Analysis of Decision Trees, Random Forests, and Boosted Decision Trees

Ivan Seslija

Oct. 5 2023

Abstract

This report investigates the effectiveness of three machine learning algorithms, in spam classification. The study involves tuning various hyperparameters and applying a manually-implemented k-fold cross-validation to evaluate each algorithm. The results offer actionable insights into optimal settings for these algorithms, serving as a foundation for further work in binary classification tasks.

1 Introduction

In this report, I scrutinize the performance of three prevalent machine learning algorithms, Decision Trees, Random Forests, and Boosted Decision Trees (AdaBoost), utilizing the Spambase dataset for classification. The assignment has two primary objectives: Firstly, to conduct individual analyses of these algorithms by tuning different hyperparameters like pruning techniques, ensemble sizes, and maximum tree depth. Secondly, to carry out a comparative analysis through a k-fold cross-validation method that I've implemented manually. This will help to fine-tune these algorithms and directly contrast their efficacy.

This assignment is designed to deepen my understanding of the performance nuances of each algorithm and the impact of hyperparameter adjustments. The findings will serve as a basis for future work in machine learning, specifically in binary classification scenarios such as spam filtering.

2 Part I - Separate Analysis

2.1 Methodology

Various hyperparameters, including pruning methods for decision trees, ensemble sizes for random forests, and maximum tree depths were systematically adjusted. The primary focus was to understand how each algorithm performs under different configurations and to lay the groundwork for more nuanced analyses in subsequent parts of the experiment.

2.2 Decision Trees

The objective of this experiment was to assess the implications of different test sizes and Cost Complexity Pruning Alpha (ccp alpha) values on the efficacy of Decision Trees, using the Spambase dataset. The primary goal is to comprehend how these key hyperparameters can influence the algorithm's overall performance in terms of accuracy and efficiency.

This investigation is structured into two separate but interlinked experiments:

- **First Experiment**: Focuses on varying ccp alpha values while keeping the test size constant. **ccp alpha values**: [0.0, 0.001, 0.01, 0.1]
- **Second Experiment**: Focuses on varying the test size while keeping the ccp alpha constant. **Test sizes**: [0.2, 0.3, 0.4, 0.5]

In both experiments, he split criterion remains constant to isolate the effects of the ccp alpha and test size variations.

2.2.1 Experiment 1: Entropy

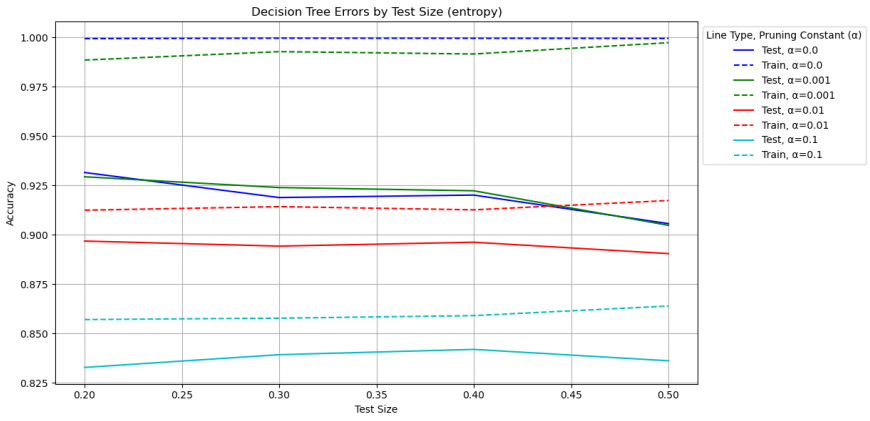

Figure 1

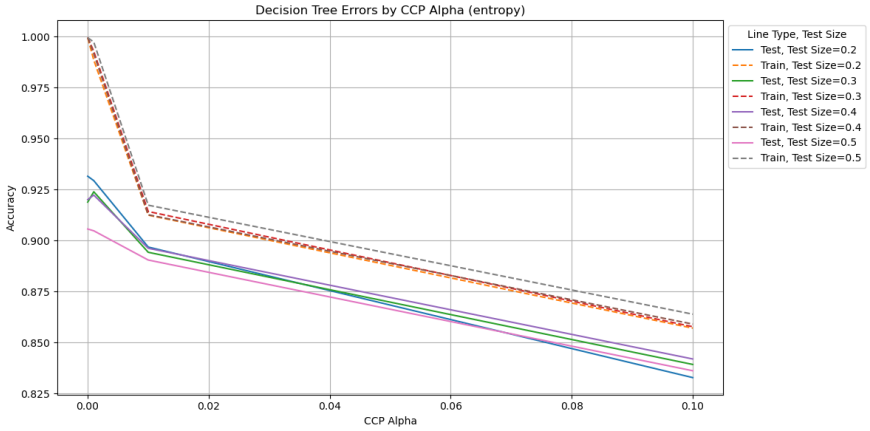

Figure 2

In the experiment, it's important to highlight that signs of overfitting emerge when the test size exceeds 40 percent. Additionally, the influence of pruning is considerable. Elevated pruning levels restrict the model's capacity to generalize, resulting in classifications that are more specific but not necessarily more accurate. Interestingly, this trend is evident even within the training dataset. With no pruning, the model attains flawless accuracy on the training data; however, this performance diminishes as the extent of pruning escalates.

