Midterm 2

Printed Name:

Due date:

1. The exam is due Sunday 6 Nov at midnight.
2. You may submit it either on sakai electronically or by putting a physical copy under my door.

Rules:

1. The exam is untimed. You do not have to complete the exam in a single sitting. You may pause and restart whenever you'd like.
2. You may use any non-human resources that you like, including notes, books, internet articles, and computers.
3. You are not allowed to discuss the exam in any way with any human until after the due date. This includes:
   (a) obviously bad behavior like copying answers,
   (b) more banal behavior such as:
      i. telling your friend "Problem 6 was really hard" or
      ii. asking your friend "Have you completed the exam yet?"

   Even after you finish the exam, you may not discuss it.
4. If you do have questions about the exam, you should email me the questions rather than posting to github.

Grading:

1. For the True/False/Open questions: Each correct answer will be awarded +1 point, each incorrect answer will result in a -1 point penalty, and each blank answer will result in 0 points.
2. All other problems are worth 1 point, with no penalty for incorrect answers.
3. There are 16 points possible on the exam. Your final grade entered into sakai will be
$$\min\{15, \text{the number of points earned}\}.$$
4. If you find a substantive error on the exam, then I will award you +1 bonus point.

Good luck :)

Problem 1. For each statement below, circle `True` if the statement is known to be true, `False` if the statement is known to be false, and `Open` if the statement is not known to be either true or false. Ensure that you pay careful attention to the formal definitions of asymptotic notation in your responses.

1. `True` `False` [✓ `Open`] For learning problems that are not linearly separable, the fastest possible algorithm for minimizing the 0-1 loss runs in exponential time in the number of feature dimensions d.

2. `True` [✓ `False`] `Open` Let $\mathcal{H}_1$ and $\mathcal{H}_2$ be two hypothesis classes satisfying $d_{\text{VC}}(\mathcal{H}_1) \leq d_{\text{VC}}(\mathcal{H}_2)$. Then $H_1 \subset H_2$.

3. [✓ `True`] `False` `Open` Let $\mathcal{H}$ be the perceptron hypothesis class with the number of feature dimensions $d = 8$. Then $m_{\mathcal{H}}(6) = 64$.

4. [✓ `True`] `False` `Open` Let $\mathcal{H}$ be the perceptron hypothesis class and $\mathcal{H}_\Phi$ be the perceptron hypothesis class with the PCA kernel applied. Furthermore let $g \in \mathcal{H}$ and $g_\Phi \in \mathcal{H}_\Phi$ be the empirical risk minimizers. Then $E_{\text{in}}(g) \leq E_{\text{in}}(g_\Phi)$.

5. `True` [✓ `False`] `Open` Let $\mathcal{H}$ be the perceptron hypothesis class and let $\mathcal{H}_\Phi$ be the perceptron hypothesis class with the decision stump feature map. Let $g \in \mathcal{H}$ and $g_\Phi \in \mathcal{H}_\Phi$ be the emperical risk minimizers. Then VC theory provides a better generalization bound for g than for g_Φ.

6. `True` [✓ `False`] `Open` Define the approximation error of a hypothesis class $\mathcal{H}$ to be

$$E_{\text{app}} = \min_{h \in \mathcal{H}} E_{\text{out}}(h). \tag{1}$$

Then applying the random feature embedding with a low output degree will decrease the approximation error.

7. [✓ `True`] `False` `Open` Let

$$\mathcal{H}_{\text{axis2}} = \left\{ \mathbf{x} \mapsto \sigma \operatorname{sign}(x_i) : \sigma \in \{+1, -1\}, i \in [d] \right\},$$

and

$$\mathcal{H}_{\text{axis}} = \left\{ \mathbf{x} \mapsto \operatorname{sign}(x_i) : i \in [d] \right\},$$

Let $g_{\text{axis2}} \in \mathcal{H}_{\text{axis2}}$ and $g_{\text{axis}} \in \mathcal{H}_{\text{axis}}$ be the outputs of the TEA algorithm on their respective hypothesis classes. Then $E_{\text{in}}(g_{\text{axis2}}) \leq E_{\text{in}}(g_{\text{axis}})$.

8. `True` [✓ `False`] `Open` Let $\mathcal{H}_1$ be the perceptron hypothesis class with the decision stump feature map applied, and $\mathcal{H}_2$ be the perceptron hypothesis class with the polynomial kernel of degree 3. Furthermore, let $g_1 \in \mathcal{H}_1$ and $g_2 \in \mathcal{H}_2$ be the emperical risk minimizers. Then VC theory predicts that $|E_{\text{out}}(g_1) - E_{\text{test}}(g_1)| \leq |E_{\text{out}}(g_2) - E_{\text{test}}(g_2)|$ with high probability.

9. [✓ True] False Open Define the hypothesis class of positive and negative intervals in 1 dimension to be

$$\mathcal{H} = \left\{ x \mapsto \sigma \left[\!\left[a \leq x \leq b \right]\!\right] : a \in \mathbb{R}, b \in \mathbb{R}, \sigma \in \{+1, -1\} \right\}. \tag{2}$$

Then the $d_{\mathrm{VC}}(\mathcal{H}) = 3$.

