ECE 759 Project 1 Part 2

Matt Conrad and Evan Williams

April 20, 2018

Abstract

In this report, we build on the foundations of the first part of the project by validating a choice of hyperparameters for the algorithm and evaluating the performance of each algorithm. Finally, we conclude with a series of proposed improvements to boost the performance of the classifiers in future work.

1 Introduction

The fields of machine learning, image processing, and computer vision are continually growing and becoming further intertwined to create state-of-the-art technology, such as real-time facial recognition. As these technologies become increasingly complex, it is essential to maintain an understanding of basic algorithms and techniques that the complex techniques rely on.

The two fundamental machine learning algorithms covered here include Linear Discriminant Analysis (LDA) and Decision Trees. Both are a form of classifier that require supervised training in order to classify new feature vectors into one class or another. While they do aim to perform the same task, LDA classifies linearly whereas the Decision Tree does so nonlinearly [TK09].

Although these classifiers are useful tools in themselves, they require feature datasets for training and testing. The raw data for this project was presented in the form of raw images in the publicly available MNIST and Caltech10 image sets. In this form, image processing and computer vision algorithms were needed for transforming the raw images into usable feature vectors.

2 Cross Validation

Cross Validation is a technique used to assess how well a model will generalize to an independent data set. One common technique employed is k-fold cross validation where the training dataset is split into k equally sized partitions. The model is then evaluated using 1 partition as the validation set and the remaining k-1 partitions as training data for the model. Repeating this process so that each of the k partitions acts as the validation set yields a total of k validation performance metrics for the model. In the case where the performance metric is accuracy of prediction on the validation set these validation accuracies can be averaged into a generalized validation accuracy for the model which helps prevent the problem of overfitting. By performing this process on a number of models with various hyperparameters you can find the model with the hyperparameters that provide the best generalized performance on the training data before evaluating performance on the test set.

For this project we performed 5-fold cross validation twice on the Decision Tree model on both the MNIST and Caltech-10 datasets. We did not perform cross validation on the LDA classifier as it does not have any hyperparameters to tune. Ultimately we found the optimal hyperparameters for the MNIST dataset were Max Splits = 25, Stopping Criteria = 0.01, Max Depth = 13, and Minimum Leaf Size = 750 which corresponded to a generalized validation accuracy of 75.85 %. The optimal hyperparameters for the Caltech-10 dataset were Max Splits = 20, Stopping Criteria = 0.01, Max Depth = 5, and Minimum Leaf Size = 10 which corresponded to a generalized validation accuracy of 26.19%.

For the MNIST Dataset we searched over the following hyperparameters:
MaxSplits = [15,20,25];
MaxDepths = [5,9,13];
StopCriterias = [0.01];
MinimumLeafSize = [100,250,500,750];

For the Caltech Dataset we searched over the following hyperparameters:
MaxSplits = [10,20,30,40,50,60,70];
MaxDepths = [3,5,7,9,11,13,15,17,50];
StopCriterias = [0.01,0.1,0.2];
MinimumLeafSize = [10];

3 Demonstration of Performances

To begin this section, we will begin with discussion of the results of the LDA classifier on the MNIST dataset. This algorithm does not have any hyperparameters to tune thus there was no need for cross validation of this algorithm. Overall, the performance of the LDA classifier on the MNIST dataset was quite good. These results are shown below in table 1. The bag of features feature extractor was used to extract features from the Caltech-10 dataset. These features performed significantly better than others such as Histogram of Gradients which had only single digit accuracy. The full results are shown below in table 2. The same features were evaluated on the Decision Tree classifier which utilizing the optimal hyperparameters found above yielded the results seen in tables 1 and 2. Overall, the performance of this classifier was worse than the LDA classifier by 20% in each category.

