Jolla, CA 92093, USA
ethan40125@gmail.com

***Abstract*—This study replicates key aspects of Caruana and Niculescu-Mizil's (2006) empirical comparison of supervised learning algorithms. We evaluate four algorithms (k-nearest neighbors, random forests, gradient boosting, and multilayer perceptron) on four binary classification problems using three performance metrics: accuracy, F1 score, and area under the ROC curve. Our results indicate that gradient boosting consistently outperforms other algorithms across different metrics and datasets, aligning with findings from the original study. We also examine the impact of feature scaling on model performance, finding significant effects particularly for KNN and MLP. This work provides insights into the relative strengths of these algorithms and the importance of preprocessing in machine learning workflows.**

***Index Terms*—supervised learning, classification, machine learning algorithms, performance evaluation, empirical study**

I. Introduction

Machine learning has seen rapid advancements in recent years, with a proliferation of algorithms for supervised learning tasks. As new methods emerge, it becomes increasingly important to conduct empirical comparisons to understand their relative strengths and weaknesses.

Caruana and Niculescu-Mizil's 2006 paper provided valuable insights into the performance of various methods across multiple datasets and metrics [1]. Our study aims to replicate aspects of their experiment on a smaller scale, focusing on four contemporary supervised learning algorithms: k-nearest neighbors (KNN), random forests (RF), gradient boosting (GB), and multilayer perceptron (MLP).

We evaluate these algorithms on four binary classification problems, including three datasets used in the original study and an additional HR Analytics dataset. Our analysis focuses on three key performance metrics: accuracy, F1 score, and area under the ROC curve (AUC).

A key aspect of our study is the use of GridSearchCV for hyperparameter optimization, which allows us to explore a broader range of algorithm configurations than the original study. Additionally, we examine the impact of feature scaling on model performance, an aspect not extensively covered in the original study.

Through this work, we seek to contribute to the ongoing evaluation of machine learning methods and provide practitioners with updated insights into the performance characteristics of these popular algorithms across different classification tasks.

II. Methodology

A. Learning Algorithms

This section summarizes the four supervised learning algorithms we evaluate.

K-Nearest Neighbors (KNN): We use 10 values of K ranging from 3 to 30, evenly spaced. We use KNN with Euclidean distance and consider both uniform and distance weights. We also explore different search algorithms including BallTree, KDTree, and brute-force search.

Random Forests (RF): We use the scikit-learn implementation. The forests have 125, 250, 500, or 1000 trees. The size of the feature set considered at each split is either 'sqrt' or 'log2' of the total number of features.

Gradient Boosting (GB): We explore a wide range of estimators with n_estimators values of [10, 50, 100, 500, 1000, 2000]. We consider the same max_features options as RF ('sqrt' and 'log2').

Multilayer Perceptron (MLP): We train neural nets with gradient descent backpropagation and vary the number of hidden units [32, 64, 128]. We use three different solvers: 'lbfgs', 'sgd', and 'adam'. The maximum number of iterations is set to either 500, 1000, or 2000. We use validation sets to select the best nets.

B. Performance Metrics

We evaluate model performance using three metrics:

- Accuracy: $\frac{TP+TN}{TP+TN+FP+FN}$
- F1 Score: $2 \cdot \frac{precision \cdot recall}{precision+recall}$
- Area Under the ROC Curve (AUC)

C. Datasets

We use four datasets in this study. LETTERP1 and LETTERP2 are derived from the Letter Recognition dataset, focusing on distinguishing the letter 'O' and classifying letters A-M against N-Z, respectively [2]. The Adult dataset predicts income levels based on census data [2]. The HR dataset is used for employee attrition prediction [3].

TABLE I
PROBLEM SET DESCRIPTIONS

PROBLEM	#ATTR	TRAIN SIZE	TEST SIZE	%POS
LETTERP1	16	5000	15000	4%
LETTERP2	16	5000	15000	50%
ADULT	14/49	5000	43842	24%
HR	12	5000	14158	25%

D. Preprocessing

For LETTERP1, LETTERP2, and Adult datasets, we prepare both scaled and non-scaled versions using StandardScaler. The HR Analytics dataset, being predominantly categorical, is not scaled. Categorical variables are encoded using label encoding or one-hot encoding as appropriate.

E. Experimental Setup

For each dataset, we randomly select 5000 samples for training. We perform 5-fold cross-validation to identify optimal hyperparameters for each algorithm, using accuracy as the selection criterion. Once the best hyperparameters are determined, we train the models on the entire 5000-sample training set and evaluate their performance on the remaining data for each problem.

