CS 5350/6350, DS 4350: Machine Learning Spring 2024

Homework 4

Handed out: March 15, 2020
Due date: March 29, 2020

General Instructions

- You are welcome to talk to other members of the class about the homework. I am more concerned that you understand the underlying concepts. However, you should write down your own solution. Please keep the class collaboration policy in mind.
- Feel free discuss the homework with the instructor or the TAs.
- Your written solutions should be brief and clear. You need to show your work, not just the final answer, but you do *not* need to write it in gory detail. Your assignment should be **no more than 10 pages**. Every extra page will cost a point.
- Handwritten solutions will not be accepted.
- The homework is due by midnight of the due date. Please submit the homework on Canvas. You should upload one file: a PDF report with answers to the questions below.
- Some questions are marked **For 6350 students**. Students who are registered for CS 6350 should do these questions. Of course, if you are registered for CS 5350, you are welcome to do the question too, but you will not get any credit for it.

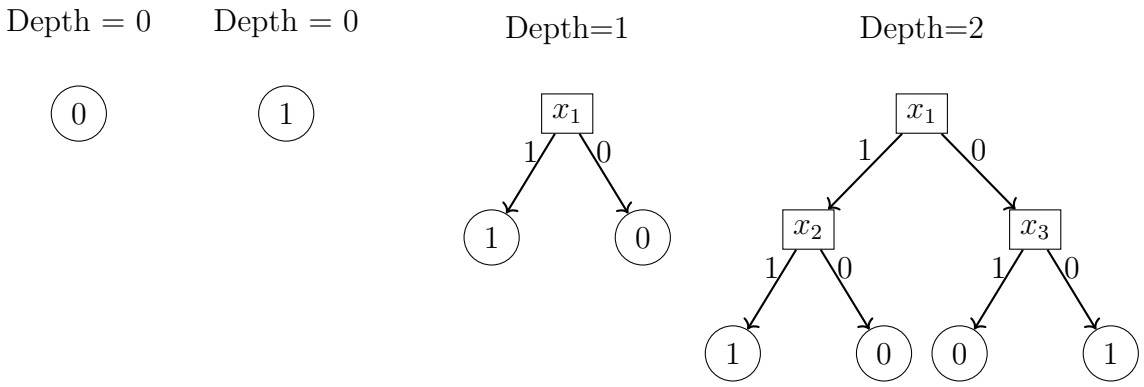
Important. Do not just put down an answer. We want an explanation. No points will be given for just the final answer without an explanation. You will be graded on your reasoning, not just on your final result.

Please follow good proof technique; what this means is if you make assumptions, state them. If what you do between one step and the next is not trivial or obvious, then state how and why you are doing what you are doing. A good rule of thumb is if you have to ask yourself whether what you are doing is obvious, then it is probably not obvious. Try to make the proof clean and easy to follow.

1 PAC Learnability of Depth Limited Decision Trees [30 points]

In this question, you will be showing that depth limited decision trees are PAC learnable.

Suppose we have a binary classification problem with n Boolean features that we seek to solve using decision trees of depth k. For this question assume trees are complete, meaning each node (other than the leaf nodes) has exactly two children. The figure below shows some examples of such trees and their depths.



1. Since decision trees represent a finite hypothesis class, the quantity of interest is the number of such trees—i.e., trees with depth k over n features. Suppose we use $S_n(k)$ to denote the number of the number of trees with depth exactly k if we have n features.

   The following questions guide you through this counting process. Recall that each answer should be accompanied with an explanation.*If you simply write the final answer, you will not get any points.* (**Please see the note at the end of this questions for further clarification**3).

   (a) [2 points] What is $S_n(0)$? That is how many trees of depth 0 exist?

   (b) [3 points] What is $S_n(1)$? That is, with n features, how many trees of depth 1 exist?

   (c) [4 points] Suppose you know the number of trees with depth i, for some i. This quantity would be $S_n(i)$ using our notation. Write down a recursive definition for $S_n(i+1)$ in terms of n and $S_n(i)$.