Table 1: Table of results for MNIST Dataset

	Training Set Accuracy	Testing Set Accuracy
LDA	98.22%	97.55%
Decision Tree	77.11%	75.96%

Table 2: Table of results for Caltech Dataset

	Training Set Accuracy	Testing Set Accuracy
LDA	71.90%	39.04%
Decision Tree	53.8%	24.76%

To replicate these results please use the following process:

- First, load in the datasets and perform feature extraction. Examples of this process are shown in the first few sections of ECE759_Project.m.
- Secondly, train the classifier and make predictions using the LDA and GreedyDecisionTree2 function calls.
- LDA(TrainFeatures, TrainLabels, TestFeatures) returns the predicted classification of the features presented in the TestFeatures input.
- GreedyDecisionTree(TrainFeatures, TrainLabels,TestFeatures, MaxSplits, StoppingCriteria, MaxDepth, MinSplitSize) returns the predicted classification of the features presented in the TestFeatures input. Please input the hyperparameters for this function as the optimal ones found during the cross validation process which are described above.
- Following this, the accuracy can be computed by comparing the predicted labels and the true labels.
- Complete examples of this process can be found in the CaltechGDT.m and CaltechLDA.m scripts.

3.1 Convergence

The LDA algorithm was trained 100 times on both the MNIST and Caltech-10 datasets. The mean convergence time was reported along with the standard deviation for each image dataset below. LDA Average Convergence time for MNIST is 31.869200 +/- 2.095990 seconds. The LDA average convergence time for the Caltech-10 dataset is 1.419451 +/- 0.053592 seconds.

To plot convergence for the Decision Tree Algorithm the average impurity at each depth level was calculated and then plotted in figure 1. This figure shows a general downward trend in the impurity as you traverse along the tree which shows that the algorithm converges as the tree continues to grow.

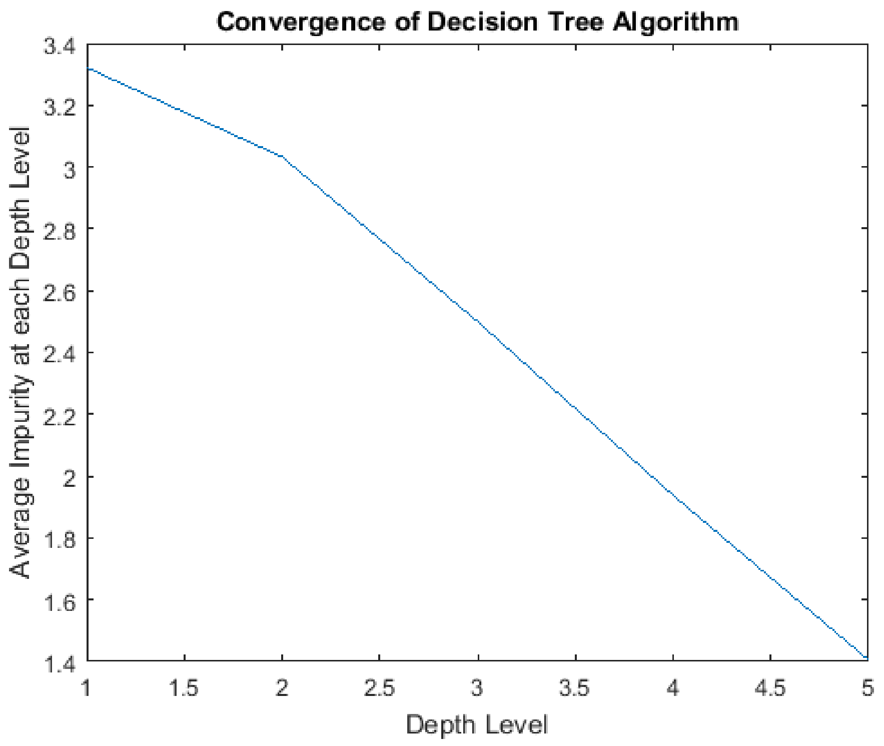

Figure 1: LDA Performance versus fraction of utilized samples in the training set of the MNIST dataset

3.2 Performance vs. Time

To test the performance of the two models versus time we implemented techniques to test the time dependence of each algorithm. For LDA we varied the number of training samples utilized in the training to decrease training time whereas for the Decision Tree we varied the number of nodes allowed in the tree. As you can see below in figures 2 and 3 the effects were rather straightforward for LDA where in general as the amount of samples were increased the classification accuracy increased. Additionally, LDA tended to reach a steady state after which additional training had only marginal improvements to the prediction accuracy of the algorithm. The effects on decision tree were similar where allowing additional splits of the tree led to general improvements in the performance of the model to the data as can be seen in figure 4.

