A Comprehensive Study of Supervised Learning Algorithms

Chaska Kentish
University of California, San Diego
COGS118A: Supervised Machine Learning Algorithms

Abstract

As machine learning continues to shape data-driven decision-making, the demand and need for understanding and harnessing diverse classification algorithms has grown exponentially. This paper seeks to help address the gap in comprehensive empirical studies by presenting an in-depth study and comparison of three supervised learning algorithms: Random Forest, Naive Bayes, K-Nearest Neighbors. The evaluation focuses on average performance accuracy across diverse problems and datasets and trials.

1 Introduction

In the ever-evolving landscape of machine learning, the role of supervised learning algorithms has become paramount in deciphering complex patterns within data. As the world continues to evolve into greater data-driven decision-making, the need to understand and harness the power of diverse classification algorithms has become more critical than ever. The surge in available algorithms offers not just a multitude of options but an opportunity to unravel the intricacies of predictive modeling in unique ways. However, there still remains a lack of comprehensive empirical studies that examine the relationship between these algorithms and their applicability across real-world data [1].

This paper addresses this issue by presenting the results of an empirical comparison of three different supervised learning algorithms across three distinct datasets, training criteria, and trials. I evaluate the performance of random forest modeling, Naive Bayes, and K-nearest neighbors (KNN) primarily looking at the average performance accuracy across three separate trials. The purpose of this study is not to simply showcase the results of the experiments but to contribute to broader discourse on the comprehension and applicability of these algorithms.

2 Methods

2.1 Learning Algorithms

2.1.1 Random Forests (RF)

The random forest learning algorithm is a learning algorithm that utilizes bagging and feature randomness to generate random subsets of features to ensure low correlation and reduced overfitting and variance of predictions [2]. I utilized a random forest algorithm with hyperparameter tuning across a grid to search for a number of estimators across 10, 50, 100, 500, and 1000 trees with the best number of estimators being reported for each trial.

2.1.2 Naive Bayes (NB)

Naive Bayes is a probabilistic classifier that assumes independence between features in the given data for prediction. It is often used for larger datasets due to greater simplicity and efficiency [3].

Submitted to 37th Conference on Neural Information Processing Systems (NeurIPS 2023). Do not distribute.

For this study, a Gaussian Naive Bayesian model was used without extra hyperparameter tuning as it does not have tunable hyperparameters like the other algorithms tested. The purpose of this model is to have a greater baseline comparison to independence of features versus independence of relationships as seen with other models.

2.1.3 K-Nearest Neighbors (KNN)

The KNN algorithm classifies data points based on distance in feature space by determining how common instances of data observations are to one another [4]. For this study, the algorithm was employed with hyperparameter tuning for the number of neighbors (k) with a weight function to influence relationship and contribution of neighbors on classification.

2.2 Performance

To assess the performance of each algorithm and trial, a systemic approach to hyperparameter tuning and accuracy measurement was employed. The RF model was tuned across the number of trees in the forest space across 10, 50, 100, 500, and 1000 trees for each trial. The KNN model was tuned across the number of neighbors by exploring how different values of k impacted the overall performance as well as creating a weight function to improve reflecting the influence of neighbors on classification.

The primary metric used for evaluating model performance was testing accuracy, which measures the proportion of correctly classified instances out of the total instances. This is a suitable metric for balanced datasets and helps provide a clear indication of the model's overall effectiveness in correct predictions across different problems.

2.3 Datasets

I compare the algorithms on 3 multinomial classification problems: (1) ADULT [5]; (2) CHESS [6]; and (3) CAR [7]. Each dataset proposes very different classification problems that utilize extreme differences in data sources and techniques necessary for prediction. The ADULT dataset from the UCI repository predicts whether an individual has an annual income of greater than $50,000 based on sociocultural features such as race, sex, and relationships. The CHESS dataset is also sourced from the UCI repository and contains whether based on positioning of a white king and rook could place a lone black king in checkmate within sixteen moves (win) or not. Finally, the CAR dataset seeks to predict how the evaluation of a vehicle would suffice based on various physical characteristics presented such as the number of doors, passenger capacity, and price of maintenance. See Table 1 for the general characteristics of these problems.