   For this expression, you can assume that we are allowed to the use same feature any number of times when the tree is constructed.

   (d) [6 points] Recall that the quantity of interest for PAC bounds is the log of the size of the hypothesis class. Using your answer for the previous questions, find a closed form expression representing $\log S_n(k)$ in terms of n and k. Since we are not looking for an exact expression, but just an order of magnitude, so you can write your answer in the big O notation.

2. Next, you will use your final answer from the previous question to state a sample complexity bound for decision trees of depth k.

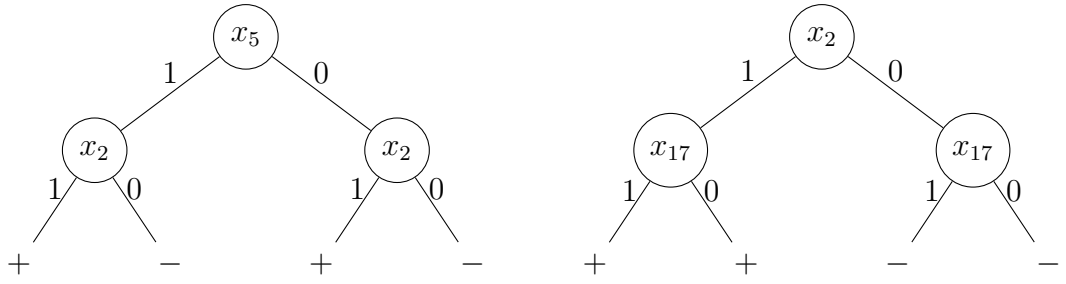
   (a) [3 points] With finite hypothesis classes, we saw two Occam's razor results. The first one was for the case of a consistent learner and the second one was for the agnostic setting. For the situation where we are given a dataset and asked to use depth-k decision trees as our hypothesis class, which of these two settings is more appropriate? Why?

   (b) [4 points] Using your answers from questions so far, write the sample complexity bound for the number of examples m needed to guarantee that with probability more than $1-\delta$, we will find a depth-k decision tree whose generalization error is no more than ϵ away from its training error.

3. [4 points] Is the set of depth-k decision trees PAC learnable? Is it efficiently PAC learnable?

4. [4 points] Suppose the number of features we have is large and the depth limit we are considering is also large (say, in the thousands or more). Will the number of examples we need be small or large? Discuss the implications of the sample complexity bound from above.

NOTE: In this question we are counting trees that are structurally different instead of functionally different. As an exercise, you can confirm that the following two trees are structurally different but equal in terms of the label they assign to any example, namely the trees are functionally equivalent.



2 Shattering [15 points, for 6350 students]

Suppose we have a set X_n consists of all binary sequences of a length n. For example, if $n = 3$, the set would consist of the eight elements {`000`, `001`, `010`, `011`, `100`, `101`, `110`, `111`}.

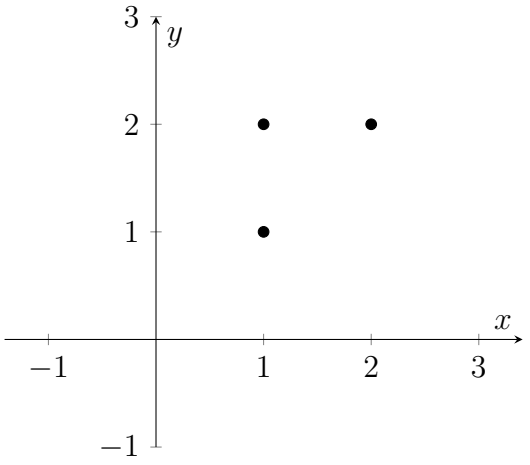
Consider a set of functions H_n that we will call the set of *templates*. Each template is a sequence of length n that is constructed using `0`, `1` or `-` and returns $+1$ for input binary sequences that match it and -1 otherwise. While checking whether a template matches an input, a `-` can match both a `0` and a `1`.