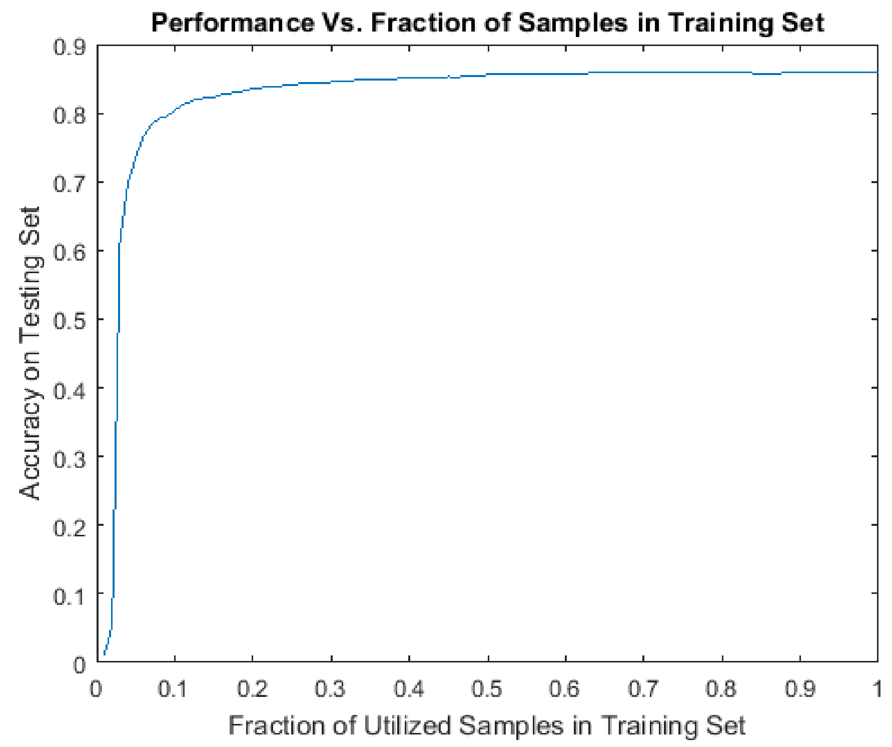

Figure 2: LDA Performance versus fraction of utilized samples in the training set of the MNIST dataset