Table 1: Description of problems

Problem	# Features	Train Size	Test Size	Split
ADULT	13	9768	39074	20/80
		24421	24421	50/50
		39074	9768	80/20
CHESS	6	5611	22445	20/80
		14028	14028	50/50
		22445	5611	80/20
CAR	6	346	1382	20/80
		864	864	50/50
		1382	346	80/20

3 Experiment

For each test problem, I randomly split the data into testing and training groups into 20/80, 50/50, and 80/20 percent respectively. I utilized grid search to optimize hyperparameters for the RF and KNN models, and then utilized 5-fold cross validation on the cases to obtain the output of each trial. The evaluation metric reported is based on the overall model testing accuracy for each trial, with the final report utilizing the average accuracy of each model across the three trials per split. Each trial was run independently to ensure greater robustness and reliability of the outputs. I would ideally run

more (extensive) trials, but the random forest model can be very expensive to run on larger amounts of data, and I am still able to inquire about critical and interesting differences between the three methods across these three trials.

Table 2 shows the normalized, average accuracy score for each algorithm across each data split and dataset. For each problem and metric, the best parameter settings were utilized respectively. For each problem, the best model (across any data split) is boldfaced.

Table 2: Normalized, average accuracy score for each algorithm

Dataset	Model	Test Size (0.2)	Test Size (0.5)	Test Size (0.8)
INCOME	Random Forest	**0.8643**	0.8591	0.8524
INCOME	KNN	0.8016	0.7985	0.7893
INCOME	Naive Bayes	0.7995	0.7968	0.7963
CHESS	Random Forest	**0.8353**	0.7778	0.6538
CHESS	KNN	0.7749	0.6775	0.5364
CHESS	Naive Bayes	0.2484	0.2432	0.2245
CAR	Random Forest	0.9678	**0.9722**	0.9586
CAR	KNN	0.9220	0.9201	0.8626
CAR	Naive Bayes	0.6879	0.7176	0.7144

4 Conclusion

My experiments revealed that Random Forest consistently outperformed KNN and Naive Bayes across all dataset problems and test sizes. Random Forest, with 1000 estimators, emerged as the optimal choice in many cases, achieving high accuracy levels.

KNN and Naive Bayes, while demonstrating reasonable performance, tended to be surpassed by the robustness of Random Forest. The choice of test size (0.2, 0.5, 0.8) did not consistently impact model performance, suggesting the models' resilience to variations in the training/test data split. Notably, Naive Bayes performed much worse for the CHESS prediction problem understandably as chess positioning and moves do contain inherent spatial relationships that are significant to the overall outcome. By assuming independence between these objects, the model struggles to properly classify and predict the outcome since they are not ever truly independent from one another.

Considering the nature of the data and specific application requirements, Random Forest stands out as a reliable choice for binary classification tasks. However, the final model selection should be guided by a thorough understanding of the dataset characteristics and the context of the problem at hand.

References

[1] Rich Caruana and Alexandru Niculescu-Mizil. *Empirical comparison of supervised learning algorithms*. 2006. URL: https://www.cs.cornell.edu/~caruana/ctp/ct.papers/caruana.icml06.pdf.

[2] *Random Forest*. IBM. URL: https://www.ibm.com/topics/random-forest.

[3] *Naive Bayes*. scikit-learn. URL: https://scikit-learn.org/stable/modules/naive_bayes.html.

[4] *K-Nearest Neighbors*. scikit-learn. URL: https://scikit-learn.org/stable/modules/neighbors.html.

[5] *Census Income Dataset*. UCI Machine Learning Repository. URL: https://archive.ics.uci.edu/dataset/20/census+income.

[6] *Chess (King, Rook vs. King) Dataset*. UCI Machine Learning Repository. URL: https://archive.ics.uci.edu/dataset/23/chess+king+rook+vs+king.

[7] *Car Evaluation Dataset*. UCI Machine Learning Repository. URL: https://archive.ics.uci.edu/dataset/19/car+evaluation.