10. True [✓ False] Open There does not exist a break point for the perceptron hypothesis class.

Problem 2. Either prove or give a counterexample to the following claim: Let f be the true label function. Then it must be the case that $E_{\text{in}}(f) = 0$.

Solution: False.

Any data distribution with noise will have $E_{\text{in}}(f) > 0$, and $E_{\text{in}}(f)$ is a measure of that noise.

Common mistakes:

1. (-1) If you didn't talk about noise/randomness/probability distributions/etc, you couldn't get credit for this problem.
2. (-1) If you stated that $E_{\text{in}}(f) = 0$ would imply overfitting.

Problem 3. Either prove or give a counterexample to the following claim: Every surrogate loss function is convex.

Solution: False. The truncated hinge loss is an example of a non-convex surrogate loss.

Problem 4. What is the VC dimension of the following hypothesis class?

$$\mathcal{H} = \left\{ \mathbf{x} \mapsto \sigma [\![\|\mathbf{x}\|_2 \leq \alpha]\!] : \sigma \in \{+1, -1\}, \alpha \in \mathbb{R} \right\} \tag{3}$$

Solution: $d_{\text{VC}}(\mathcal{H}) = 2$.

Select any dataset with two data points of different norms, and $\mathcal{H}$ will shatter this dataset. This shows that $d_{\text{VC}} \geq 2$.

There is no dataset of size 3 that can be shattered by $\mathcal{H}$, so $d_{\text{VC}} < 3$.

Problem 5. You are a bank using logistic regression to learn a formula for whether or not to issue a loan. Your dataset has N data points and d features, and you have trained a model g using second order gradient descent so that your optimization error is negligible. You evaluate your model on the training and test sets and get values of $E_{\text{in}}(g) = 0.05$ and $E_{\text{test}}(g) = 0.41$.

Your boss suggests that augmenting the dataset with more features might improve performance. Is this a good idea? Use VC to justify why.

Solution: No.

Although we can't directly measure the generalization error $|E_{\text{in}} - E_{\text{out}}|$, we can estimate it with $E_{\text{in}} - E_{\text{test}}$ because $E_{\text{test}} \approx E_{\text{out}}$. This estimated generalization error is large relative to the in-sample error E_{in}. Therefore, reducing the VC dimension will likely improve performance.

Adding more feature dimensions, however, will result in the VC-dimension increasing, and the generalization error increasing. Since the in-sample error is already so small, it cannot decrease much as a result of these additional dimensions. And so the increase in generalization error is unlikely to be counterbalanced by a decrease in in-sample error.

Common mistakes:

1. (-1) Not discussing VC theory/generalization error at all.
2. (-0.5): Stating that changing d will have no effect on E_{test} is incorrect. It is true that changing d will not have an effect on $E_{\text{test}} - E_{\text{out}}$, since this is only determined by the size of the test set.
3. (-0.5): If you stated that adding dimension increases generalization error, but did not state why that is bad for this problem. Several students implied that increasing generalization error is always bad for every problem, and this is incorrect.
4. (no points deducted) Many students (especially non-CS students) assumed that the word "performance" referred to runtime. In general, the "performance" of an algorithm refers to the quality of the solution foremost and to runtime only secondarily. For these machine learning algorithms in particular, "performance" refers to the statistical guarantees that they offer about the quality of E_{out}. The generalization error $|E_{\text{out}} - E_{\text{in}}|$ is closely related to E_{out}, but it is not the ultimate measure of performance for an algorithm.

   One way to think about this is that we have already run the algorithm to completion above. So why should we care about the runtime of something that's already complete?

Problem 6. You are training a logistic regression model with $N = 10^{12}$ and $d = 10^6$. Which optimization algorithm do you choose and why?

Solution: SGD is the best option.

Because the number of data points and feature dimensions is large, you are likely to be computationally limited rather than statistically limited. In this situation, we want the optimization algorithm that has the fastest time to excess error ϵ rather than the algorithm with the fastest time to accuracy ρ. SGD has the fastest time to excess error ϵ.

You could get full credit for choosing 2SGD if you stated that: You are not computationally limited on this problem, and so are trying to minimize the optimization error ρ, and that 2SGD is the fastest time to accuracy ρ.

Common mistakes:

1. (-1) No mention of the tradeoff between being computationally limited or not or discussion about the difference between the time to excess error ϵ and the time to accuracy ρ.
2. (-1) Lots of people stated that the runtime per iteration of SGD does not depend on N and used this to justify using SGD for large N. It is true that the time per iteration is independent of N, but it's not a statement that we care about since it neglects the number of iterations.
3. (-1) Discussing feature maps instead of optimization algorithms.
4. For students who had already missed many points, I graded slightly more generously.

Problem 7. You work at a car manufacturer and are developing a model to determine whether a part is defective or not. You are required by regulators to use support vector machines optimized with 2nd order gradient descent, but you are free to select any feature maps that you would like. Your training data has many features ($d = 10^6$) but only a small number of data points ($N = 10^3$). Which feature maps does VC theory predict would be a good choice?

Solution: The VC dimension is much larger than the number of training data points, and so the feature map needs to reduce the dimensionality. The random feature embeddings, PCA, and decision stumps all achieve this goal, and any one of those answers received full credit.

Common mistakes:

1. (-0.5) If you did not state why we want to decrease generalization error in this specific case.