1. Preliminaries and some basic tools

**Assigned reading:** Chapters 1, 2, and 5 of the course notes. (Introduction, Concentration Inequalities, and Formulation of the Learning Problem). **Additional recommended reading:** Shalev-Shwartz and Ben-David, *Understanding Machine Learning, from Theory to Algorithms*, Chapters 1-3 and Appendix B.

**Problems and solutions**

1. **[Empirical cumulative distribution functions on the reals]**

   Let $F$ be a cumulative distribution function (CDF) on $\mathbb{R}$ and let $X_1, \ldots, X_n$ be $n$ independent random variables with distribution $F$. Denote the empirical distribution function by $F_n(c) = \frac{1}{n} \sum_{i\in [n]} 1\{X_i \leq c\}$ for $c \in \mathbb{R}$ and let $\Delta_n = \sup_{c \in \mathbb{R}} |F_n(c) - F(c)|$.

   (a) Use the Hoeffding inequality to get an upper bound on $\mathbb{P}\left\{ |F_n(c) - F(c)| \geq \frac{t}{\sqrt{n}} \right\}$ valid for any $c \in \mathbb{R}$ and $t \geq 0$ that depends only on $t$.

   **Solution:** For any $c$, $F_n(c)$ is a binomial random variable with parameters $n$ and $p = F(c)$. Or $F_n(c)$ is the sum of $n$ independent random variables with values in the interval $[0, 1/n]$, and $\mathbb{E}[F_n(c)] = F(c)$. Hence, by Hoeffding’s inequality, $\mathbb{P}\left\{ |F_n(c) - F(c)| \geq \frac{t}{\sqrt{n}} \right\} \leq 2 \exp\left(-\frac{(2t/n)^2}{n(1/n)^2}\right) = 2e^{-2t^2}$.

   (b) Using your answer to (a), find an upper bound on $E[|F_n(c) - F(c)|]$ valid for any $c \in \mathbb{R}$ that depends only on $n$. (Hint: For any nonnegative random variable $Y$, $E[Y] = \int_0^\infty \mathbb{P}\{Y > t\} dt$.)

   **Solution:** If $Y = \sqrt{n}|F_n(c) - F(c)|$ then by (a) and the hint, $E[Y] \leq \int_0^\infty 2e^{-2t^2} = \sqrt{\pi}/2$. (Compare to integration of a Gaussian density with variance 1/4.) So for any $c$, $E[|F_n(c) - F(c)|] \leq \sqrt{\pi/(2n)}$.

   (c) Use McDiarmid’s bound to find an upper bound on $P\left\{ |\Delta_n - E[\Delta_n]| \geq \frac{t}{\sqrt{n}} \right\}$. (A considerably stronger result is known, namely, $\mathbb{P}\left\{ |\Delta_n| \geq \frac{t}{\sqrt{n}} \right\} \leq 2e^{-2t^2}$, due to Massart, “The tight constant in the Dvoretzky-Kiefer-Wolfowitz inequality, *Annals of Probability*, vol. 18, no. 3, 1990, pp. 1269-1283.)

   **Solution:** $\Delta_n = G(X_1, \ldots, X_n)$ where $G_n$ has the bounded difference property for $c_i = \frac{1}{n}$, so that $P\left\{ |\Delta_n - E[\Delta_n]| \geq \frac{t}{\sqrt{n}} \right\} \leq 2e^{-2t^2}$ (same bound as in part (a)).

2. **[PAC learnability of right subintervals]**

   Consider the concept learning problem for the triple $(X, \mathcal{P}, \mathcal{C})$, where $X = [0, 1]$, $\mathcal{C} = \{ [\tau, 1] : 0 \leq \tau \leq 1 \}$, and $\mathcal{P}$ is the set of all probability distributions on $X$. Describe an ERM classifier $\hat{C}_n$ and show that it PAC learns $\mathcal{C}$ in the realizable case to accuracy $\epsilon$ with probability at least $1 - \delta$, (i.e. $P\{P(C^* \Delta \hat{C}) \leq \epsilon \} \geq 1 - \delta$, if the sample size is at least $n(\epsilon, \delta) = \left[ \frac{\log(1/\delta)}{\epsilon} \right]$). The training set is given by $Z^n = (Z_1, \ldots, Z_n)$, where $Z_i = (X_i, Y_i) = (X_i, 1_{(X_i \in C^*)})$ and $C^* = [\tau^*, 1]$ is the target concept.

