Homework 4

Due: Thursday, February 22, 2024 at 12:00pm **(Noon)**

Written Assignment

Problem 1: Bias-Complexity Tradeoff

(10 points)

a. Explain why the No Free Lunch Theorem requires us to introduce bias into our machine learning models.

b. While training a machine learning model, Steve notices that the model's error remains large even after many training iterations. Steve determines that this error results from overfitting, and he proposes changing the complexity of the hypothesis class to reduce the error. Will changing the complexity of the hypothesis class successfully reduce the error? If so, how should the complexity be changed? If not, why not? Explain your reasoning.

c. After tinkering with his model, Steve eventually manages to prevent his model from overfitting. However, Steve notices that his model's error remains large because his model is now underfitting the data. To decrease underfitting, Steve decides to collect more training data. Will collecting more training data successfully reduce the error? Explain your reasoning. *Hint: Consider if collecting more training data affects approximation and/or estimation error, and if so, explain how.*

d. Tuning hyperparameters to strike a balance between underfitting and overfitting is a critical part of the machine learning process. Going too far in either direction can often lead to unintended consequences. What are some ways that overfitting or underfitting could produce unfairness in a learned model?

Problem 2: Assumptions & Guarantees

(10 points)

a. In lecture, we made several assumptions before proving that an ERM solution is PAC learnable. What are these assumptions?

b. The HTAs want to produce a model to predict the final grade of a student based off of their Homework 6 score. During the sampling process, the HTAs select the next example based on how similar it was to the previous example. Why does the ERM solution not perform well at test time? Explain in terms of the assumptions above.

c. Steve wants to produce a model to predict the grams of sugar in a cake based on its characteristics (grams of flour, calories and number of eggs). He trains his model on a dataset of **muffins** and then uses it to make predictions about **cakes**. Why does the ERM solution not perform well at test time?

Problem 3: PAC Learning of Partial Orderings

(20 points)

Background Information

A *partial ordering* (denoted $\preceq$) is a binary relation over a set X that satisfies the following properties:

- *Reflexivity*: $\forall s \in X, \quad s \preceq s$
- *Antisymmetry*: $\forall s, t \in X,$ if $s \preceq t, t \preceq s$, then $s = t$
- *Transitivity*: $\forall r, s, t \in X,$ if $r \preceq s, s \preceq t$, then $r \preceq t$

A *total ordering* is a partial ordering with the additional guarantee that all elements within X can be compared in the relation.

Example: The "$\leq$" relation is a total ordering on the real numbers. Any two numbers a, b may be compared such that either $a \leq b$ or $b \leq a$. Additionally, the "$\leq$" relation satisfies the conditions of reflexivity, antisymmetry and transitivity stated above.

Example: The "$\subseteq$" relation is a partial but not total ordering. "$\subseteq$" satisfies reflexivity, transitivity and antisymmetry. However, given two sets A and B, there is no guarantee that either $A \subseteq B$ or $B \subseteq$ A.

Problem

Now consider the following:

- Let X be a set of n unique elements: $\{1, 2, 3, ..., n\}$.
- **Input Space:** The set S of all possible permutations of length n, composed the elements of X.
  For example, if X is a set of length $n = 5$, a permutation could look like $(3, 4, 2, 5, 1)$, and S would be the set of all different possible 5-element permutations.
- **Hypothesis Space:** The set of all hypotheses, where each hypothesis maps all n-element permutations to either 0 or 1. Each hypothesis is equivalent to a partial ordering. So, a n-element permutation would be mapped to 1 if it abides by the partial ordering/hypothesis and 0 otherwise.
  For example, if a hypothesis h is $1 < 3$, and we have the permutation $(3, 1)$, the permutation does not follow the hypothesis since 3 is before 1, reading left to right. Therefore, h would map $(3, 1)$ to 0.

Suppose that there is a hidden partial ordering that we do not have access to. We do have a sample of permutations, and a label for each sample indicating whether the permutation is consistent with the true hidden partial ordering. Assume that the sample has both postively and negatively labeled permutations. We are trying to learn the rule from this sample.

Example:

Suppose $n = 3$, the set $X = \{1, 2, 3\}$, and we have a hidden partial ordering $1 \preceq 3$.
Then our input space is the set of all possible 3-element permutations consisting of elements from $\{1, 2, 3\}$, and our hypothesis space is the set of all hypotheses mapping each permutation to 0 or 1.
We have access to a sample of permutations that are labeled. For example, let $(2, 3, 1)$ be a permutation that is available to us. It has a label of 0 since it violates the hidden partial ordering - 3 precedes 1 in this permutation, which does not match the hidden ordering. However, since we don't have access to the ordering, we have to use this permutation and the rest of sample to learn this true ordering.

a. Show that $|H|$ is between $n!$ and $3^{(n^2)}$. That is, find the number of possible valid partial orders generated from n elements, and show that it is bounded by $n!$ and $3^{(n^2)}$. *Hint: For the upper bound, it may be helpful to consider that $h \in H$ may be represented as a Directed Acyclic Graph (DAG). The nodes of the DAG can be thought of as the elements that make up the n-element permutations of the input space. Think about how the relations between nodes in a DAG are similar to the relations between elements within a partial ordering. This question may also therefore be approached by thinking about the number of valid DAGs with n nodes.*

b. Show that the amount of data needed is polynomial in n, in order to ensure the ERM solution is PAC. *Hint: Consider the bounds for $|H|$ we proved in part (a).*

c. Describe an algorithm you could use to efficiently find an ERM solution (the true partial ordering, or DAG) for a given sample of observations. Be sure to explain the steps of your algorithm, prove the correctness of the algorithm and analyze its runtime. Note, we do not have a strict runtime requirement, but the algorithm should be in polynomial time. The algorithm itself can be written in pseudocode or in words. *Hint: The true hypothesis is within our hypothesis space. Consider the direction of the ordering between any two elements in every permutation with label 1.*

Grading Breakdown

The grading breakdown for the assignment is as follows:

Problem 1	25%
Problem 2	25%
Problem 3	50%
Total	100%

Handing in

Your written assignment should be uploaded to gradescope under "Homework 4". If you have questions on how to set up or use Gradescope, ask on Edstem! For this assignment, you should have written answers for Problems 1, 2, and 3.

Anonymous Grading

You need to be graded anonymously, so do not write your name anywhere on your handin.

Obligatory Note on Academic Integrity

Plagiarism — don't do it.

As outlined in the Brown Academic Code, attempting to pass off another's work as your own can result in failing the assignment, failing this course, or even dismissal or expulsion from Brown. More than that, you will be missing out on the goal of your education, which is the cultivation of your own mind, thoughts, and abilities. Please review this course's collaboration policy and if you have any questions, please contact a member of the course staff.