IA4

Yongsung Cho

Computer Science, Oregon State University

AI 534: Machine Learning

Prof. Fern

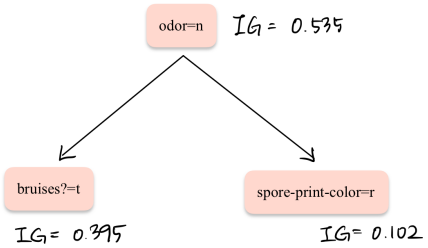
Due Dec 3, 2021

Data Preprocessing

The data was already one-hot encoding, so data preprocessing was not required. However, when AdaBoost is implemented, the class value should be -1 and 1 so that it can be predicted with alpha values. Therefore, 0 of the class features of AdaBoost's training dataset and validation dataset is replaced to -1.

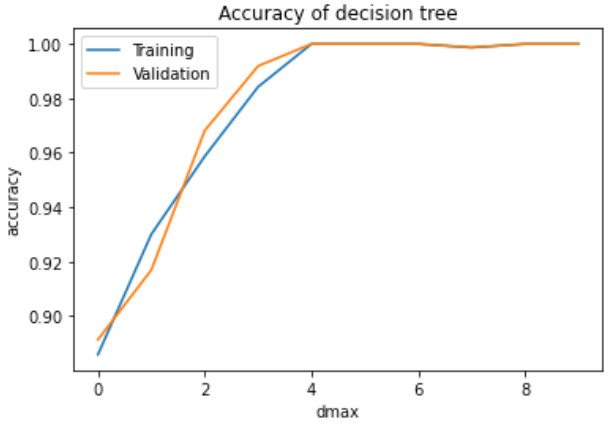
Part 1 Decision Tree

(a) What are the first three splits selected by your algorithm? This is for the root, and the two splits immediately beneath the root. What are their respective information gains?



As shown in the picture, when split from root node to odor=n feature, information gain was the largest at 0.535. When split from root node to odor=n, a dataset with odor=n value = 0 was returned to the left node, and the dataset with odor=n value = 1 was returned to the right node. The information gain value of bruises?=t was 0.395, which is the largest information gain value. Likewise, the right node was calculated, and the score-print-color=r feature was the highest, with an Information gain of 0.102.

(b) Evaluate and plot the training and validation accuracies of your trees as a function of dmax ranging from 1 to 10. At which depth does the train accuracy reaches to 100%? If your tree could not get to 100% before the depth of 10, keep on extending the tree in depth until it reaches 100% for the train accuracy. Do you observe any overfitting?

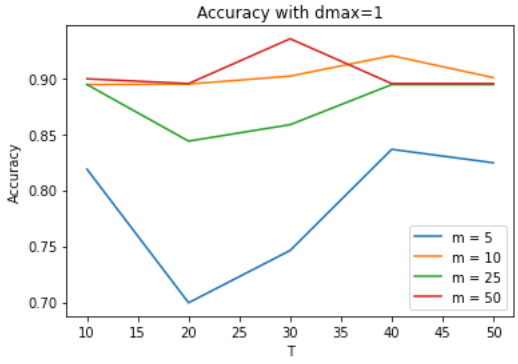


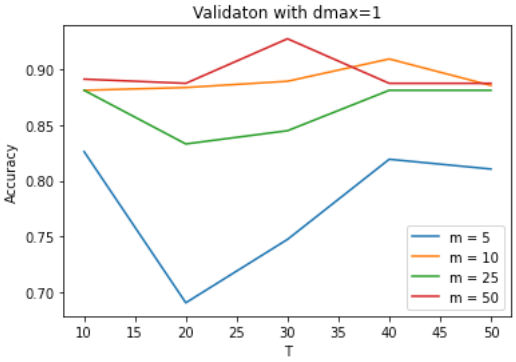
The graph above represents the accuracy of the decision tree according to the dmax value, which is the hyperparameter. When the dmax value was 5, the training accuracy reached 100%. Overfitting is suspected when dmax is more than 5 because training accuracy converges almost 100%. However, it is not sure whether the validation accuracy is overfitting because it also converges nearly 100%. Overfitting is suspected because the training accuracy is 100%, but it cannot be concluded that it is overfitting because of the limitations of the dataset.

