Implementation Assignment 3

Team member: Yi Herng Ong, Shiyi Zhang

Short Description:

In this assignment, we learned how to implement decision tree classifier and two ensemble methods, which are random forest and adaboost, to enhance the performance of the decision tree we built. Gini index and benefit were calculated in the splitting process to determine a“best” feature to classify the data in a node at each depth. Results and plots are shown in the following and accompanied by a thorough discussion.

Part 1

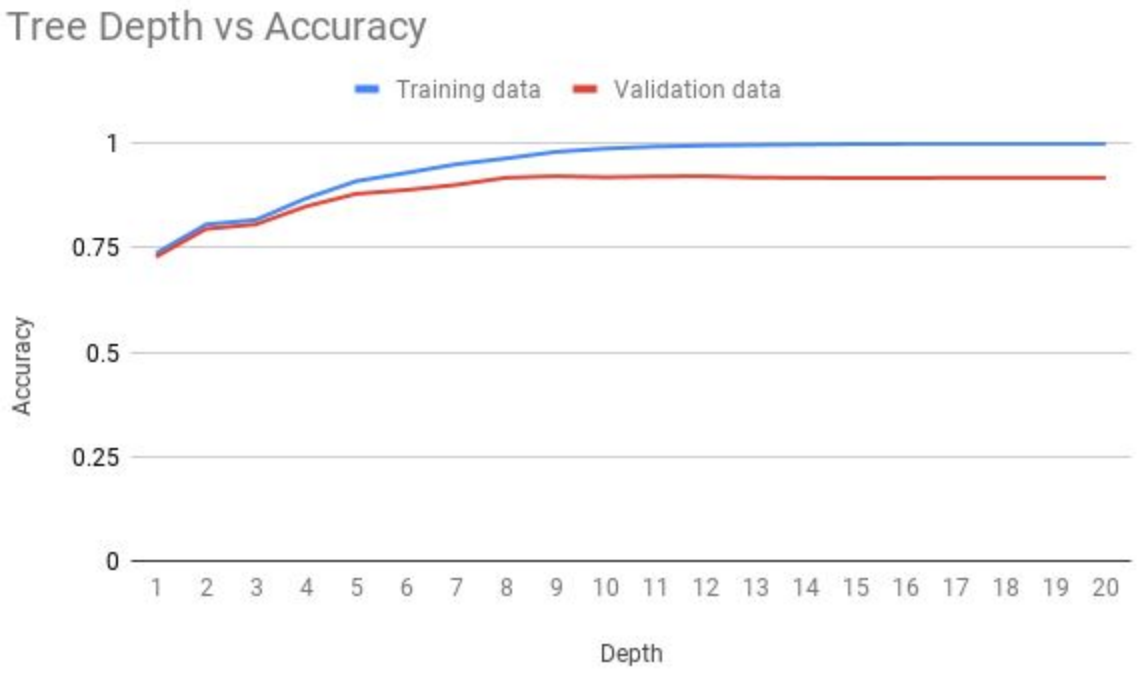

Figure 1. Depth of decision tree versus train and validation accuracy

C. Explain the behavior of train/validation performance against the depth. At which depth the train accuracy reaches to 100% accuracy? If your tree could not get to 100% before the depth of 20, keep on extending the tree in depth until it reaches 100% for the train accuracy.

Since we have access to all features in splitting process, accuracy increases when the depth of the tree increases. Training accuracy reaches 100 % at the depth of 17, while validation accuracy reaches its maximum which is about 92.2 % at the depth of 12. Validation accuracy drops slightly to 91.8 % after depth 12, perhaps this is because the decision tree is a little overfitting the training data.

D. Report the depth that gives the best validation accuracy?
Depth 12 give the best validation accuracy, which is about 92.2 %. Later it maintains at 91.8 % accuracy.

Part 2

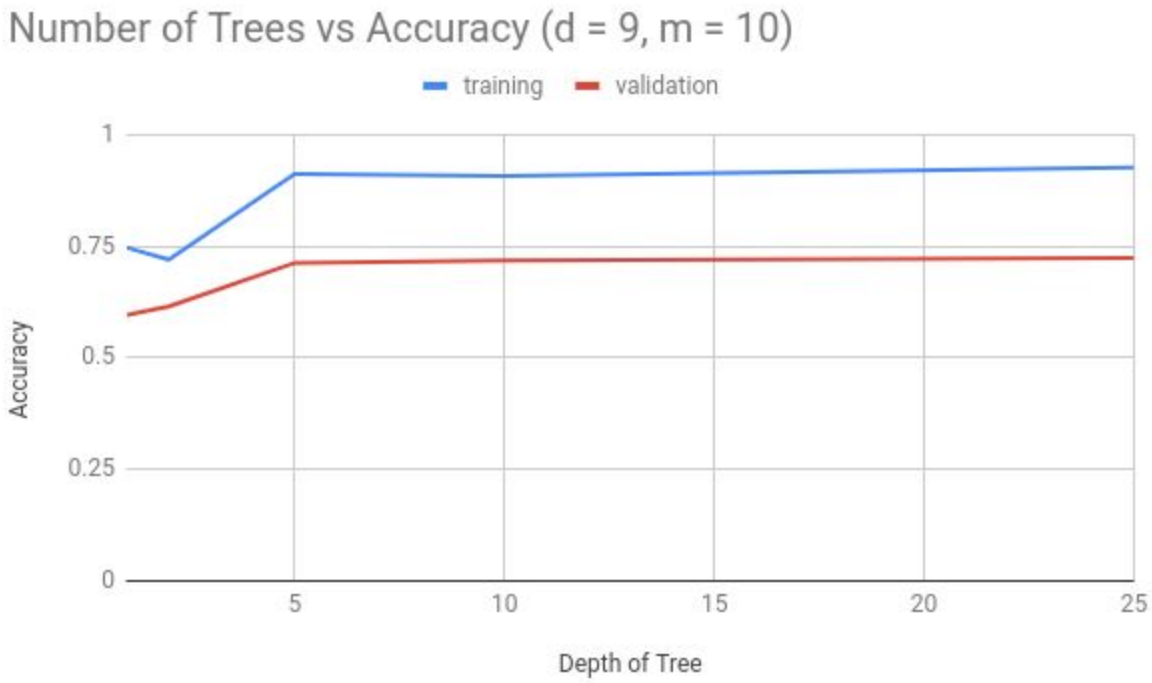

Figure 2. Number of trees in forest vs train and validation accuracy (depth = 9, number of features = 10)

C. What effect adding more tree into a forest has on the train/validation performance? Why?
Adding more tree into a forest seems to have significant effect up to 5 trees, but it only has slight improvement on performance after 5 trees. As it is shown in Figure 2, accuracy increases significantly from 72% to 91.2 % on training data, and from 61.5 % to 71.2 % on validation data. However, the accuracy becomes stagnant afterwards even though the number of trees increases to 25 trees. The reason could be the number of random features are too few, as there are only 10 features for the algorithm to pick which feature to classify data, which could cause underfitting the data even we increase the number of trees. Besides, we can see that the accuracy drops a little on training data when it comes to 2 trees in a forest, the reason is that forest with even number trees can have equal votes, and that could affect the performance of the classifier.

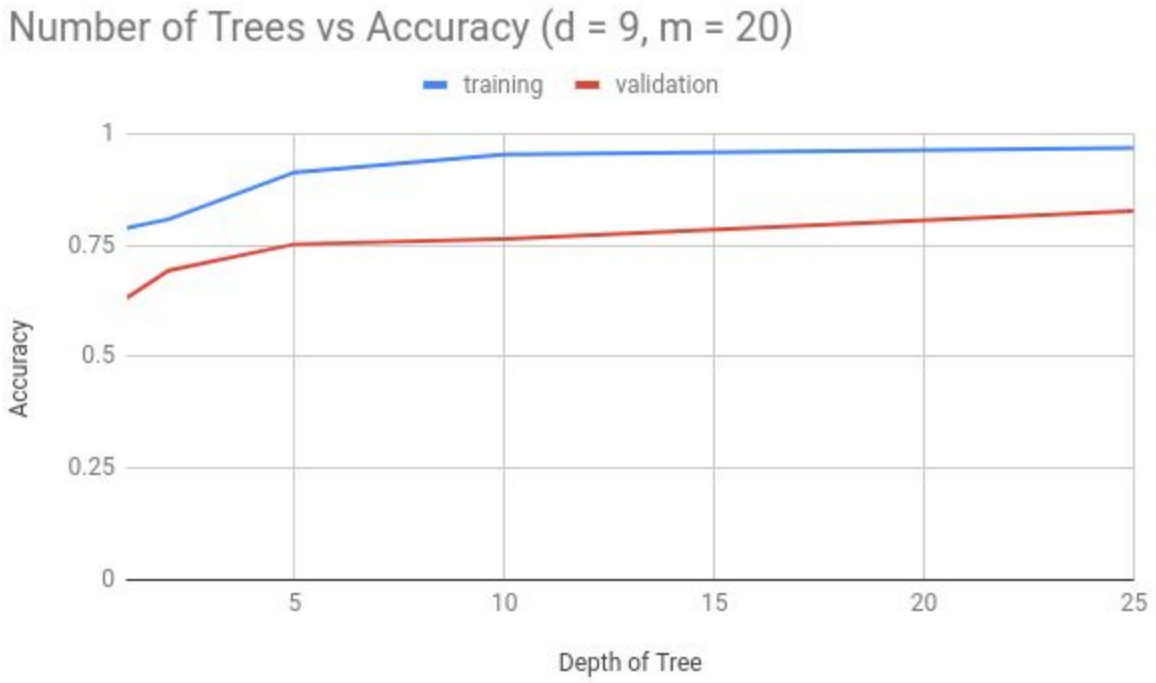

Figure 3. Number of trees in forest vs train and validation accuracy (depth = 9, number of features = 20)

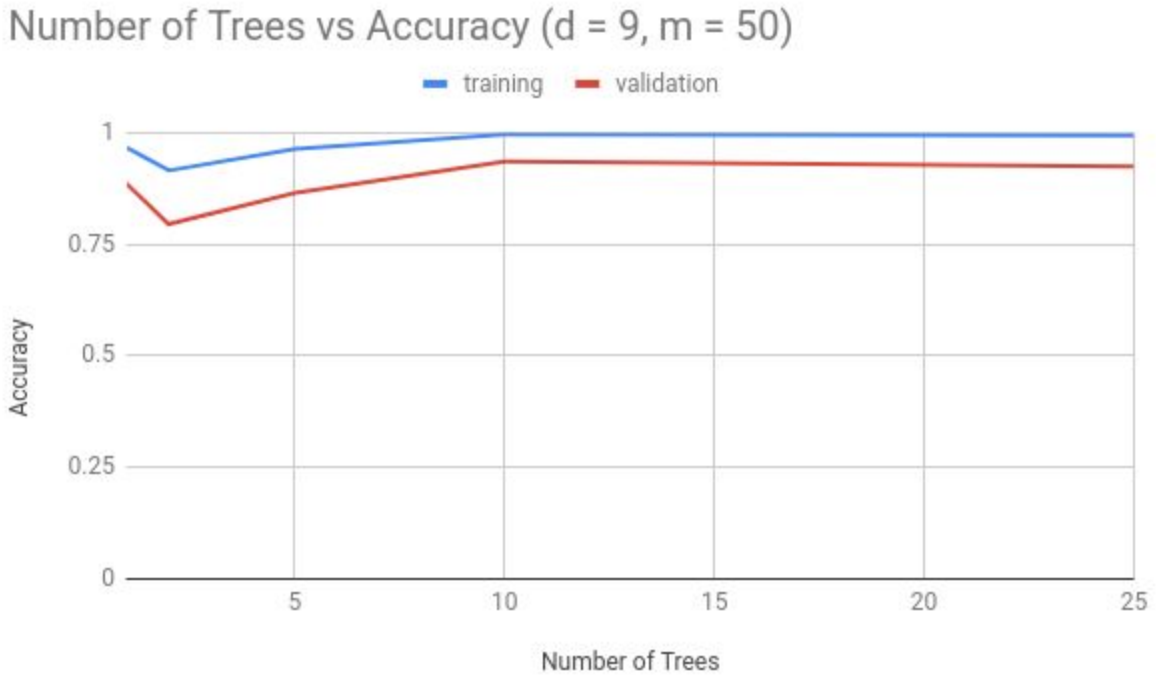

Figure 4. Number of trees in forest vs train and validation accuracy (depth = 9, number of features = 50)

D. How greater m changes the train/validation accuracy? Why?
According to Figure 3, when we double the number of features of a tree for a random forest, Training accuracy increases up to 96.7 % for training data and 82.6% for validation data in a forest with 25 trees. Furthermore, training and validation accuracy increase up to 99.3 % and 92.44 % respectively when m features increase to 50. The reason is that the algorithm now has larger number of features to split and classify the data, which minimize the risk of underfitting.

Part 3
C.

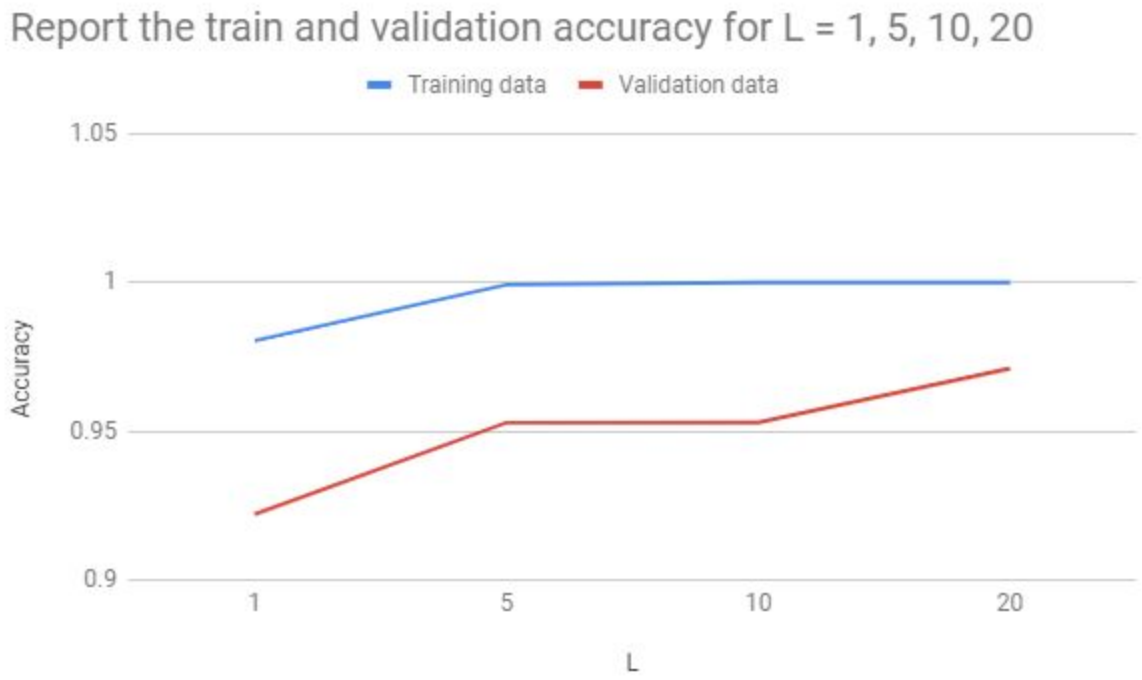

Figure 3. The Train and validation accuracy for L = 1, 5, 10, 20 applying Adaboost one decision tree with depth 9.

D. Explain the behavior of AdaBoost against the parameter L.
The training and validation accuracy are both going to decrease a little bit when starting to increase L (this part are not showing on the graph, because we just record L = 1, 5, 10, 20). The reason of this phenomenon is that when increase the weight of some wrong classified points, it may also make other correct classified points become wrong classified. So, it may decrease the accuracy when starting the Adaboost Then, both two accuracy start to increase and exceed the accuracy of original decision tree. When L is equal to 5, the training accuracy is achieving maximum 1, the validation accuracy also does not change too much between 5 to 10. However, the validation accuracy will start to increase after 10, though training accuracy keeps same. The reason of it is that boosting is often robust to overfitting. Individual tree will over fit in some parts of the data and may under fit in other parts of data. But, Adaboost, generally boosting, will do “average” of all of them instead of using individual trees. Then, the data points that the trees will

overfit will be averaged with the underfit trees. Finally, the combined average should neither or under fit and will be better.