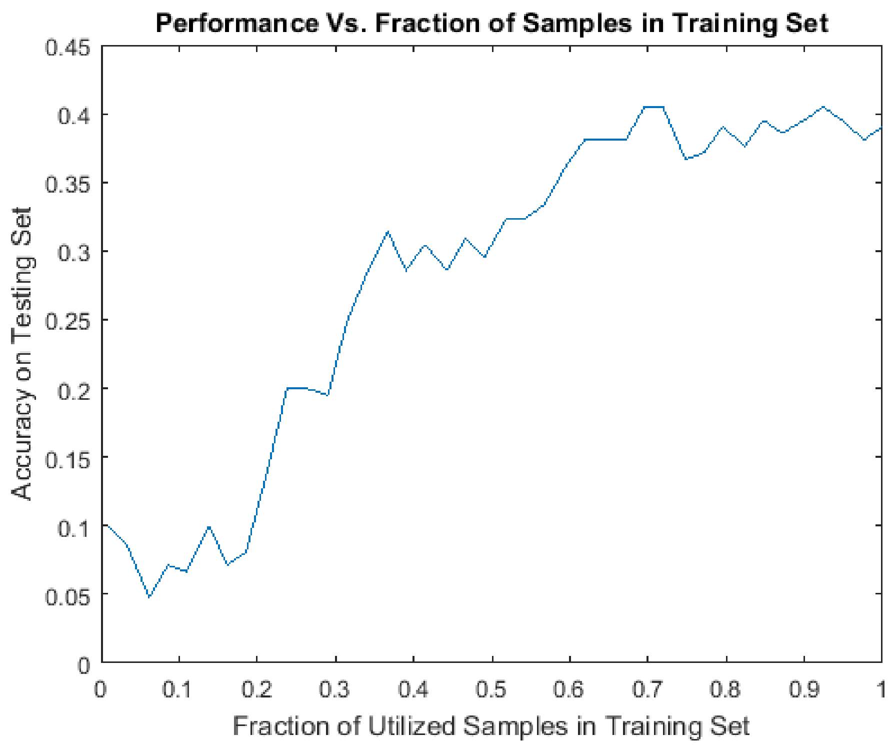

Figure 3: LDA Performance versus fraction of utilized samples in the training set of the Caltech dataset

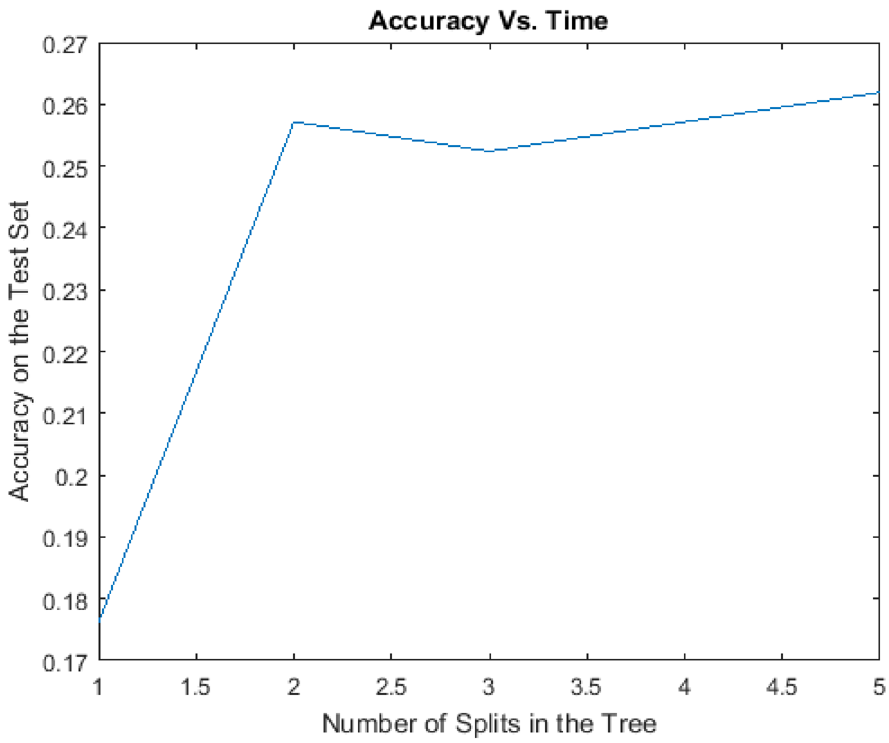

Figure 4: Decision Tree Performance versus the number of splits in the tree which dictates how quickly the algorithm runs.

3.3 Performance vs. Different Hyperparameters

In this section we report the performance of the Decision Tree classifier against various hyperparameters. The LDA classifier is absent from this section as it does not have any hyperparameters.

