COMS 4771 (SPRING 2021) EXAM 2 APRIL 16, 2021

Name & UNI: ____

Instructions

- You have a 24 hour period (from Friday April 16, 2021 00:01 AM EDT to Friday April 16, 2021 11:59 PM EDT) to take this exam. You may open the exam at any time in this 24 hour period, but **as soon as** you open the exam you will only have two hours and a half hours to complete it and submit your responses on Courseworks. In particular, you only have 2 hours to complete the exam itself. You have an extra 30 minutes to print the exam (if you want) at the beginning, and then scan and submit your responses at the end. Do NOT use the extra 30 minutes as additional time to complete the exam, as this time is very critical for you to print/scan/upload your work.

- You may either print this exam and write your answers in the space provided or you may write your answers on separate blank pages. In the latter case, use a different page for each of the problems and clearly label which part of which problem you are answering. In either case you must scan and upload your solutions to Courseworks within 2.5 hours of the time you open this exam. Since you are given 30 minutes to resolve any "technical issues", **we will NOT accept any work once the 2.5 hour window closes**. There are **absolutely no exceptions** to this rule.

- This is an open book exam. Hence, you may use any resource of your choice to complete the exam (books, lecture slides, lecture videos, homework assignments and solutions, etc.). However, you may **not** receive help from any other person (whether student or not, whether in person or virtually).

- Do not discuss this exam with **anyone** whether in person or virtually until the end of the 24 hour exam period (11:59 PM EDT on Friday). In particular do NOT post any questions about the exam content on Piazza.

- Even after the 24 hour period and after this class ends do not share this exam. In particular it would be a breach of academic honesty to post this exam online or share it with a future COMS 4771 student.

- Show all your work! Right answers without supporting work will not be given full credit and wrong answers with supporting work may receive partial credit.

- You may use/cite any result from the lectures or the homeworks without proof.

- Suggestion: Do not spend too much time on any one question. In particular, I **strongly** suggest that you do **not** try to Google answers that you do not know. Every question is new (and so will not be found on the Internet) and trying to learn a concept that you do not understand would not be a wise use of your very limited time.

A signed version of the honor pledge below **must** appear in your submission. If you choose not to print the exam then you must copy and sign the honor pledge onto the first blank sheet of paper that you use to write your answers.

Honor pledge: I have not given or received unauthorized assistance on this examination. I will not retain or re-distribute any handwritten or electronic copies of this examination, either in part or in full.

Sign here: ______________________________ Date: ______________

1. **(20 points)** For each statement below state either True or False. Justify your answer by providing a short explanation/proof sketch (for true statements) or a counter example (for false statements).

   (a) ____________ Any non-convex optimization problem is NP-hard to solve. In other words, there are no non-convex optimization problems for which we have algorithms to find the exact optimal solution in polynomial time.

   (b) ____________ SVM with slack (with a fixed hyperparameter C) always returns the linear classifier with the smallest training error.

   (c) ____________ All decision trees with a single non-leaf node (i.e. all decision trees that make a single split) are linear classifiers.

   (d) ____________ Any function class with finite VC dimension is efficiently PAC learnable.

(e) ____________ A function class with infinite VC dimension is never efficiently PAC learnable, but it may still be (non-efficiently) PAC learnable.

(f) ____________ The coefficients of the weight vector returned by Ridge regression will tend to be larger (either more positive or more negative) than those of the weight vector returned by OLS.

(g) ____________ Suppose we train a multivariate Gaussian probabilistic classifier on binary labeled data and also train a Gaussian Mixture Model with $k = 2$ on the same training data (the labels are ignored when training the Gaussian Mixture Model). The two models will always have the same boundary.