We use GridSearchCV for hyperparameter optimization, exploring the parameter ranges described in the Learning Algorithms subsection. This approach allows us to assess both the algorithms' performance and the effectiveness of our hyperparameter optimization strategy across a range of classification tasks.

III. EXPERIMENTS AND RESULTS

We conducted experiments evaluating the four learning algorithms on each dataset, with and without scaling (except for the HR dataset). Here, we present the key findings and trends observed across these experiments. A comprehensive overview of the results for all algorithms and datasets is provided in Table IV in the Appendix.

A. Overall Performance

Tables II and III summarize the performance of each algorithm across metrics and problems, respectively.

Gradient Boosting (GB) demonstrated strong overall performance across all metrics, particularly in the Adult and HR datasets. However, the relative performance of algorithms varied across datasets. In LETTERP1, GB performed best, followed closely by KNN and MLP, while RF showed slightly lower performance. For LETTERP2, all algorithms performed well, with KNN and RF slightly outperforming GB and MLP.

TABLE II
AVERAGE SCORES OF EACH LEARNING ALGORITHM BY METRIC

MODEL	ACCURACY	F1	AUC	MEAN
GB	0.91	0.79	0.85	0.85
RF	0.91	0.75	0.83	0.83
MLP	0.90	0.76	0.84	0.83
KNN	0.91	0.76	0.84	0.84

TABLE III
AVERAGE SCORES OF EACH LEARNING ALGORITHM BY PROBLEM

MODEL	LetterP1	LetterP2	ADULT	HR	MEAN
KNN	0.93	0.95	0.76	0.57	0.80
RF	0.89	0.95	0.76	0.61	0.80
GB	0.94	0.94	0.79	0.63	0.83
MLP	0.92	0.93	0.76	0.61	0.81

In the Adult dataset, GB outperformed other algorithms, while the rest showed similar performance. The HR dataset proved challenging for all algorithms, with GB performing best, followed by RF and MLP, while KNN showed the lowest performance. The best-performing algorithm for each dataset is summarized in Table V in the Appendix.

B. Impact of Scaling

Scaling had a significant impact on the performance of certain algorithms, particularly KNN and MLP. For the LETTERP1 and LETTERP2 datasets, scaling led to noticeable improvements in performance, particularly for KNN and MLP. In the Adult dataset, scaling improved the performance of all algorithms, with MLP showing the most significant improvement. RF and GB were relatively unaffected by scaling, demonstrating their robustness to different data distributions. The impact of scaling on algorithm performance is detailed in Table VI in the Appendix.

C. Hyperparameter Sensitivity

Our experiments revealed varying degrees of sensitivity to hyperparameter tuning. GB and RF showed relatively consistent performance across different hyperparameter settings. In contrast, MLP and KNN were more sensitive to hyperparameter tuning, with performance varying significantly based on the chosen parameters. The full results of this hyperparameter optimization can be found in Table VII in the Appendix.

D. Computational Considerations

While GB and RF consistently performed well across datasets, they are more computationally expensive than simpler algorithms like KNN. This trade-off between performance and computational cost should be considered when selecting algorithms for large-scale or real-time applications.

Our results partially align with the findings of Caruana and Niculescu-Mizil. GB demonstrates strong performance across different metrics and datasets in both studies. However, KNN showed superior performance in certain datasets in our study, particularly in the letter recognition tasks, which was not as prominent in the original study. This discrepancy could be

attributed to differences in dataset characteristics, hyperparameter tuning, or scaling effects, which had a significant impact on performance, particularly for KNN and MLP.

IV. Conclusion

This study presents a simplified replication of Caruana and Niculescu-Mizil's empirical comparison of supervised learning algorithms, focusing on four popular algorithms across four binary classification problems. Our findings offer several insights into the performance of these algorithms and their applicability to different types of classification tasks.

Gradient Boosting emerged as a top-performing algorithm across the datasets and metrics examined, demonstrating the highest average performance, particularly in the HR and Adult datasets. However, K-Nearest Neighbors showed superior performance in the Letter datasets, outperforming other algorithms in both accuracy and F1 score. This result highlights the importance of dataset-specific characteristics and the critical role of feature scaling. Random Forests and Multilayer Perceptron also performed well, with their relative performance varying across datasets and metrics.