Upon examining the individual alpha values, it's evident that both 0 and 0.001 yield comparably high performance, with the latter showing a marginal improvement at larger test sizes. This suggests that a minimal degree of pruning may actually contribute to more accurate labeling, rather than detract from it.

2.2.2 Experiment 2: Gini

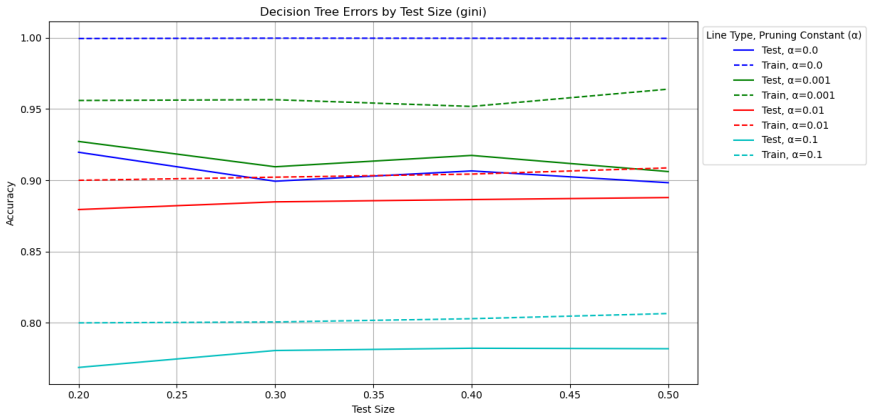

Figure 3

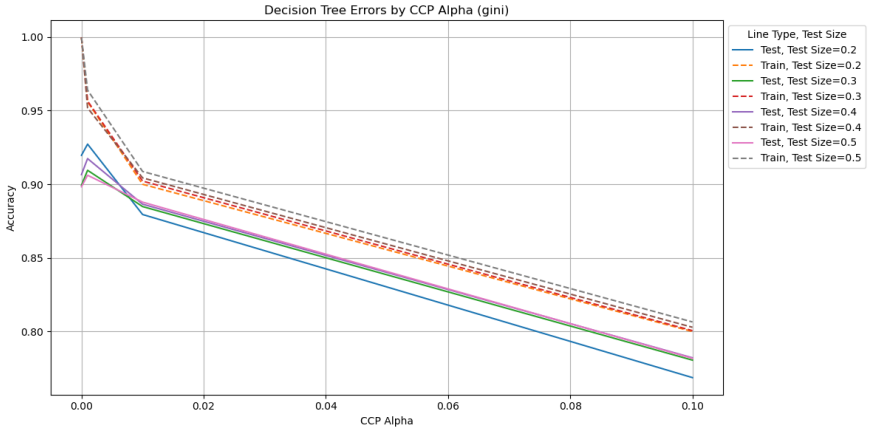

Figure 4

For the gini criterion, the observations largely mirror those made for the alpha values. Both 0 and 0.001 demonstrate similar levels of effectiveness, this time with a more constant improvement with the pruning. This implies that a modest amount of pruning could enhance the model's labeling accuracy under the gini criterion as well.

2.2.3 Conclusion

This experiment provides valuable insights into several key factors affecting decision tree performance, including test size, pruning level, and criterion selection. A 40 percent test size serves as a noteworthy threshold for overfitting, signaling the need for meticulous parameter selection. The role of pruning is significant, with a slight but consistent benefit observed for an alpha value of 0.001, and detriment at higher pruning levels.

Interestingly, only marginal differences were observed between the 'gini' and 'entropy' split criteria, suggesting that for this specific dataset, either criterion could be effectively used without substantially affecting the model's accuracy.

2.3 Random Forests

The objective of this experiment was to scrutinize the effects of various test sizes and hyperparameters on the performance of the RandomForestClassifier, utilizing the Spambase dataset. This aims to understand the how these hyperparameters affect both the efficiency and accuracy of the algorithm.