Part 2 Random Forest

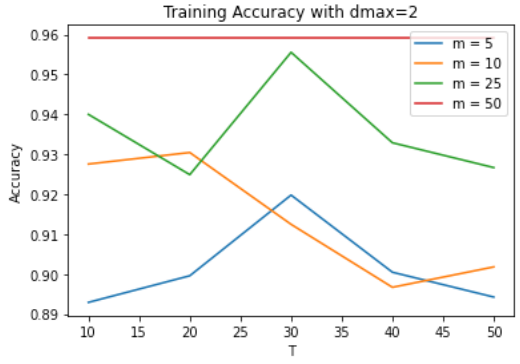
(a) For each dmax value, create two figures, one for training accuracy and one for validation accuracy. The training accuracy figure should contain four curves, each showing the train accuracy (y-axis) of your random forest with a particular m value as a function of T (x-axis). Be sure to use different colors/lines to indicate which curve corresponds to which m value, and include a clear legend to help the readability. Repeat the same process for validation accuracy. Compare your training curves with the validation curves, do you think your model is overfitting or underfitting for particular parameter combinations? And why?

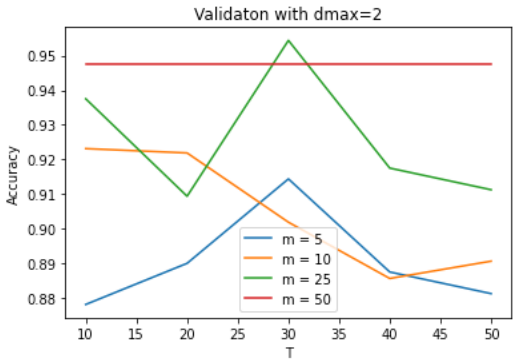
dMax = 1



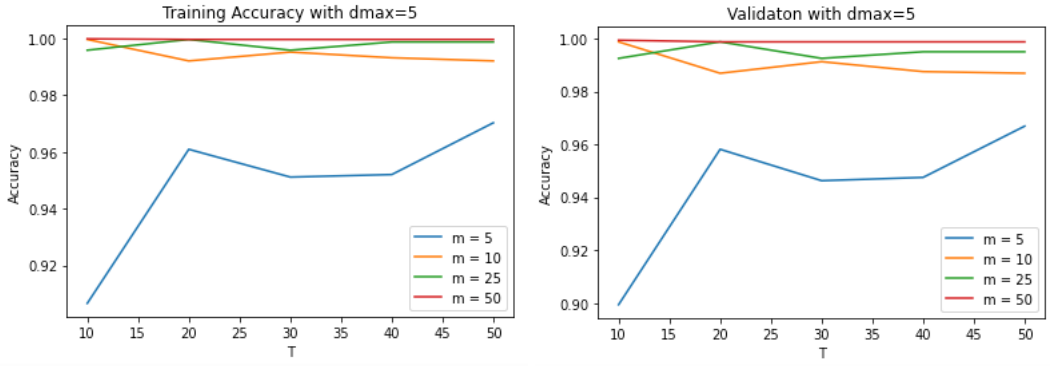


dMax = 2





dMax = 5



At dmax=2, m=5, the training accuracy showed about 96%, and the validation accuracy showed about 95% accuracy. It is overfitting because there is little change compared to other m values and shows high and constant accuracy. When graph with dmax=5 and m=10,25,50 also seems similar. This means that the higher the' m' value, the higher the probability of selecting a feature with a high information gain, and the training accuracy is close to 100%. Moreover, since the accuracy of the single decision tree was nearly 100% when dmax=5 was already achieved, the individual tree may not show a significant difference even if it is bagged. In other words, since a single decision tree is already overfitting, it tends to overfit even in a Random Forest.