(h) ____________ Given a dataset $S = \{(x_1, y_1), (x_2, y_2), ..., (x_n, y_n)\}$ with $x_i \in \mathbb{R}^D$ and $y_i \in \{0, 1\}$ suppose we use **PCA** (applied only on the x_i, the y_i are ignored) to find a transformation $\phi : \mathbb{R}^D \to \mathbb{R}^d$ with $d < D$. Let $S' = \{(\phi(x_1), y_1), (\phi(x_2), y_2), ..., (\phi(x_n), y_n)\}$. Then the minimum training error of D-dimensional linear classifiers on S is **always** less than or equal to the minimum training error of d-dimensional linear classifiers on S'.

(i) ____________ Given a dataset $S = \{(x_1, y_1), (x_2, y_2), ..., (x_n, y_n)\}$ with $x_i \in \mathbb{R}^D$ and $y_i \in \{0, 1\}$ suppose we use **a non-linear dimensionality reduction technique** (again the y_i are ignored) to find a transformation $\phi : \mathbb{R}^D \to \mathbb{R}^d$ with $d < D$. Let $S' = \{(\phi(x_1), y_1), (\phi(x_2), y_2), ..., (\phi(x_n), y_n)\}$. Then the minimum training error of D-dimensional linear classifiers on S is **always** less than or equal to the minimum training error of d-dimensional linear classifiers on S'.

(j) ____________ The partition induced by the Lloyd's method for k-means optimization always results in convex cells. That is, let $c_1, \dots, c_k \in \mathbb{R}^d$ be the solution returned by the algorithm on a given dataset. Define

$$S_j := \Big\{ x \in \mathbb{R}^d \;\Big|\; \arg\min_i \|x - c_i\|^2 = j \Big\} \qquad \text{for } j = 1, \dots, k.$$

Then each S_j is necessarily a convex set.

2. **[Facilities location via clustering]** You are hired as the lead data scientist in the city planning office. As your first important project your boss tells you that they have received funding to build k hospitals throughout the city. The city has identified $m > k$ different potential sites $\{s_1, \ldots, s_m\}$ to build these hospitals. The goal obviously is to pick k sites that collectively minimize the worst-case commute distance to the closest hospital.

   More formally, you are given n households $X = \{x_1, \ldots, x_n\} \subset \mathbb{R}^2$, and m potential sites $S = \{s_1, \ldots, s_m\} \subset \mathbb{R}^2$, and a number k. You goal is to select k sites (let's collectively call them centers $\{c_1, \ldots, c_k\} = C \subset S$), that minimizes the largest (worst-case) Euclidean distance between a household x_i and its closest center c_j. In other words,

$$\underset{C \subset S \text{ s.t. } |C|=k}{\arg\min} \left[\max_{x_i \in X} \quad \min_{c_j \in C} \quad \|x_i - c_j\|^2 \right] \tag{1}$$

   (a) **(5 points)** Your boss identifies it as a clustering problem (finding k hospital centers to "group" and serve n households), and proposes that any reasonable k-means algorithm should be able to give a good solution to this problem. Show that even when $k = 1$ and there is no restriction on the cite locations (that is $S = \mathbb{R}^2$), the optimal 1-means solution ($k = 1$) is not necessarily the optimal solution for the objective function stated above (Eq. 1).

(b) **(12 points)** Having demonstrated to your boss (in part (a)) that this optimization problem is different from the classic k-means problem, you are tasked with coming up with a methodology to find the centers.

Given X, S and k, give an algorithm that returns the set of k centers C that *exactly* minimizes the given objective (1).

(While your algorithm need not be 'efficient', more points will be awarded to an algorithm that runs faster while still returning a correct solution.)

(c) **(3 points)** What is the run time (in terms of n, m and k) of your stated algorithm?

3. **[Probabilistic linear embeddings]** Having recently learned about linear dimension reduction, you decided to explore linear maps for yourself.

   Given a dataset $X = \{x_1, \ldots, x_n\} \subset \mathbb{R}^D$ on n points. You want to study the effects of applying a $d \times D$ matrix P to the dataset ($d < D$).

   Since distances between pairs of datapoints is an important property, you want to study how P distorts interpoint Euclidean distances.