The study is divided into two experiments. The first experiment focuses on modifying individual hyperparameters: 'Number of Trees' (`n_estimators`), 'Max Features' (`max_features`), and 'Max Samples' (`max_samples`), while keeping the test size fixed. The range for these hyperparameters is as follows:

- `n_estimators`: [5, 10, 20, 50, 100]
- `max_features`: ['sqrt', 'log2', 0.3, 0.5, 0.7]
- `max_samples`: [5, 4, 3, 2, 1]

In the second experiment, the approach is reversed. Here, the test size is varied across four different fractions of the dataset: [0.2, 0.3, 0.4, 0.5], while each hyperparameter is held at a constant value.

Both training and test accuracies are measured for each parameter configuration, and the results are visualized through 2D line graphs.

2.3.1 Experiment 1: Number of Trees

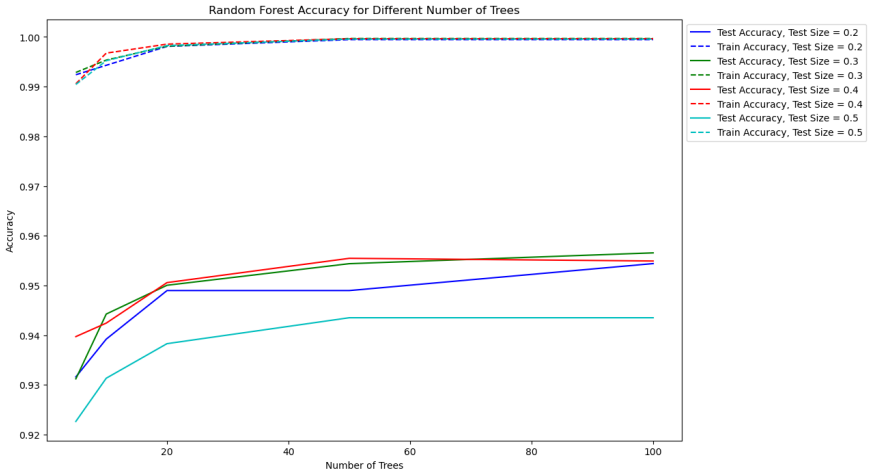

Figure 5

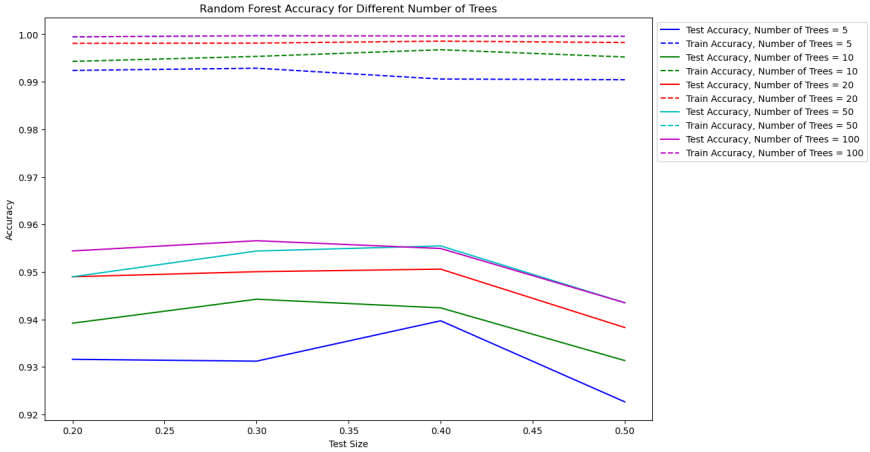

Figure 6

In the initial experiment utilizing the Random Forest algorithm, we altered the quantity of trees involved in the model. The observations indicate a typical overfitting trend manifesting beyond the 40 percent mark. However, what stands out is that the marginal gain in model performance plateaus when the number of trees reaches approximately 50. Beyond this point, additional trees contribute little to no further improvement.

This insight is valuable primarily because it offers an opportunity for resource optimization. My hypothesis is that once the model accumulates a sufficient number of effective trees, they collectively produce a majority vote that is often correct. Therefore, adding more trees beyond this point results in negligible impact on the model's predictive power.