For example, the template `-10` matches the binary strings `010` and `110`, while `-1-` matches all strings that have a `1` in the middle position, namely `010`, `011`, `110` and `111`.

Does the set of templates H_n shatter the set X_n? Prove your answer.

3 VC Dimension [45 points]

1. [5 points] Assume that the three points below can be labeled in any way. Show with pictures how they can be shattered by a linear classifier. Use filled dots to represent positive classes and unfilled dots to represent negative classes.



2. **VC-dimension of axis aligned rectangles in $\mathbb{R}^d$**: Let H^d_{rec} be the class of axis-aligned rectangles in $\mathbb{R}^d$. When $d = 2$, this class simply consists of rectangles on the plane, and labels all points strictly outside the rectangle as negative and all points on or inside the rectangle as positive. In higher dimensions, this generalizes to d-dimensional boxes, with points outside the box labeled negative.

   (a) [10 points] Show that the VC dimension of H^2_{rec} is 4.

   (b) [10 points] Generalize your argument from the previous proof to show that for d dimensions, the VC dimension of H^d_{rec} is $2d$.

3. In the lectures, we considered the VC dimensions of infinite concept classes. However, the same argument can be applied to finite concept classes too. In this question, we will explore this setting.

   (a) [10 points] Show that for a finite hypothesis class $\mathcal{C}$, its VC dimension can be at most $\log_2(|\mathcal{C}|)$. (Hint: You can use contradiction for this proof. But not necessarily!)

   (b) [5 points] Find an example of a class $\mathcal{C}$ of functions over the real interval $X = [0, 1]$ such that $\mathcal{C}$ is an **infinite** set, while its VC dimension is exactly one.

   (c) [5 points] Give an example of a **finite** class $\mathcal{C}$ of functions over the same domain $X = [0, 1]$ whose VC dimension is exactly $\log_2(|\mathcal{C}|)$.

4 Extra Credit - Decision Lists [25 points]

In this problem, we are going to learn the class of k-decision lists. A decision list is an ordered sequence of if-then-else statements. The sequence of if-then-else conditions are tested in order, and the answer associated to the first satisfied condition is output. See Figure 1 for an example of a 2-decision list.

A *k-decision list* over the variables $x_1, \ldots, x_n$ is an ordered sequence $L = (c_1, b_1), \ldots, (c_l, b_l)$ and a bit b, in which each c_i is a conjunction of at most k literals over $x_1, \ldots, x_n$. The bit b is

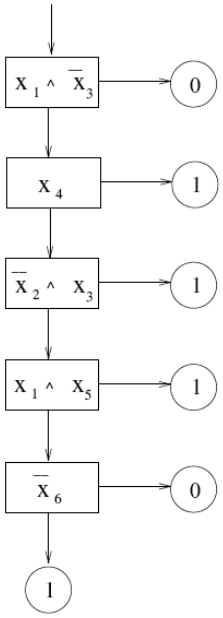


Figure 1: A 2-decision list.

called the *default* value, and b_i is referred to as the bit *associated* with condition c_i. For any input $x \in \{0,1\}^n$, $L(x)$ is defined to be the bit b_j, where j is the smallest index satisfying $c_j(x) = 1$; if no such index exists, then $L(x) = b$.

We denote by k-DL the class of concepts that can be represented by a k-decision list.

1. [8 points] Show that if a concept c can be represented as a k-decision list so can its complement, $\neg c$. You can show this by providing a k-decision list that represents $\neg c$, given $c = \{(c_1, b_1), \ldots, (c_l, b_l), b)$.

2. [9 points] Use Occam's Razor to show:
   For any constant $k \geq 1$, the class of k-decision lists is PAC-learnable.

3. [8 points] Show that 1-decision lists are a linearly separable functions. (Hint: Find a weight vector that will make the same predictions a given 1-decision list.)