   **Solution:** Let $\hat{C}_n = \emptyset$ if $Y_i = 0$ for all $i$. Else, let $\hat{C}_n = [\hat{\tau}_n, 1]$, where $\hat{\tau}_n = \min\{X_i : Y_i = 1\}$. The classifier $\hat{C}_n$ correctly classifies all the given data, so it is an ERM rule. To show the PAC learnability statement, fix $\epsilon, \delta > 0$, $\tau^* \in [0, 1]$, $n \geq 1$, and a probability distribution $P$ on $X$. It suffices to show that when $Z^n$ is drawn according to $P$, the classifier $\hat{C}_n$ has expected classification error less than or equal to $\epsilon$ with probability at least $1 - \delta$ if $n \geq \left[ \frac{\log(1/\delta)}{\epsilon} \right]$.
Note that $P(\hat{C}_n \subset C^*) = 1$, so the probability of classification error given $\hat{C}_n$ (i.e. the accuracy of $\hat{C}_n$, given its realization) is $P(C^* \setminus \hat{C}_n) = P(\{\tau^*, \tau_n\})$. If $P(C^*) < \epsilon$, then $\hat{C}_n$ is $\epsilon$ accurate with probability one, so for the remainder of this analysis we assume $P(C^*) \geq \epsilon$. Let $\tau_n = \min(\tau : \tau \geq \tau^* \text{ and } P(\{\tau^*, \tau\}) \geq \epsilon)$. Note that $\tau_n$ is determined by $C^*$ and $P$ and does not depend on the data. The value $\tau_n$ exists because the set the minimum operator is applied to contains the point 1 and so is not empty, and it is closed under limits from the right because $P(\{\tau^*, \tau\})$ is right-continuous in $\tau$ by continuity of probability. Note that $P(\{\tau^*, \tau_n\}) \geq \epsilon \geq P(\{\tau^*, \tau_o\})$. It at least one data point $X_i$ falls in the interval $[\tau^*, \tau_n]$, then $C^* \setminus \hat{C}_n \subset [\tau^*, \tau_n]$ and hence $P(C^* \setminus \hat{C}_n) \leq \epsilon$. So $P(\hat{C}_n \neq C^*) \leq (1 - \epsilon)^n \leq e^{-\epsilon n}$. So if $n \geq \lceil \frac{\log(1/\delta)}{\epsilon} \rceil$, $P(\hat{C}_n \neq C^*) \leq \delta$.

3. [A converse result – aka no free lunch theorem]

Let $n$ be a fixed positive integer and suppose $B_1, \ldots, B_{2n}$ be Bernoulli($1/2$) random variables, modeling fair coin tosses. Let $X_1, \ldots, X_n$ be selector random variables, that are each uniformly distributed over $[2n]$. Finally, assume all $3n$ of the above random variables are mutually independent. These variables have the following interpretation. A random data sample is given by $Z^n = (Z_1, \ldots, Z_n)$, where $Z_i = (X_i, B_{X_i})$. Thus, if $G = \{k : X_i = k \text{ for at least one } i \in [n]\}$, then seeing the data sample is equivalent to learning $G$ and the values of the coin tosses for indices $k \in G$. Suppose $A$ is a learning algorithm that takes $Z^n$ as input and outputs a classifier function $\hat{f} : [2n] \rightarrow \{0, 1\}$.

(a) Show that for any choice of $A$ and any $k \in [2n]$, $P\{\hat{f}(k) \neq B_k\} \geq 0.25$.

Solution: By the union bound, the random set $G$ is such that $P\{k \in G\} \leq \sum_{i \in [n]} P\{Z_i = k\} \leq 0.5$. In words, the value $B_k$ is included in the data sample with probability less than or equal to $0.5$. Given $\{k \not\in G\}$, $B_k$ is conditionally independent of the data $Z^n$, which determines $\hat{f}$. So, given $\{k \not\in G\}$, $\hat{f}(k)$ and $B_k$ are conditionally independent. Thus, $P\{\hat{f}(k) \neq B_k\} \geq P\{\hat{f}(k) \neq B_k, k \not\in G\} = P\{\hat{f}(k) \neq B_k, k \notin G\} \geq P\{\hat{f}(k) \neq B_k \text{ and } k \not\in G\} \geq 0.5 \geq 0.25$.

(b) Thus, by part (a), the average fraction of variables among $\{B_1, \ldots, B_{2n}\}$ that $\hat{f}$ correctly predicts is at most 75%. Find an upper bound smaller than one on the probability that $\hat{f}$ correctly predicts at least 90% of the variables. (Apply Markov’s inequality.)

Solution: Let $V$ denote the fraction of variables that $\hat{f}$ correctly predicts (i.e. $V = |\{k : \hat{f}(k) = B_k\}|/(2n)$). By part (a), $E[V] \leq 0.75$. Thus, by Markov’s inequality, $P\{V \geq 0.90\} \leq \frac{0.75}{0.90} = \frac{5}{6}$.

