COMS 4771 HW4

Due: Sat Apr 21, 2018 at 11:59pm

You are allowed to work in groups of (at max) three students. These group members don't necessarily have to be the same from previous homeworks. Only one submission per group is required by the due date. Name and UNI of all group members must be clearly specified on the homework. You must cite all the references you used to do this homework. You must show your work to receive full credit.

1 **[PAC learning and VC dimension]**

(a) Consider a domain with n binary features and binary class labels. Let $\mathcal{F}_d$ be the hypothesis class that contains all decision trees over these features that have depth at most d. Give the tightest possible VC-dimension estimate of $\mathcal{F}_d$.

(b) **[a sharper learning rate for finding perfect predictors!]** Consider a finite size hypothesis class $\mathcal{F}$ that contains the perfect predictor, that is, $\exists f^* \in \mathcal{F}$, s.t. $\forall (x, y) \sim \mathcal{D}(X \times Y)$, $y = f^*(x)$. We will show that for such a hypothesis class $\mathcal{F}$, we can derive a sharper sample complexity rate than what was shown in class (i.e., Occam's Razor bound).

(i) For a fixed $\epsilon > 0$, we say that a classifier f is ϵ-bad if

$$P_{(x,y)\sim\mathcal{D}}[f(x) = y] < 1 - \epsilon.$$

What is the chance that output of a given ϵ-bad f matches the true label for all m samples drawn i.i.d. from $\mathcal{D}$? That is, for a given ϵ-bad f, calculate

$$P_{(x_i,y_i)_{i=1}^m\sim\mathcal{D}}\big[\forall i \;\; f(x_i) = y_i \;\mid\; f \text{ is } \epsilon\text{-bad}\big].$$

(ii) What is the chance that event in part (i) occurs for *some* $f \in \mathcal{F}$? That is, calculate

$$P_{(x_i,y_i)_{i=1}^m\sim\mathcal{D}}\big[\exists\ \epsilon\text{-bad } f \in \mathcal{F}, \text{ such that } \forall i \;\; f(x_i) = y_i\big].$$

(iii) Observe that event is part (ii) is a "bad" event: an ERM-algorithm can pick a ϵ-bad classifier if it happens to perfectly classify on the m i.i.d. samples. We want to limit the probability of this bad event by at most $\delta > 0$ amount. So how many i.i.d. samples suffice to ensure that with probability at least $1 - \delta$, an ERM-algorithm does not return an ϵ-bad classifier?

(c) Given a collection of models $\mathcal{F}$, suppose you were able to develop an algorithm $\mathcal{A} : (x_i, y_i)_{i=1}^n \mapsto f_n^{\mathcal{A}}$ (that is, given n labeled training samples, $\mathcal{A}$ returns a model $f_n^{\mathcal{A}} \in \mathcal{F}$) that has the following property: for all $\epsilon > 0$, with probability 0.55 (over the draw of $n = O(\frac{1}{\epsilon^2})$ samples,

$$\text{err}(f_n^{\mathcal{A}}) - \inf_{f \in \mathcal{F}} \text{err}(f) \leq \epsilon,$$

where $\text{err}(f) := P_{(x,y)}[f(x) \neq y]$.

Show that one can construct an algorithm $\mathcal{B} : (x_i, y_i)_{i=1}^{n'} \mapsto f_{n'}^{\mathcal{B}}$ with the property: for all $\epsilon > 0$ and all $\delta > 0$, with probability at least $1 - \delta$ over a draw of n' samples:

$$\text{err}(f_{n'}^{\mathcal{B}}) - \inf_{f \in \mathcal{F}} \text{err}(f) \leq \epsilon.$$

Moreover show that $n' = \text{poly}(\frac{1}{\epsilon}, \frac{1}{\delta})$ samples are enough for the algorithm $\mathcal{B}$ to return such a model $f_{n'}^{\mathcal{B}}$. Hence, the model class $\mathcal{F}$ is *efficiently* PAC-learnable.
(Hint: Algorithm $\mathcal{B}$ can make multiple calls to the algorithm $\mathcal{A}$.)