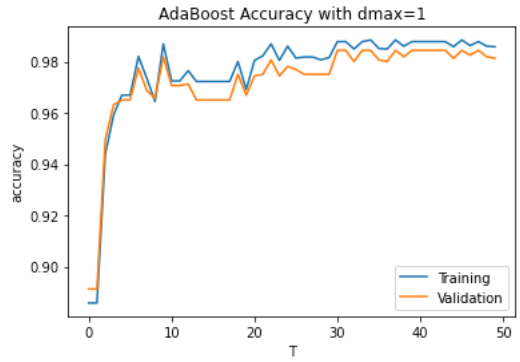
(b) For each dmax value, discuss what you believe is the dominating factor in the performance loss based on the concept of bias-variance decomposition. Can you suggest some alternative configurations of random forest that might lead to better performance for this data? Why do you believe so?

Based on the concept of bias-variance decomposition, we can reduce the variance of the expected loss through bagging. Therefore, in bagging, the bias greatly affects the expected loss. At a low m value, it is likely to select a biased feature, so it is thought that the low m value represents a lower accuracy than other graphs. Therefore, Random Forest, which reduces bias through high m value, shows good performance. I think that more various feature sets and methods should be ensembled to improve the performance of Random Forest. For example, there is a way to predict by mixing hard voting and soft voting, and you can find information gain by selecting two or more of the features and using two columns. If you divide into two columns, the decision tree may not be binary anymore.

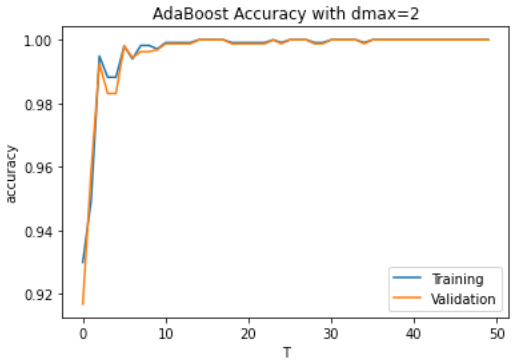
Part 3 AdaBoost(Boosting)

(a) For each dmax value, create a figure showing two curves, showing the accuracy (y-axis) on train and validation of your ensemble as a function of T (x-axis). Be sure to use different colors/lines for train and validation and include a clear legend to help the readability. Compare your training curves with the validation curves, do you think your model is overfitting or underfitting for particular parameter combinations? And why?

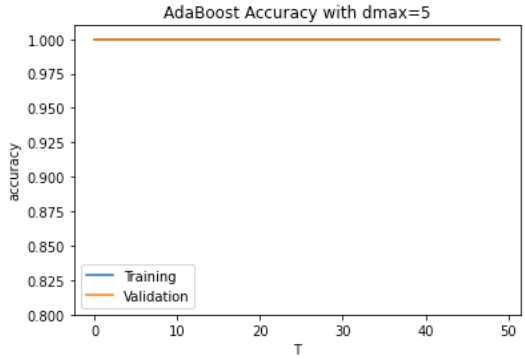
dmax = 1



dmax = 2



dmax = 5



Originally T values, ranging from 10 to 50 increase by 10, but to plot a more accurate graph, I calculated the accuracy for every iteration and expressed it by a graph. As a result, when dmax=1, I can see that the accuracy increases as the weight is updated. However, when dmax=2 and dmax=5, the accuracy reached almost 100%. Validation accuracy is also nearly 100%, so it can not be concluded that it is overfitting. However, due to the limitations of the validation dataset, it can be said that it has become overfitting. In particular, the graph with dmax=5 shows that the accuracy is close to 100% from the first iteration. This is thought to be a phenomenon that occurs because the decision tree has already been overfitting with 100% accuracy. Because of these problems, the error could not be obtained. So, the training and updating weight vector did not work well.

(b) For different dmax values, discuss what you believe is the dominating factor in the performance loss based on the concept of bias-variance decomposition. Can you suggest some alternative configurations of ensemble that might lead to better generalization performance for this data?

Based on the concept of bias-variance decomposition, boosting can reduce both bias and variance. The bias decreases as learning become because the next tree is constructed using the updated weight as the weight vector is updated. As the number of iterations increases, boosting is good because it reduces the variance in high iteration. However, the computational speed is slow because it is not calculated in parallel; it can also be overfitting or adversely affecting performance depending on Outliers and noise. To solve these problems, I suggest a combination of boosting and bagging. By bagging the Boosted Trees that are not overfitting, the advantages of the Boosting and the advantages of the Bagging are expected to make the better performance of the embedded decision tree.