Mahon Khoshzaban
Satish Solanki
Thomas Pierron de Mondesir

This project used decision trees and ensemble techniques in order to predict whether or not a species of mushroom is poisonous based on its features. All participants played an equal role in this project (33.3% each).

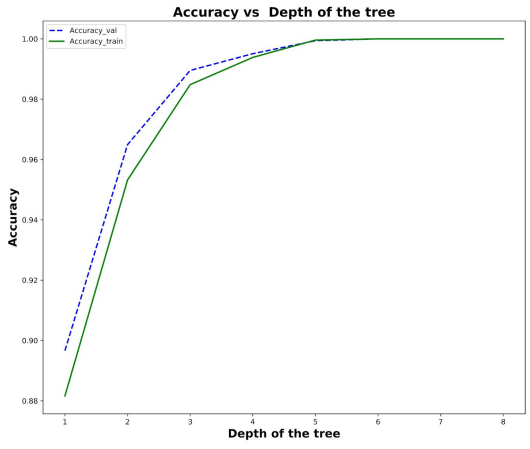
Part 1: Decision Tree

Table 1 and Figure 1 summarize the training and validation accuracy for a binary decision tree of varying depths that was built on the training data:

Table 1: Numeric accuracies for training and validation for binary decision tree of varying depths

Depth	Accuracy_train	Accuracy_val
1	0.8816167419	0.8966153846
2	0.9532211736	0.9649230769
3	0.9848173984	0.9895384615
4	0.9938448913	0.9950769231
5	0.9995896594	0.9993846154
6	1	1
7	1	1
8	1	1

Figure 1: Graphical Representation of Training and Validation Accuracy for Decision Tree



First, the tree was made with a depth of 2 which was used as a baseline to determine accuracy. The depth 2 tree had a training accuracy of 0.953 and a validation accuracy of 0.964. Interestingly, the validation accuracy was higher than the training accuracy, and this was observed as depths increased as well. One explanation for this is that the validation data is cleaner, and may be easier to classify whereas the training data by comparison may have been more difficult to create a model with. Unsurprisingly, the accuracy for both train and validation improved as the depth of the tree increased. This is because a decision tree with more splits is able to classify more accurately based on the principle of measuring similarities with more features. After depth 5, 100% accuracy was obtained on training and validation data. Therefore, in the interest of keeping the trees as small as possible, depth 6 was determined to be the best for accuracy.

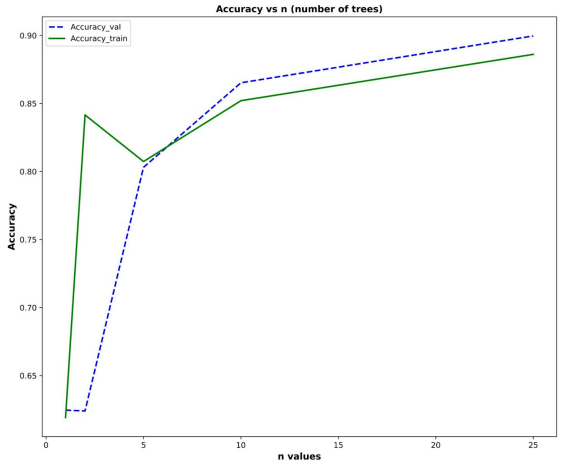
Part 2: Random Forest

For this section, the random forest implementation was used to create the trees and make predictions. Depth was held constant at 2 for all aspects of the random forest, but the number of trees *n* and the number of random feature samples *m* varied. First, for *m=5* and n being in the set of [1,2,5,10,25], the following results in Table 2 and Figure 2 were obtained:

Table 2: Numeric Accuracy Determined by Varying the Number of Trees

Number of trees	Accuracy_train	Accuracy_val
1	0.6192039393	0.6246153846
2	0.8416085351	0.624
5	0.8073450964	0.8030769231
10	0.8520722199	0.8652307692
25	0.8861304883	0.8996923077