   (a) **(3 points)** For two arbitrary (but fixed) datapoints x and x' from the dataset X, the squared Euclidean distance between the projected datapoints, that is, $\|Px - Px'\|^2$ equals (circle the correct option)

   - $$\sum_{i=1}^{d}\sum_{j=1}^{D}\sum_{j'=1}^{D}\Big(P_{ij}P_{ij'}(x_j - x'_j)(x_{j'} - x'_{j'})\Big)$$

   - $$\sum_{i=1}^{d}\sum_{j=1}^{D}\Big(P_{ij}(x_j - x'_j)\Big)^2$$

   - $$\sum_{j=1}^{D}\left[\sum_{i=1}^{d}\sum_{i'=1}^{d}\Big(P_{ij}(x_j - x'_j)P_{i'j}\Big)\right]$$

   - $$\sum_{j=1}^{D}\sum_{i=1}^{d}\sum_{i'=1}^{d}\Big(P_{ij}x_j - P_{i'j}x'_j\Big)^2$$

   - $$\sum_{i=1}^{d}\sum_{i'=1}^{d}\sum_{j=1}^{D}\sum_{j'=1}^{D}\Big(P_{ij}x_j - P_{i'j}x'_{j'}\Big)$$

   - $$\sum_{i=1}^{d}\sum_{j=1}^{D}\sum_{j'=1}^{D}\Big(P_{ij}x_j - P_{ij'}x'_{j'}\Big)$$

   - $$\Big(\sum_{i=1}^{d}\sum_{j=1}^{D}P_{ij}x_j - P_{ij}x'_j\Big)^2$$

   (note: P_{ij} *denotes the ith row and jth column of the matrix* P)

(b) You want to study what effects does a random matrix P has on interpoint distances. For all $1 \leq i \leq d$ and $1 \leq j \leq D$, let each entry P_{ij} be drawn independently uniformly at random[1] from the discrete set $\{-\alpha, +\alpha\}$.

i. **(12 points)** For two arbitrary (but fixed) datapoints x and x' from the dataset X. Compute (simplify as much as possible)

$$\mathbb{E}_P\left[\|Px - Px'\|^2\right]$$

ii. **(5 points)** For what value of α we have: $\mathbb{E}_P\left[\|Px - Px'\|^2\right] = \|x - x'\|^2$?

[1] That is, $P_{ij} = -\alpha$ with probability 0.5 and $P_{ij} = +\alpha$ with probability 0.5.

4. **[Regression and MLE (20 points)]** After your great success in Exam 1, you have once again been invited to a game show. To keep things interesting, the game show host has invented a new game.

   In this new game, the host first *secretly* picks a *unit* vector $w \in \mathbb{R}^d$ (you may assume it is chosen uniformly at random from the unit sphere). Then a uniformly random integer k between 1 and 1000 (inclusive) is chosen and revealed. Finally, there are k trials that occur. Trial $i \in \{1, 2, ..., k\}$ happens as follows:

   - You pick a *unit* vector $x_i \in \mathbb{R}^d$.
   - The quantity $\lambda_i := (w \cdot x_i)^{-2}$ is computed but is *not* revealed to you.
   - Finally y_i is drawn from an exponential distribution with parameter[2] λ_i and you are given y_i dollars.

   Note that when you pick x_i you only get to see y_i (you do not get to see λ_i).

   As a smart ML student, you decide to model this as an ML problem. Specifically, you decide to split your k trials into two groups. You use the first c trials to try to learn what the best x_i to give is. You then use the remaining trials to try to maximize your earnings by repeatedly picking the (same) best possible input. We will focus on the first c trials (the learning or training phase). For this phase you decide to pick each x_i uniformly at random from the unit sphere. We denote the set of inputs and outputs thus obtained by $S := \{(x_1, y_1), (x_2, y_2), ..., (x_c, y_c)\}$.