(c) Show that for any learning algorithm $A$, there exists a deterministic choice $b_1, \ldots, b_{2n}$ (depending on $A$), such that, given $(B_1, \ldots, B_{2n}) = (b_1, \ldots, b_{2n})$, the conditional probability that $\hat{f}$ correctly predicts at least 90% of the variables is less than or equal to 5/6. (Once $A$ and the $b$’s are fixed, the randomness remaining is due to the randomness of the selector variables $X_1, \ldots, X_n$.)

Solution: By part (b), for any algorithm $A$, $P\{\hat{f} \text{ produced by } A \text{ is 90% accurate} \} \leq 5/6$. Thus, by the total law of probability, $\sum_{b} P\{\hat{f} \text{ produced by } A \text{ is 90% accurate}|B = b\} P\{B = b\} \leq 5/6$. Since the minimum over $b$ is less than or equal to the average, it follows there exists a value of $b$ so that $P\{\hat{f} \text{ produced by } A \text{ is 90% accurate}|B = b\} \leq 5/6$.

(d) In this part we consider a single realizable concept learning problem $(X, P, F)$ that simultaneously has the above problem embedded in it for all $n \geq 1$, and show that it is not PAC learnable. Let $X = (0, 1)$ and for $m \geq 1$ let $F_m$ denote the set of all binary valued functions on $X$ that are constant over intervals of the form $[\frac{i}{m}, \frac{i+1}{m}]$ for some $0 \leq i \leq m - 1$. Note that $|F_m| = 2^m$, because selecting an $f$ in $F_m$ is equivalent to deciding its value on each of the $m$ subintervals. Let $F = \cup_{m \geq 1} F_m$. Also, let $P$ contain only a single distribution $P$, namely, the uniform probability distribution over $X$. Thus, the data is represented by $Z^n = (Z_1, \ldots, Z_n)$, where $Z_i = (X_i, f^*(X_i))$ such that the $X_i$’s are independent and uniformly distributed over $(0, 1]$ and $f^*$ is the true concept, with $f^* \in F$. Show that $(X, P, F)$ is not PAC learnable. You may give a high level description without all details.
Solution: It suffices to show that for any \( n \) and choice of learning algorithm \( A_n \), there exists a choice of \( f^* \in \mathcal{F} \) such that \( P^n \{ P(f^*(X) \neq f_n(X)) > 0.1 \} \geq 1/6 \), where \( f_n = A_n(Z^n) \) and \( X \) represents a fresh feature variable, uniformly distributed over \( X \) and independent of \( X_1, \ldots, X_n \).

By the reasoning of (a)-(c) above, there exists such a predictor \( f^* \in \mathcal{F}_{2n} \). The predictors in \( \mathcal{F}_{2n} \) are in one-to-one correspondence with binary vectors of length \( 2n \). As in part (a), if the true classifier \( f^* \) is random and uniformly distributed over all \( 2^{2n} \) possibilities in \( \mathcal{F}_{2n} \), then the algorithm learns at most half of the bits from the data samples, and hence has expected accuracy less than or equal to 0.75. Therefore, the probability \( f_{mn} \) is accuracy greater than or equal to 0.90% on a fresh data sample is less than or equal to 5/6. Since the average case performance is at least as good as worst case, there exists a value of \( (b_1, \ldots, b_{2n}) \) so that the given learning algorithm has accuracy less than 90% with probability at least 1/6.

4. [Bregman divergence]

Let \( S \) denote a convex subset of \( \mathbb{R}^d \) with nonempty relative interior\(^1\), \( \text{ri}(S) \), and let \( \phi \) be a strictly convex function on \( S \) that is differentiable on \( \text{ri}(S) \). The Bregman divergence generated by \( \phi \) is the function \( d_\phi : S \times \text{ri}(S) \rightarrow \mathbb{R}_+ \) defined by \( d_\phi(p, q) = \phi(p) - \phi(q) - \langle p - q, \nabla \phi(q) \rangle \). It is positive if \( p \neq q \) and strictly convex in \( p \), but not necessarily convex in \( q \).

Solution: (a) Identify the Bregman divergence for the special cases (i) \( S = \mathbb{R}^d \), \( \phi(p) = \|p\|^2 = \sum_i p_i^2 \), and (ii) \( S \) is the set of \( d \) dimensional probability vectors and \( \phi(p) = \sum_i p_i \ln p_i \), where \( 0 \ln 0 \) is taken to be zero (i.e. \( \phi(p) \) is the negative of the entropy of \( p \)).

Solution: (b) Let \( p_1, \ldots, p_n \in S \). Their centroid, \( \bar{p} \), is defined by \( \bar{p} = \frac{1}{n} \sum_{i=1}^n p_i \). Show that if \( \bar{p} \in \text{ri}(S) \), then \( \bar{p} \) is the unique minimizer of the function \( q \mapsto \frac{1}{n} \sum_{i=1}^n d_\phi(p_i, q) \).