2.3.2 Experiment 2: Number of Random Features

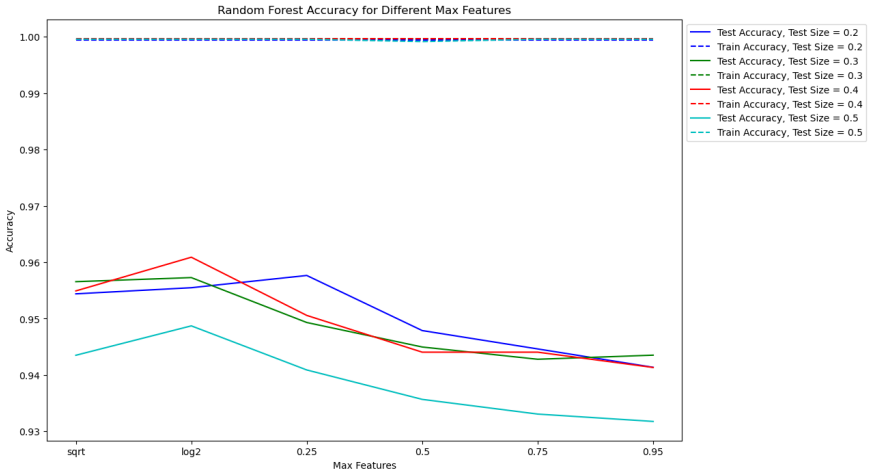

Figure 7

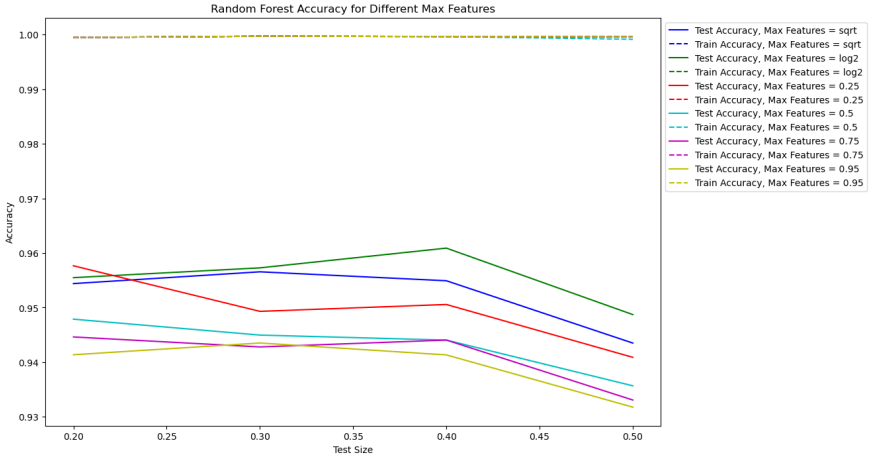

Figure 8

In our second experiment, we varied the maximum feature size for the Random Forest model. We observed that the model achieved its peak performance when the maximum feature size was set to log base 2. Interestingly, as we continued to increase the maximum feature size beyond this point, the performance of the model started to decline.

This finding suggests that optimal feature selection plays a crucial role in model efficiency. Utilizing a maximum feature size of log base 2 appears to be the sweet spot for maximizing predictive accuracy, while further increases in feature size lead to diminishing returns in performance.

When the maximum feature size is increased beyond log base 2, the model may start to include less relevant or noisy features. This could dilute the importance of more meaningful features, thereby negatively affecting the overall predictive performance.