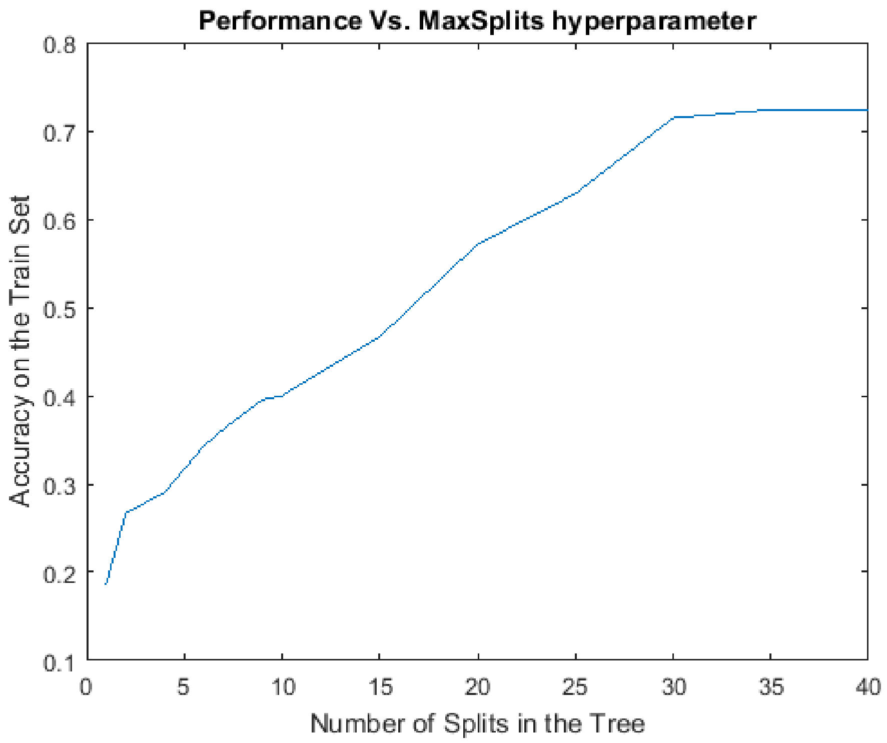

Figure 5: Decision Tree Performance versus the max number of splits in the tree on the Caltech-10 dataset.

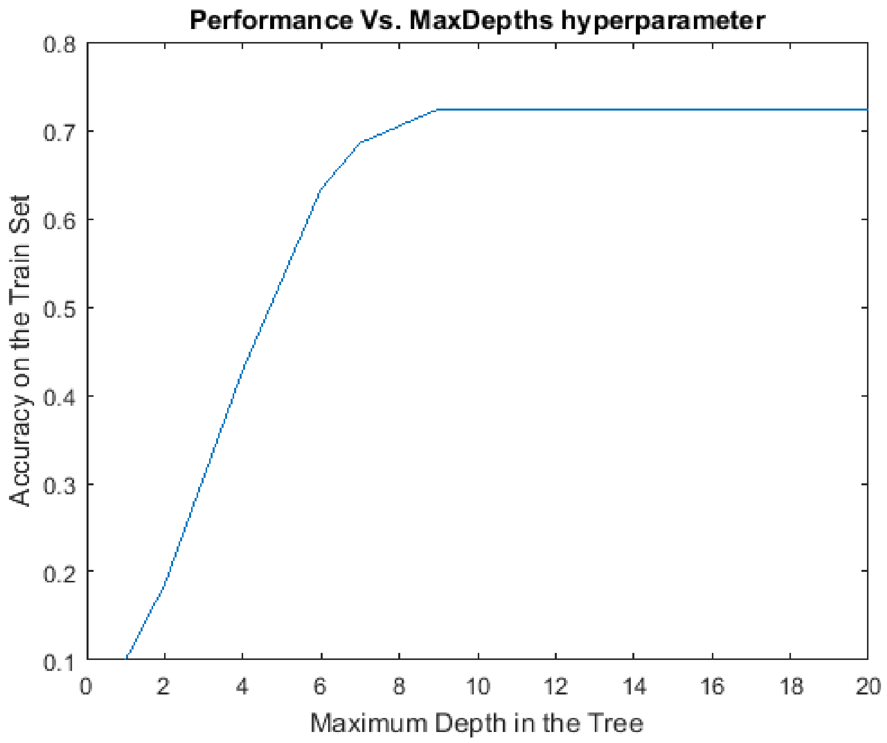

Figure 6: Decision Tree Performance versus the max depth of leaves in the tree on the Caltech-10 dataset.