Figure 2: Graph of Accuracy Determined by Varying Number of Trees



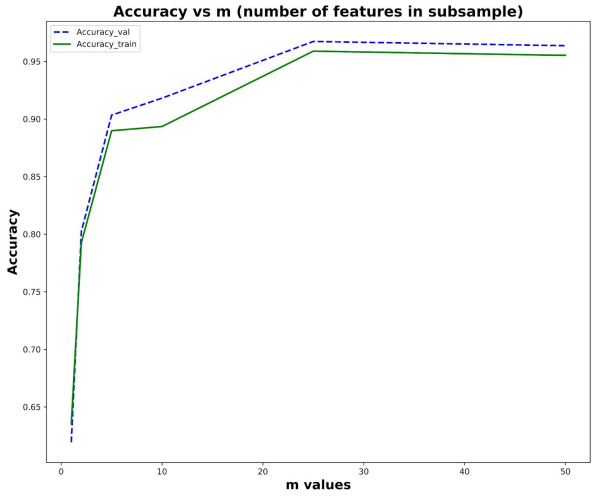
As expected, increasing the number of trees in the forest did also increase the train/validation data accuracy. However, the overall curve for the training data accuracy does not completely fit the expectation for how the accuracy should be increasing; increasing the number of trees should logarithmically increase the accuracy, which is seen in the validation curve. Fundamentally this is because more trees correspond to more votes, and a greater probability of the majority vote being more accurate. This logarithmic curve is observed for n = 5 or greater for training accuracy. At n = 2 there was a major disconnect between the training and validation accuracy, and this is most likely caused by the fact that having a forest made of 2 trees each with equal vote could more easily result in a “bad” tree having half of the say in the overall prediction. However, as a whole, adding more trees did correspond with a positive correlation to accuracy.

Additionally, the number of random feature samples, *m*, was also varied while the number of trees was kept constant (n = 15). Table 3 and Figure 3 summarizes the results:

Table 3: Training and Validation Accuracies From Varying *m*

m	Accuracy_train	Accuracy_val
1	0.635207222	0.6190769231
2	0.7925728355	0.8024615385
5	0.8898235535	0.9033846154
10	0.8935166188	0.9181538462
25	0.9589659417	0.9673846154
50	0.9552728765	0.9636923077

Figure 3: Graphical Representation of Training/Validation Accuracies with Varying *m*



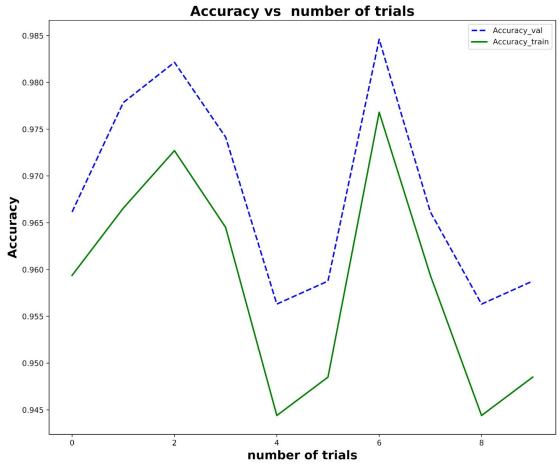
As expected, increasing the number of random features produces higher accuracy for both training and validation. The reason for this is because, unlike how increasing the number of trees reduces the variance, *m* is ultimately a hyperparameter that reduces the bias by decorrelating the trees in the forest. If *m* is higher, there are more features to randomly subsample from each split, and this means that the resultant trees have higher probability of being decorrelated from one-another. Reducing the correlation between the trees ultimately lowers the bias since individually biased trees aren't as likely to be replicated.