2.3.3 Experiment 3: Number of Max Samples

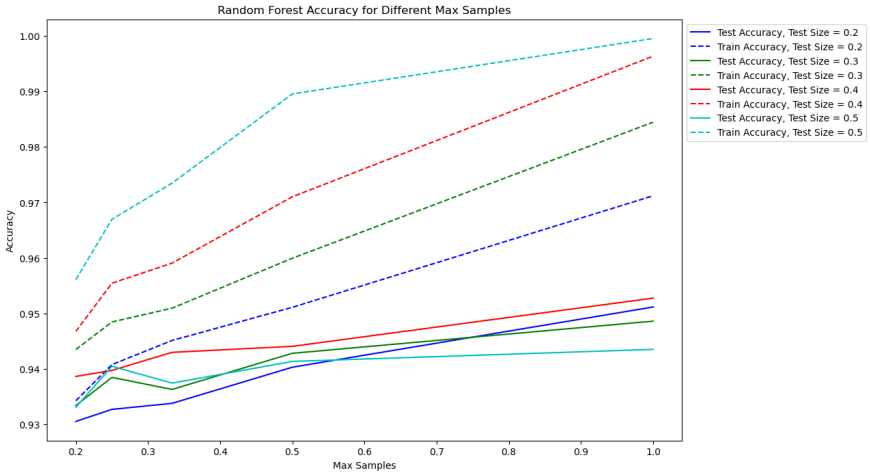

Figure 9

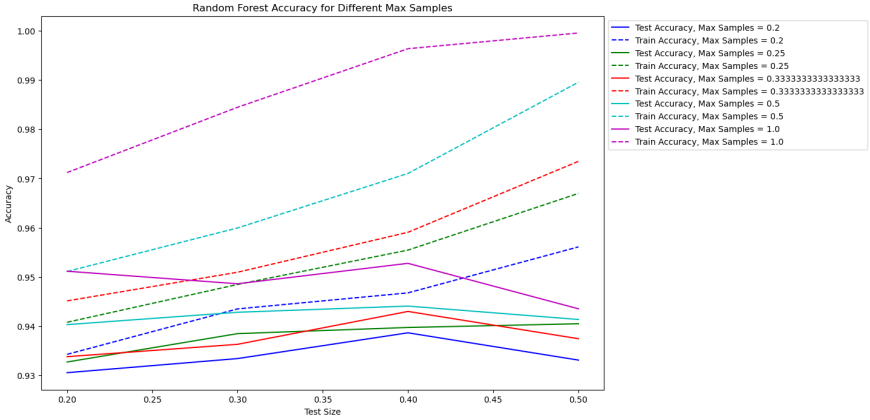

Figure 10

In the third experiment, we focused on altering the number of samples drawn from the dataset X for training each base estimator within the Random Forest model. We observed a direct relationship between the number of samples used and the model's accuracy; as we increased the number of samples, the accuracy of the model correspondingly improved.

This trend likely indicates that increasing the number of samples enhances the model's learning capability by providing a richer, more comprehensive set of data for each base estimator to train on. As a result, each tree in the ensemble can capture more nuanced patterns in the data, thereby increasing the overall predictive power of the Random Forest model.

2.3.4 Conclusion

These series of experiments with the Random Forest algorithm yielded valuable insights into optimizing its performance. In the first experiment, it was found that increasing the

number of trees beyond 50 offers diminishing returns, highlighting the importance of resource optimization. The second experiment revealed that the model performs optimally when the maximum feature size is set to log base 2, emphasizing the role of feature selection. Finally, the third experiment demonstrated a direct correlation between the number of samples used to train each base estimator and the model's accuracy.

Collectively, these findings provide a roadmap for fine-tuning Random Forest models. They highlight the need to balance the number of trees and feature size for optimal performance, while also stressing the importance of using a sufficiently large sample size for training. These optimizations not only improve model accuracy but also contribute to more efficient use of computational resources.

2.4 Boosted Decision Trees

The objective of this experiment is to evaluate the impact of varying test sizes and hyperparameters on the performance of AdaBoost classifiers with Decision Trees, using the Spambase dataset. The aim is to understand how these parameters influence both the efficiency and accuracy of the algorithm.

The investigation is divided into two distinct experiments. The first experiment focuses on modifying individual hyperparameters: 'Max Depth of the Tree' (`max_depth`) and 'Number of Trees' (`n_estimators`), while keeping the test size constant. The range for these hyperparameters is as follows:

- `max_depth`: [1, 5, 10, 25, 50]
- `n_estimators`: [5, 10, 20, 50, 100]

In the second experiment, the approach is reversed. Here, the test size is modified across four different fractions of the dataset: [0.2, 0.3, 0.4, 0.5], while each hyperparameter is held constant.

Both training and test accuracies are recorded for each parameter configuration. The results are visualized using 2D line graphs and a 3D scatter plot for a more comprehensive analysis.