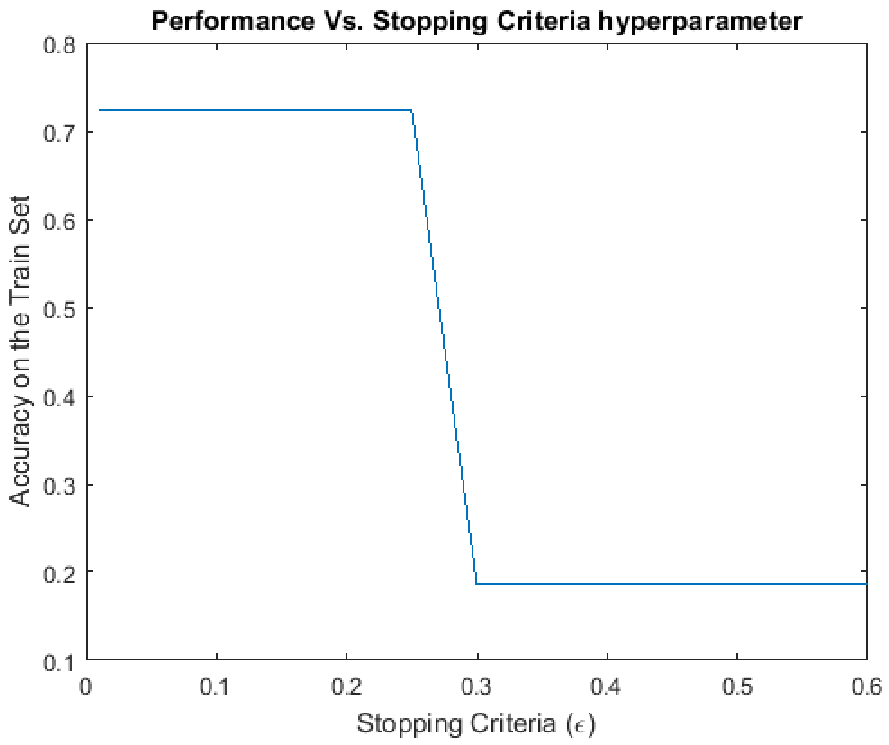

Figure 7: Decision Tree Performance versus the stopping criteria on the Caltech-10 dataset.

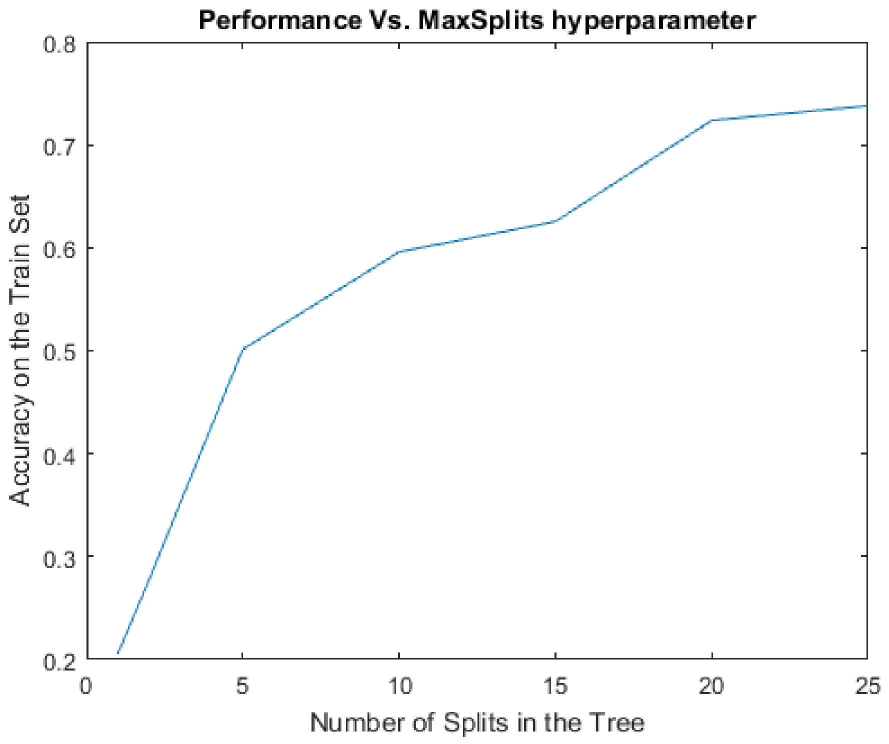

Figure 8: Decision Tree Performance versus the max number of splits in the tree on the MNIST dataset.