The impact of scaling on model performance, particularly for KNN and MLP, highlights the crucial role of preprocessing in machine learning workflows. Our study revealed that no single algorithm consistently outperforms all others across all datasets, underscoring the importance of algorithm selection based on the specific characteristics of the problem at hand.

These findings align with some aspects of Caruana and Niculescu-Mizil's study, particularly in the strong performance of Gradient Boosting. However, they also emphasize the variability in algorithm performance across different datasets and the critical role of hyperparameter tuning and preprocessing. The superior performance of KNN in our letter recognition tasks, which was not as prominent in the original study, underscores the importance of thorough hyperparameter optimization and the potential for simpler algorithms to excel in certain problem domains.

Our study also revealed the complex interplay between algorithms, datasets, and performance metrics. While Gradient Boosting showed strong overall performance, the best-performing algorithm varied depending on the specific dataset and metric used. The computational considerations discussed provide an additional practical perspective for choosing between high-performing but computationally expensive models (RF, GB) and simpler models (KNN, MLP).

While our study confirms the strong performance of Gradient Boosting across various classification tasks, it also highlights the importance of careful algorithm selection, preprocessing, and hyperparameter tuning based on the specific characteristics of the problem at hand. As the field of machine learning continues to evolve, ongoing empirical evaluations of this nature will remain crucial for guiding both research directions and practical applications.

Future research could expand on this work by including more algorithms, broader hyperparameter ranges, and a wider variety of datasets. Investigating different preprocessing techniques and feature engineering methods would also be valuable, potentially uncovering more ways to optimize model performance across various scenarios.

References

[1] R. Caruana and A. Niculescu-Mizil, "An empirical comparison of supervised learning algorithms," in *Proceedings of the 23rd international conference on Machine learning*, 2006, pp. 161-168.

[2] D. Dua and C. Graff, "UCI Machine Learning Repository," University of California, Irvine, School of Information and Computer Sciences, 2019. [Online]. Available: http://archive.ics.uci.edu/ml

[3] N. K. Gupta, "Who Will Leave a Job," Kaggle, 2020. [Online]. Available: https://www.kaggle.com/code/nkitgupta/who-will-leave-a-job/

APPENDIX

TABLE IV
COMPREHENSIVE RESULTS FOR ALL ALGORITHMS AND DATASETS

Dataset	Method	Scaling	Accuracy	F1	AUC	Average Score
adult	KNeighboursClassifier	No Scale	0.844647	0.633601	0.748236	0.742161
adult	RandomForestClassifier	No Scale	0.844920	0.652385	0.764586	0.753964
adult	GradientBoostingClassifier	No Scale	0.866475	0.699795	0.793321	0.786530
adult	MLPClassifier	No Scale	0.849733	0.639409	0.749993	0.746378
adult	MLPClassifier	Scale	0.848570	0.672585	0.781439	0.767531
adult	GradientBoostingClassifier	Scale	0.866954	0.700795	0.793898	0.787216
adult	RandomForestClassifier	Scale	0.846015	0.655684	0.766885	0.756195
adult	KNeighboursClassifier	Scale	0.858857	0.689482	0.789869	0.779403
hr	KNeighboursClassifier	No Scale	0.760842	0.344307	0.590880	0.565343
hr	GradientBoostingClassifier	No Scale	0.780266	0.473515	0.652041	0.635274
hr	MLPClassifier	No Scale	0.763243	0.383370	0.606946	0.584520
hr	RandomForestClassifier	No Scale	0.770448	0.435372	0.631696	0.612505
letterP1	KNeighboursClassifier	No Scale	0.990267	0.869643	0.932294	0.930734
letterP1	RandomForestClassifier	No Scale	0.988333	0.821611	0.859198	0.889714
letterP1	MLPClassifier	Scale	0.991267	0.880365	0.928522	0.933385
letterP1	KNeighboursClassifier	Scale	0.990200	0.869565	0.934834	0.931533
letterP1	RandomForestClassifier	Scale	0.988200	0.820669	0.860845	0.889905
letterP1	GradientBoostingClassifier	Scale	0.991400	0.879776	0.920009	0.930395
letterP1	MLPClassifier	No Scale	0.989333	0.860870	0.938675	0.929626
letterP1	GradientBoostingClassifier	No Scale	0.992133	0.890130	0.925539	0.935934
letterP2	KNeighboursClassifier	Scale	0.950667	0.950242	0.950666	0.950525
letterP2	MLPClassifier	No Scale	0.923533	0.922839	0.923527	0.923300
letterP2	GradientBoostingClassifier	No Scale	0.942400	0.942052	0.942422	0.942291
letterP2	RandomForestClassifier	No Scale	0.952000	0.951678	0.952016	0.951898
letterP2	KNeighboursClassifier	No Scale	0.955267	0.954982	0.955286	0.955178
letterP2	RandomForestClassifier	Scale	0.951533	0.951172	0.951543	0.951416
letterP2	GradientBoostingClassifier	Scale	0.941533	0.941137	0.941548	0.941406
letterP2	MLPClassifier	Scale	0.952467	0.951899	0.952437	0.952268