2.4.1 Experiment 1: Max Depth

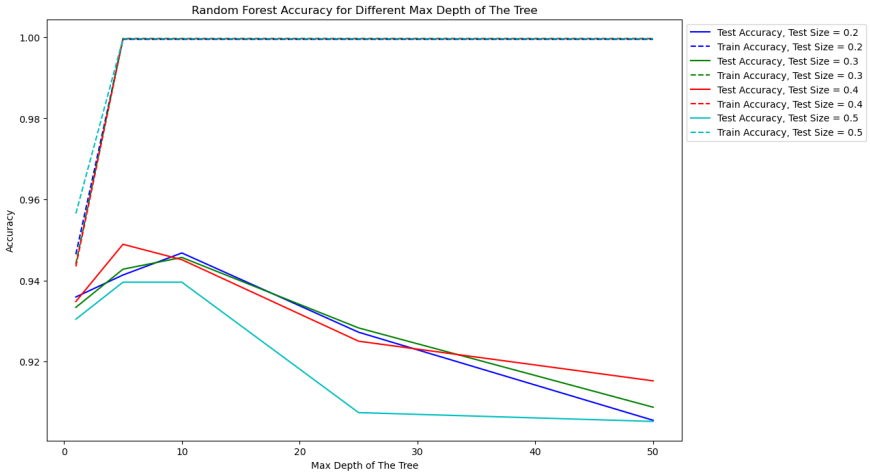

Figure 11

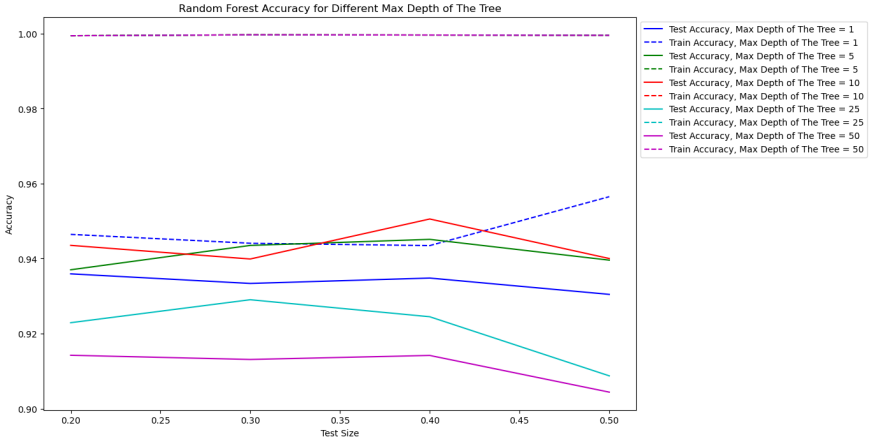

Figure 12

In the initial experiment with Boosted Trees, we focused on manipulating the maximum depth of the individual trees within the ensemble. As we increased the depth of the trees, the training accuracy soared to 100 percent. However, the testing accuracy, after reaching a depth of around 10, began to decline, peaking at around 94-95 percent.

This pattern suggests that while deeper trees excel at capturing the intricacies of the training data—hence the perfect training accuracy—they also risk overfitting. The declining test accuracy beyond a depth of 10 implies that the model becomes too specialized in the training data and loses its ability to generalize well to new, unseen data.

2.4.2 Experiment 2: Number of Trees

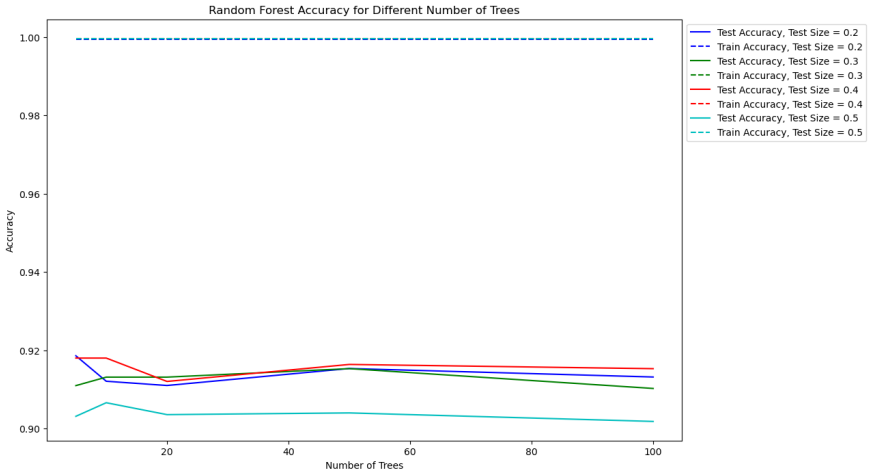

Figure 13

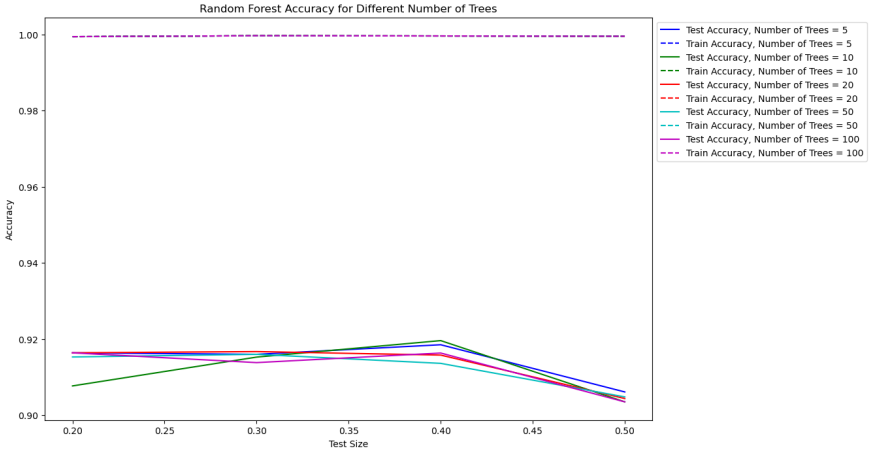

Figure 14

In the case of varying the number of trees in the Boosted Trees ensemble, our observations showed a relatively consistent performance across different settings. Unlike our experience with Random Forest, where increasing the number of trees led to diminishing returns, the Boosted Trees model maintained a stable level of accuracy regardless of the number of trees used.