   (a) **(4 points)** Armed with your new regression knowledge, you first decide to view this as a regression problem. Show that the optimal L_2 regressor for this problem is $f^*(x) = (w \cdot x)^2$. In other words, letting x and y be random variables created as above (you pick x uniformly at random from the unit ball, y is then created as outlined in the bullet points above), show that $f^*(x) = (w \cdot x)^2$ is the function that minimizes $\mathbb{E}_{x,y}(f(x) - y)^2$ over all $f : \mathbb{R}^d \to \mathbb{R}$.

[2] In other words $p(y_i) = \lambda_i e^{-\lambda_i y_i}$. You can use any known facts about the exponential distribution without proof. Most useful is that the mean of the exponential distribution with parameter λ_i is $1/\lambda_i$.

(b) **(4 points)** Using the result from part (a), explain why using OLS to learn a prediction function from S is not a good idea.

(c) **(5 points)** What simple preprocessing step could be used on S to fix the issue in (b) and allow us to estimate $f^*(x)$ using the OLS algorithm? Make sure to explain why your suggested preprocessing step would help.

(d) **(4 points)** You also consider using MLE rather than OLS in the training phase. Let w_{MLE} be the MLE of w given S. The best way to find w_{MLE} is by optimizing the *negative log likelihood* function. Write down the negative log likelihood optimization problem for this particular data distribution and simplify it as much as possible. Make sure to completely specify the optimization problem. In particular, specify what variable(s) you optimize over, whether it is a minimization or a maximization problem, whether the optimization is subject to any constraints, etc. You do not need to solve the optimization problem, just simplify it as much as possible.

(e) **(3 points)** Assume that you were able to find an estimate $\hat{w}$ of w using S. You now want to pick a single value x to use in all subsequent trials. If your only goal for those trials is to maximize profits what unit length x should you use? Why?

5. **[Learning Theory]** For a set $I \subseteq \{1, 2, \ldots, n\}$ we define a classifier $h_I : \{0,1\}^n \to \{0,1\}$ as follows. For a binary vector $\vec{x} = (x_1, \ldots, x_n) \in \{0,1\}^n$,

$$h_I(\vec{x}) = \left(\sum_{i \in I} x_i \cdot \mathbf{1}[\, i \text{ is even} \,] \right) \bmod 2,$$

where

- $\mathbf{1}[\cdot]$ is the indicator function, and
- $(\cdot) \bmod 2$ is the modulus 2 operation; thus, when the left expression $(\cdot)$ is even, the overall expression returns 0, and when the left expression $(\cdot)$ is odd the overall expression returns 1.

(a) For this part, assume $n = 2$, $I_0 = \{\}$, $I_1 = \{1\}$, $I_2 = \{2\}$ and $I_3 = \{1, 2\}$.

i. **(3 points)** Two classifiers h_α and h_β are considered *identical* if for all $\vec{x} \in \{0,1\}^2$, $h_\alpha(\vec{x}) = h_\beta(\vec{x})$.

Consider four classifiers h_{I_0}, h_{I_1}, h_{I_2}, and h_{I_3}. List all the classifiers which are identical to each other.

(*example response: classifiers h_{I_0}, h_{I_1} and h_{I_2} are identical*)

ii. **(2 points)** Consider an arbitrary (but fixed) dataset $X \subseteq \{0,1\}^2$. How many different ways can the classifier h_{I_2} label the dataset X?

iii. **(3 points)** Define $\mathcal{H} := \{h_{I_1}, h_{I_2}, h_{I_3}\}$ as the hypothesis class of three specified classifiers. What is the VC-dimension of $\mathcal{H}$? Justify your answer for full credit.

(b) **(12 points)** For an arbitrary (but fixed) $n \in \mathcal{N} = \{1, 2, 3, \ldots\}$. Define $\mathcal{H}_n := \{h_I : I \subseteq \{1, \ldots, n\}\}$. Provide the tightest upper and lower bounds for the VC-dimension of $\mathcal{H}_n$.

[blank page 1 for scratch work]

[blank page 2 for scratch work]

The End.