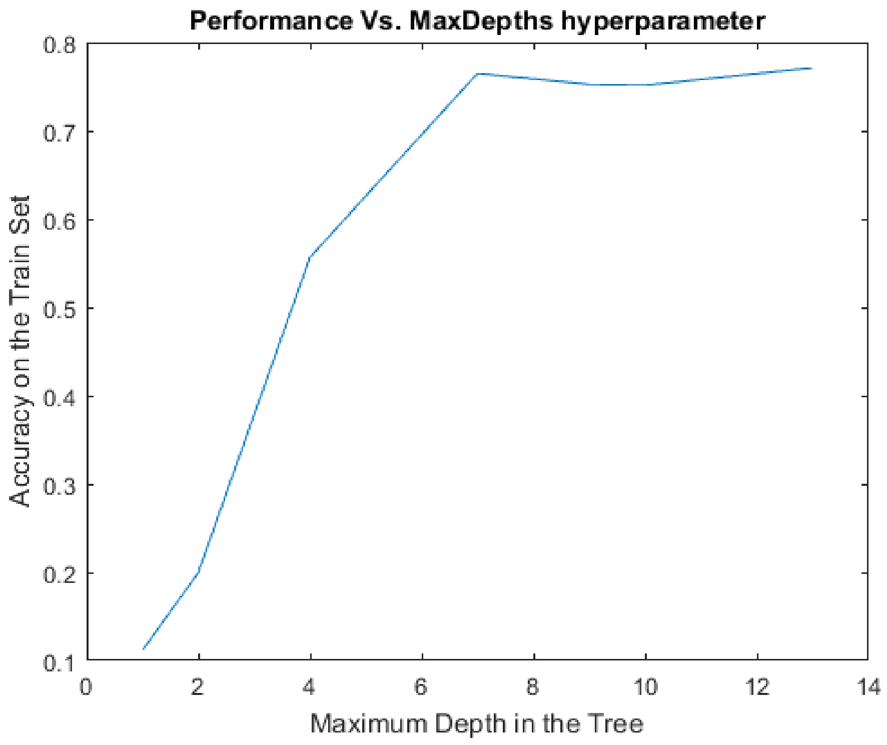

Figure 9: Decision Tree Performance versus the max depth in the tree on the MNIST dataset.

4 Analysis of Results

4.1 Advantages and Drawbacks of each Algorithm

In light of the results, an obvious advantage of Decision Tree over LDA is that the Decision Tree is more customizable for the problem at hand. LDA required no hyperparameters to train, whereas Decision Tree takes a handful of parameters to get variously shaped trees. This aspect allows the designer to successfully apply this classifier to a wider range of applications and in some cases perform better than the LDA classifier. With greater customizable control, the Decision Tree also opens up the possibility of improvements in the form of bootstrapping or pruning; if LDA is applied to a problem, then there is little the designer can do to improve those results.

In both cases, LDA performed better than the Decision Tree by roughly 20 and 15 percent for MNIST and Caltech, respectively. While the LDA did well here, the Decision Tree may come at a disadvantage in this image classification application since Decision Tree is an algorithm best suited for categorical problems such as medical diagnosis and probably not image classification as shown in this report.

One fact about the LDA classifier is that it is reliant on the assumption that the underlying distribution is Gaussian. In this application, this assumption is fine since many natural occurring images, such as those in Caltech, follow a Gaussian distribution. In other settings, this Gaussian assumption may act as a disadvantage.

4.2 Proposed Improvements for Smooth Performance

LDA would be primarily improved through the use of better feature generation and feature selection. There is a wealth of literature on the MNIST and Caltech classification problems which could be exploited to provide better features. This process would be beneficial for the decision tree classifier as well. Furthermore, a more comprehensive hyperparameter search during the cross-validation stage could provide a more optimal set of hyperparameters to improve the performance of the decision tree. Finally, ensemble methods such as random forests could be employed to bolster performance compared to that of a single decision tree.

References

[TK09] S. Theodoridis and K. Koutroumbas. *Pattern Recognition.* Associated Press, 4 edition, 2009.