This characteristic can be advantageous as it allows for more flexibility in resource allocation. Whether you have fewer or more trees, the Boosted Trees model appears to sustain a relatively consistent level of performance, indicating that computational resources can be managed more efficiently without significantly sacrificing accuracy.

2.4.3 Conclusion

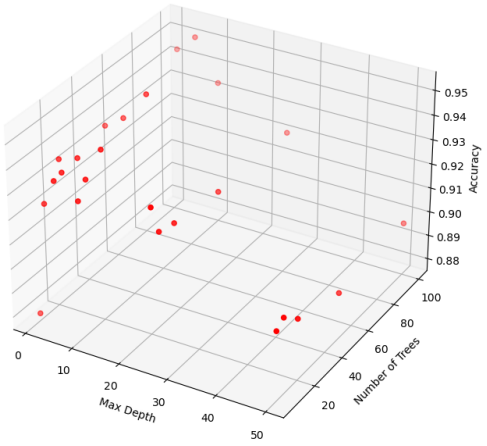

Figure 15

The 3D graph, with maximum tree depth on the X-axis, the number of trees on the Y-axis, and model accuracy on the Z-axis, vividly illustrates the findings. It becomes clear from the graph that while the number of trees remains a relatively neutral factor, the maximum depth of the trees plays a pivotal role in influencing the model's accuracy.

These observations underscore the importance of carefully selecting tree depth as a key hyperparameter when tuning Boosted Trees models. They also suggest that efforts to improve the model could focus more on optimizing tree depth rather than increasing the number of trees, thereby achieving a balance between performance and efficiency.

3 Part II - Comparative Analysis

3.1 Methodology

The objective of Part II is to conduct a comparative analysis between Random Forests and Boosted Decision Stumps using k-fold cross-validation to fine-tune a specific hyperparameter—the size of the ensemble. The performance of the two tuned methods will be evaluated on a test set that has not been used during the tuning process.

Both the RandomForest and AdaBoosting algorithms will use their default values, except the ensemble size which is as follows:

Ensemble sizes to test: [1, 2, ..., 100]

Post-tuning, we'll evaluate the test errors of both models using a separate, untouched test set to determine which algorithm is more effective on new data.

3.2 Random Forests

Default Settings:

- criterion: "gini"
- max depth: None
- min samples split: 2
- min samples leaf: 1

We keep the settings for the random forest to the defaults, and test several values for the "n estimator".

Running the experiment with the random forest gave us an optimal forest size of 88 trees with an accuracy of 93.1 on the training data. The Random Forest algorithm achieved an accuracy rate of approximately 82.4 percent on the unseen test data.

3.3 Boosted Decision Trees

Default Settings:

- learning rate: 1.0
- algorithm: "SAMME.R"
- base estimator: DecisionTreeClassifier with max depth=1 (Decision Stump)

We keep the settings for adaboost to the defaults, and test several values for the "n estimator".

Running the experiment with adaboost gave us an optimal forest size of 55 trees with an accuracy of 89.7 on the training data. The Adaboost algorithm achieved an accuracy rate of approximately 77.2 percent on the unseen test data.