TABLE V
BEST PERFORMING ALGORITHM FOR EACH DATASET (BASED ON AVERAGE SCORE)

Dataset	Best Algorithm	Scaling	Average Score
adult	GradientBoostingClassifier	Scale	0.787216
hr	GradientBoostingClassifier	No Scale	0.635274
letterP1	GradientBoostingClassifier	No Scale	0.935934
letterP2	KNeighboursClassifier	No Scale	0.955178

TABLE VI
IMPACT OF SCALING ON ALGORITHM PERFORMANCE (AVERAGE SCORE)

Dataset	Algorithm	No Scale	Scale
adult	KNeighboursClassifier	0.742161	0.779403
adult	RandomForestClassifier	0.753964	0.756195
adult	GradientBoostingClassifier	0.786530	0.787216
adult	MLPClassifier	0.746378	0.767531
letterP1	KNeighboursClassifier	0.930734	0.931533
letterP1	RandomForestClassifier	0.889714	0.889905
letterP1	GradientBoostingClassifier	0.935934	0.930395
letterP1	MLPClassifier	0.929626	0.933385
letterP2	KNeighboursClassifier	0.955178	0.950525
letterP2	RandomForestClassifier	0.951898	0.951416
letterP2	GradientBoostingClassifier	0.942291	0.941406
letterP2	MLPClassifier	0.923300	0.952268

TABLE VII
BEST HYPERPARAMETERS FROM GRID SEARCH

Dataset	Algorithm	Scaling	Best Parameters
LETTERP1	MLPClassifier	No	hidden_layers_sizes: [128], max_iter: 500, solver: adam
		Yes	hidden_layers_sizes: [128], max_iter: 500, solver: adam
	GradientBoostingClassifier	No	max_features: sqrt, n_estimators: 2000
		Yes	max_features: sqrt, n_estimators: 1000
	RandomForestClassifier	No	max_features: sqrt, n_estimators: 250
		Yes	max_features: sqrt, n_estimators: 125
	KNeighborsClassifier	No	algorithm: kd_tree, n_neighbors: 6, weights: distance
		Yes	algorithm: ball_tree, n_neighbors: 6, weights: distance
LETTERP2	MLPClassifier	No	hidden_layers_sizes: [128], max_iter: 1000, solver: lbfgs
		Yes	hidden_layers_sizes: [128], max_iter: 1000, solver: adam
	GradientBoostingClassifier	No	max_features: log2, n_estimators: 2000
		Yes	max_features: log2, n_estimators: 2000
	RandomForestClassifier	No	max_features: log2, n_estimators: 500
		Yes	max_features: log2, n_estimators: 500
	KNeighborsClassifier	No	algorithm: kd_tree, n_neighbors: 6, weights: distance
		Yes	algorithm: ball_tree, n_neighbors: 3, weights: distance
ADULT	MLPClassifier	No	hidden_layers_sizes: [32], max_iter: 1000, solver: adam
		Yes	hidden_layers_sizes: [32], max_iter: 1000, solver: adam
	GradientBoostingClassifier	No	max_features: log2, n_estimators: 500
		Yes	max_features: log2, n_estimators: 500
	RandomForestClassifier	No	max_features: log2, n_estimators: 250
		Yes	max_features: sqrt, n_estimators: 1000
	KNeighborsClassifier	No	algorithm: brute, n_neighbors: 24, weights: distance
		Yes	algorithm: brute, n_neighbors: 24, weights: distance
HR	MLPClassifier	No	hidden_layers_sizes: [128], max_iter: 1000, solver: lbfgs
	GradientBoostingClassifier	No	max_features: log2, n_estimators: 50
	RandomForestClassifier	No	max_features: sqrt, n_estimators: 1000
	KNeighborsClassifier	No	algorithm: kd_tree, n_neighbors: 30, weights: distance