Finally, the best resultant forest was selected to run for 10 trials. The results are shown in Table 4 and Figure 4:

Table 4: Training/Validation Accuracy per Trial for n = 25, depth = 2, m = 50

Trial	Accuracy_train	Accuracy_val	Average Train Accuracy	Average Val Accuracy
0	0.9593762823	0.9661538462	0.95851456709068	0.96812307692307
1	0.9665572425	0.9778461538		
2	0.9727123513	0.9821538462		
3	0.9645055396	0.9741538462		
4	0.944398851	0.9563076923		
5	0.9485022569	0.9587692308		
6	0.9768157571	0.9846153846		
7	0.9593762823	0.9661538462		
8	0.944398851	0.9563076923		
9	0.9485022569	0.9587692308		

Figure 4: Graphical Representation of the Data from Table 4



From Figure 4, the peak validation accuracy in the 10 trials was found to be 98.46% with the average as 96.81%. Randomness does affect the performance because, as seen by the fluctuations, the determined accuracy can vary depending on the selection of the initial seed. However, since the number of trees reduces the variance, increasing the number of trees should dilute the effect of selecting a different initial seed every time. As a whole, the process can be dependent on seeds through its sampling.

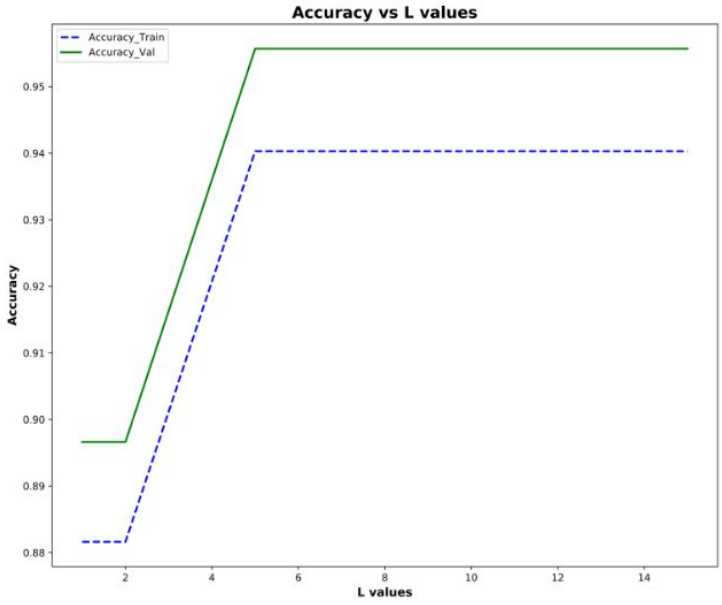
Part 3: Adaboost

Finally, the Adaboost algorithm was implemented as a way to make classifications. Using decision stumps (depth = 1) and varying values of L of [1,2,5,10,15], or the number of base classifiers, the following data was obtained which is shown in Table 5 and Figure 5:

Table 5: Numeric Training and Validation Accuracies with Depth of 1 and varying L values

L	Train accuracy	Validation accuracy
1	0.8816167419	0.8966153846
2	0.8816167419	0.8966153846
5	0.9402954452	0.9556923077
10	0.9402954452	0.9556923077
15	0.9402954452	0.9556923077

Figure 5: Training and Validation Accuracies with Depth of 1 and varying L values



Increasing the L-value has a positive effect on both training and validation accuracy. This falls in line with expectations because the purpose of Adaboost is to use several weak learners, or classifiers that perform at least as well as random guessing, in order to ultimately produce a strong classifier. Since this implementation uses decision stumps, and each subsequent decision stump learns from the error of the previous stump, it follows that increasing the number of base classifiers should result in increased accuracy for both training and validation data.

Another version of Adaboost was also ran, this time with a tree depth of 2 and an L value of 6. The training accuracy was found to be 97.56% and the validation accuracy 98.22%. This version outperformed even 15 base classifiers with a depth of 1 in the previous run. The reason for this is likely that a decision tree of depth 2 can make better predictions than a decision tree of depth 1. It is still technically a weak learner, but not as weak as a depth 1 tree. Therefore, the error on each iteration with trees of depth 2 will be less relative to any iteration with depth 1, and thus accuracy will be higher.