3.4 Comparative Analysis

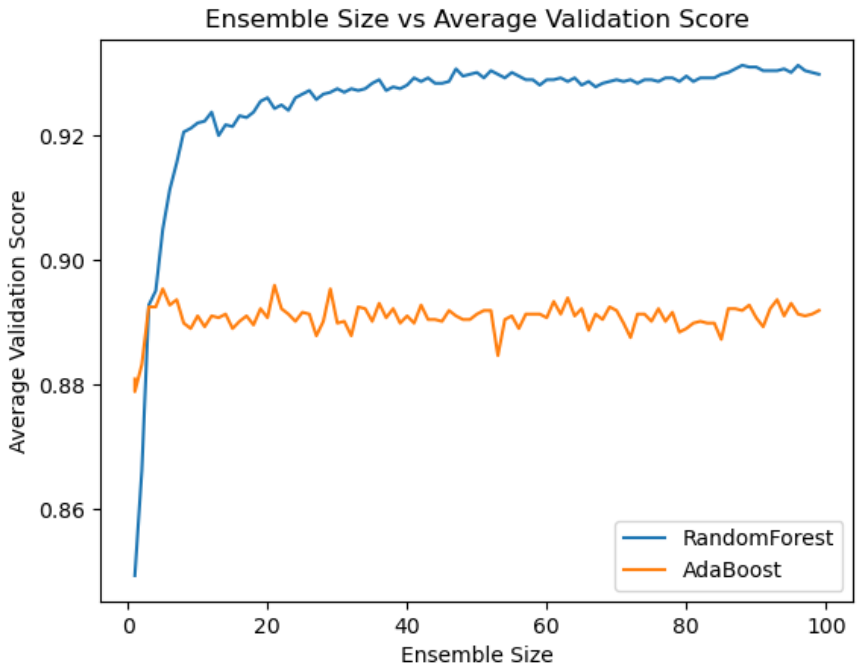

Figure 16

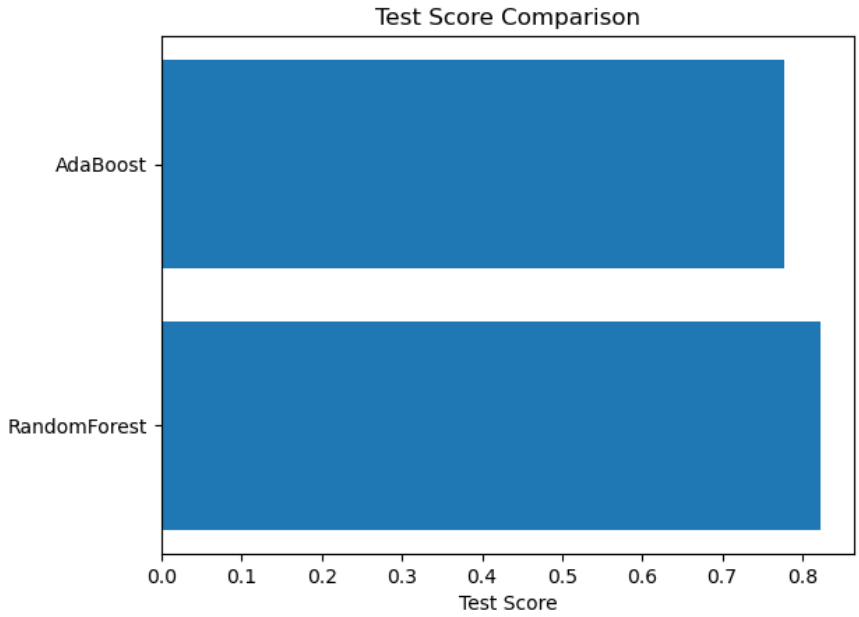

Figure 17

In this k-fold cross-validation experiment, Random Forest, using its default settings with our fine tuned amount of estimators, significantly outperformed AdaBoost, which was similarly configured. Notably, AdaBoost utilized decision stumps (single-level decision trees) as

its base estimator, while Random Forest employed full-fledged decision trees. While both permored well on the k-fold cross-validation, we had them losing about 10 percent accuracy when shifted to the test data. Regardless, in both scenarios, the random forest outperformed adaboost by a margin of about 5 percent.

This disparity in performance could be attributed to Random Forest's ensemble approach of using multiple decision trees, which offers a more accurate and stable prediction. On the other hand, AdaBoost's use of decision stumps may have made it more sensitive to noisy data and outliers in the spam dataset, affecting its overall performance. Should we expand the base estimator to be a tree itself, we may see improved results, though at the cost of time. Therefore, based on these results, Random Forest appears to be a more reliable choice for spam classification in this particular scenario.

4 Conclusion

The key takeaways from this study are:

- Decision Trees generally perform well with smaller test sizes and may benefit from some pruning for optimal results.
- Random Forests tend to perform optimally when the number of trees and feature size are well-balanced.
- AdaBoost classifiers often achieve good accuracy with moderate tree depth and ensemble size.

The experiments conducted on decision tree performance, Random Forest, and Boosted Trees models offer comprehensive insights into optimizing machine learning algorithms. Key factors such as test size, pruning level, and criterion selection significantly influence decision tree models, with a 40 percent test size identified as a threshold for overfitting. For Random Forest models, the number of trees, feature size, and sample size for training are crucial for optimal performance. Boosted Trees models, on the other hand, benefit more from optimizing tree depth rather than increasing the number of trees. These findings collectively serve as a guide for fine-tuning these models, emphasizing the importance of specific hyperparameters and efficient use of computational resources.

In light of our k-fold cross-validation experiment, it's worth noting that Random Forest consistently outperformed AdaBoost by approximately 5 percent in both the cross-validation and test data scenarios. This suggests that Random Forest's ensemble approach, which leverages multiple fully developed decision trees, offers a more robust and accurate prediction model. On the other hand, AdaBoost's reliance on decision stumps as its base estimator may make it more vulnerable to noisy data and outliers, as observed in our spam dataset. While enhancing AdaBoost's base estimator to a full decision tree could potentially improve its performance, this would likely increase computational time.

These findings not only contribute to the understanding of these algorithms but also offer actionable insights for practitioners aiming to deploy machine learning models for spam classification or similar binary classification tasks.