Solution: Given any \( q \in \text{ri}(S) \), cancelling terms of the form \( \phi(p_i) \) yields:

\[
\frac{1}{n} \sum_{i=1}^n d_\phi(p_i, q) - \frac{1}{n} \sum_{i=1}^n d_\phi(p_i, \bar{p}) = -\phi(q) - \langle \bar{p} - q, \nabla \phi(q) \rangle + \phi(\bar{p}) + \langle \bar{p} - p, \nabla \phi(q) \rangle
\]

Thus, \( \bar{p} \) gives a strictly smaller value than \( q \) unless \( q = \bar{p} \).

(c) Briefly explain the significance of part (b) for generalization of the \( K \)-means clustering algorithm.


Solution: The \( K \)-means clustering algorithm is an iterative algorithm for clustering data, and a version of it can be naturally defined using Bregman divergence. Given fixed points \( p_1, \ldots, p_n \in S \), the algorithm iteratively calculates \( K \) centers \( q^1, \ldots, q^K \in \text{ri}(S) \) and a \( K \) partition of the set of points, \( \{C_k\}_{k=1}^K \). Given initial values of the centers, the algorithm alternates between two steps:

(Assignment step) For \( i \in [n] \), assign point \( i \) to cluster \( C_k \) such that \( k = \arg \min_{k \in [K]} d_\phi(p_i, q^k) \), with ties broken arbitrarily.

(Recentering step) For \( k \in [K] \), let \( q^k = \frac{1}{|C_k|} \sum_{i \in C_k} p_i \).

---

\(^1\)The aff span of a set \( C \subset \mathbb{R}^n \) is defined by \( \text{aff}(C) = \{ \sum_{k=1}^K \lambda_k x_k : k \geq 1, x_k \in C, \sum \lambda_k = 1 \} \). (Similar to convex hull but without the constraint that \( \lambda_k \geq 0 \) for all \( k \)) The relative interior of a convex set \( C \) is the set of \( x \in C \) such that for some \( \epsilon > 0 \), \( B(\epsilon, x) \cap \text{aff}(C) \subset C \), where \( B(\epsilon, x) \) is the radius \( \epsilon \) ball centered at \( x \).
Part (b) simply ensures that the recentering step in the algorithm is selecting centers that minimize the average Bregman divergence between the points in a cluster and the cluster center. Thus, if \( \pi(i) \) is the cluster that point \( i \) is assigned to, the loss function \( \sum_i d(\pi(i), q^{\pi(i)}) \) is non-increasing in each step of the algorithm.

(d) The Kullback-Leibler (KL) divergence for probability distributions \( p \) and \( q \), defined by \( d_{KL}(p||q) = \sum_i p_i \ln \frac{p_i}{q_i} \), is an example of an \( f \) divergence, which is something of the form \( d_f(p, q) = \sum_i q_i f \left( \frac{p_i}{q_i} \right) \), corresponding to the function \( f(u) = u \ln u \). (For \( f \) divergence, the function \( f \) should be convex function over \( \mathbb{R}_+ \), strictly convex at 1, with \( f(1) = 0 \). You may assume \( f \) is also twice continuously differentiable.) Show that the only \( f \) divergence that is also a Bregman divergence for probability distributions, is the KL divergence, up to a constant multiple. (Hint: For Bregman divergences, \( p \mapsto d(p, q) - d(p, q') \) is a linear function over the space of probability distributions, so its Hessian must be proportional to the all ones matrix. By computation, the Hessian is a diagonal matrix, so it must be zero.)

Solution: Consider an \( f \) divergence that is also a Bregman divergence. Following the hint, we calculate the Hessian \( H \) of \( p \mapsto d_f(p, q) - d_f(p, q') \) with respect to \( p \). We find it is diagonal with

\[
H_{k,k} = \frac{\partial^2}{(\partial p_k)^2} \left( q_k f \left( \frac{p_k}{q_k} \right) - q'_k f \left( \frac{p_k}{q'_k} \right) \right) = \frac{1}{q_k} f'' \left( \frac{p_k}{q_k} \right) - \frac{1}{q'_k} f'' \left( \frac{p_k}{q'_k} \right)
\]

In order that \( H_{k,k} \equiv 0 \) for \( q_k, q'_k \in (0, 1) \), there must be a constant \( c(p_k) \) so that \( \frac{1}{q} f'' \left( \frac{p}{q} \right) = c(p_k) \) for all \( q \in (0, 1) \). Or, setting \( x = \frac{p}{q} \), \( f''(x) = \frac{c(p_k)p_k}{x} \) for all \( x > 0 \). Integrating yields \( f'(x) = c(p_k)p_k \ln(x) + C \) and thus \( f(x) = c(p_k)p_k (x \ln x - x) + