CSCI 6963 - Reinforcement Learning
Homework 3

Christopher Okonkwo

Please use LMS to submit a zip file containing 1) your .py code, along with instructions on how to run it, 2) a .pdf file containing your answers to the above questions, 3) a .png file with your accuracy plot [including the plot in the pdf file is fine also]; 4) a .csv file with your HW2 weights so we can run your code (if you used my weights, please submit those instead).

The deadline is 11:59pm on Thursday, October 2.

Question 1

What happens if you increase the number of leaves/regions in your tree? Plot a graph with two curves: train and test accuracy. On the x-axis you have the number of leaves, and on the y-axis you have accuracy. You need to show enough points so that you illustrate when overfitting begins. For example, you can try the following 10 points for number of leaves: {100, 200, ..., 1000}.

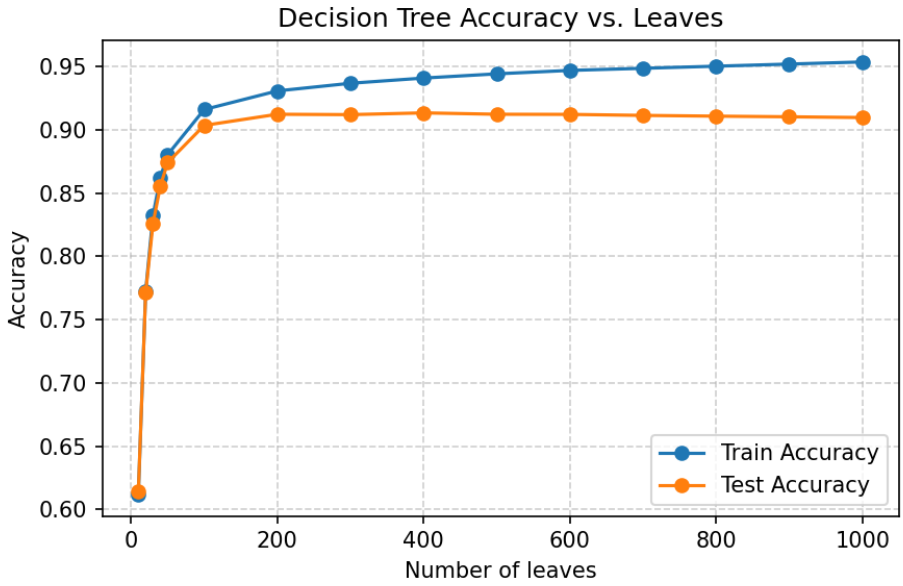

Figure 1: Accuracy vs. number of leaves for the decision tree.

Solution 1. *As we increase the number of leaves, the tree's capacity grows, and train accuracy increases from* $\approx 91.6\%$ *at* 100 *leaves to* $\approx 95.4\%$ *at* 1000. *Test accuracy improves also but peaks around* $300 - 400$ *leaves at* 91.3%, *then plateaus/gradually declines at* $\approx 91.0\%$ *by* 1000 *leaves, indicating overfitting. Overfitting begins around 300–400 leaves as the gap between train and test starts widening (variance increases). This is because as more leaves are introduced the model tries to fit too closely to the data while test stops improving and eventually dips.*

Question 2

How does your threshold step size affect accuracy and computation? What happens if you try a very big step size? How about a very small step size?

Solution 2. *Since we are using exact search, we don't need to set a step size. The algorithm sorts feature values left to right, updating counts and checks every possible threshold, so we always find the optimal split for the threshold. This guarantees accurate thresholding at a computational cost of* $O(n \log n)$ *per feature for sorting and* $O(n)$ *for scanning. Step size only matters for approximate search; with exact search, we are already getting the best thresholds.*

Leaves	**Train Time (min)**	**Test Acc. (%)**	**Train Acc. (%)**
10	0.01	61.40	61.20
20	0.01	77.14	77.25
30	0.01	82.59	83.22
40	0.01	85.54	86.24
50	0.01	87.41	88.03
100	0.02	90.35	91.60
200	0.02	91.23	93.08
300	0.02	91.20	93.68
400	0.02	91.34	94.09
500	0.02	91.23	94.42
600	0.03	91.22	94.69
700	0.03	91.14	94.86
800	0.03	91.08	95.03
900	0.03	91.03	95.20
1000	0.03	90.97	95.36

Table 1: Accuracy vs. number of leaves for the decision tree.