2 **[Exploring k-means in detail]** Recall that in k-means clustering we attempt to find k cluster centers $c_j \in \mathbb{R}^d, j \in \{1, \ldots, k\}$ such that the total (squared) distance between each datapoint and the nearest cluster center is minimized. In other words, we attempt to find $c_1, ..., c_k$ that minimizes

$$\sum_{i=1}^{n} \min_{j \in \{1,\ldots,k\}} \|x_i - c_j\|^2, \tag{1}$$

where n is the total number of datapoints. To do so, we iterate between assigning x_i to the nearest cluster center and updating each cluster center c_j to the average of all points assigned to the jth cluster (aka Lloyd's method).

(a) **[it is unclear how to find the best k, i.e. estimate the correct number of clusters!]** Instead of holding the number of clusters k fixed, one can think of minimizing (1) over both k and c. Show that this is a bad idea. Specifically, what is the minimum possible value of (1)? what values of k and c result in this value?

(b) **[efficient optimal solution when either $k = 1$ or $d = 1$]** Optimizing the k-means objective (1) for the general case when $k \geq 2$ and $d \geq 2$ is NP-hard. But one can find an efficient optimial solution when either $k = 1$ or $d = 1$.

(i) What is the optimal objective value and the setting of the cluster centers in the case of $k = 1$?

(ii) For the case $d = 1$ (and $k \geq 2$), show that Lloyd's algorithm is *not* optimal. That is, there is a suboptimal setting of cluster assignment for some dataset (with $d = 1$) for which Lloyd's algorithm will not be able to improve the solution.

(iii) **(note: this part is extra credit, only attempt it once everything else is done)** Propose an efficient algorithm (that is, an algorithm that runs in polynomial in n and k steps) for optimally solving k-means for $d = 1$ case.
(Hint: Use the fact that when $d = 1$, the dataset can be ordered as $x_{i_1} \leq x_{i_2} \leq \cdots \leq x_{i_n}$ and each cluster in the optimal assignment will be a nonoverlapping interval)

(c) **[kernelizing k-means]** k-means with Euclidean distance metric assumes that each pair of clusters is linearly separable. This may not be the case. A classic example is where we have two clusters corresponding to data points on two concentric circles in the $\mathbb{R}^2$.

(i) Implement Lloyd's method for k-means algorithm and show the resulting cluster assignment for the dataset depicted above. Give two more examples of datasets in $\mathbb{R}^2$, for which optimal k-means setting results in an undesirable clustering. Show the resulting cluster assignment for the two additional example datasets.

Let ϕ denote an explicit non-linear feature space mapping in some inner product space. We will show how one can derive an *implicit* kernel-based version of the Lloyd's method for k-means, where we only operate on data as $\phi(x_i) \cdot \phi(x_j) = K(x_i, x_j)$.

(ii) Let z_{ij} be an indicator that is equal to 1 if the $\phi(x_i)$ is currently assigned to the jth cluster and 0 otherwise ($1 \leq i \leq n$ and $1 \leq j \leq k$). Show that the jth cluster center c_j can be written as $\sum_{i=1}^{n} \alpha_{ij} \phi(x_i)$, where α_{ij} only depends on z_{ij} variables.

(iii) Given any two data points $\phi(x_i)$ and $\phi(x_j)$, show that the square distance $\|\phi(x_i) - \phi(x_j)\|^2$ can be computed using only (linear combinations of) inner products.

(iv) Given the results of parts (ii) and (iii), show how to compute the square distance $\|\phi(x_i) - c_j\|^2$ using only (linear combinations of) inner products between the data points $x_1, ..., x_n$.

(v) From results from parts (ii), (iii) and (iv), propose the algorithm for kernelized version of Lloyd's method of finding k-means.

(vi) Implement your proposed kernelized k-means method and run it on the three example datasets of part (i). Compare the resulting cluster for various choices of kernels (e.g. linear kernel, quadratic kernel, rbf kernel).
(submit your datasets and kernelized code on Courseworks to receive full credit)

3 **[different learning rates for different strategies]** Here will explore how the rate by which an algorithm learns a concept can change based on how labeled examples are obtained. For that we look at three settings: (i) an active learning setting where the algorithm has the luxury of specifying a data point and querying its label, (ii) a passive learning setting where labeled examples are drawn at random and (iii) an adversarial setting where training examples are given by an adversary that tries to make your life hard.

Consider a binary classification problem where each data point consists of d binary features. Let $\mathcal{F}$ be the hypothesis class of conjunctions of subsets of the d features and their negations. So for example one hypothesis could be $f_1(x) = x_1 \wedge x_2 \wedge \neg x_d$ (where $\wedge$ denotes the logical "and" and $\neg$ denotes the logical "not"). Another hypothesis could be $f_2(x) = \neg x_3 \wedge x_5$. A conjunction in $\mathcal{F}$ cannot contain both x_i and $\neg x_i$. We assume a consistent learning scenario where there exists a hypothesis $f^* \in \mathcal{F}$ that is consistent with the true label for all data points.

(i) In the active learning setting, the learning algorithm can query the label of an unlabeled example. Assume that you can query *any possible example*. Show that, starting with a single positive example, you can exactly learn the true hypothesis f^* using d queries.

(ii) In the passive learning setting, where the examples are drawn i.i.d. from an underlying fixed distribution $\mathcal{D}$. How many examples are sufficient to guarantee a generalization error less than ϵ with probability $\geq 1 - \delta$?

(iii) Show that if the training data is not representative of the underlying distribution, a consistent hypothesis f^* can perform poorly. Specifically, assume that the true hypothesis f^* is a conjunction of k out of the d features for some $k > 0$ and that all possible data points are equally likely. Show that there exists a training set of $2^{(d-k)}$ unique examples and a hypothesis f that is consistent with this training set but achieves a classification error $\geq 50\%$ when tested on all possible data points.

4 **[From distances to embeddings]** Your friend from overseas is visiting you and asks you the geographical locations of popular US cities on a map. Not having access to a US map, you realize that you cannot provide your friend accurate information. You recall that you have access to the relative distances between nine popular US cities, given by the following distance matrix D:

Distances (D)	BOS	NYC	DC	MIA	CHI	SEA	SF	LA	DEN
BOS	0	206	429	1504	963	2976	3095	2979	1949
NYC	206	0	233	1308	802	2815	2934	2786	1771
DC	429	233	0	1075	671	2684	2799	2631	1616
MIA	1504	1308	1075	0	1329	3273	3053	2687	2037
CHI	963	802	671	1329	0	2013	2142	2054	996
SEA	2976	2815	2684	3273	2013	0	808	1131	1307
SF	3095	2934	2799	3053	2142	808	0	379	1235
LA	2979	2786	2631	2687	2054	1131	379	0	1059
DEN	1949	1771	1616	2037	996	1307	1235	1059	0

Being a machine learning student, you believe that it may be possible to infer the locations of these cities from the distance data. To find an embedding of these nine cities on a two dimensional map, you decide to solve it as an optimization problem as follows.

You associate a two-dimensional variable x_i as the unknown latitude and the longitude value for each of the nine cities (that is, x_1 is the lat/lon value for BOS, x_2 is the lat/lon value for NYC, etc.). You write down the an (unconstrained) optimization problem

$$\text{minimize}_{x_1,\ldots,x_9} \sum_{i,j} \big(\|x_i - x_j\| - D_{ij}\big)^2,$$

where $\sum_{i,j}(\|x_i - x_j\| - D_{ij})^2$ denotes the embedding discrepancy function.

(i) What is the derivative of the discrepancy function with respect to a location x_i?

(ii) Write a program in your preferred language to find an optimal setting of locations $x_1, \ldots, x_9$. You must submit your code to Courseworks to receive full credit.

(iii) Plot the result of the optimization showing the estimated locations of the nine cities. (here is a sample code to plot the city locations in Matlab)

```
>> cities={'BOS','NYC','DC','MIA','CHI','SEA','SF','LA','DEN'};
>> locs = [x1;x2;x3;x4;x5;x6;x7;x8;x9];
>> figure; text(locs(:,1), locs(:,2), cities);
```

What can you say about your result of the estimated locations compared to the actual geographical locations of these cities?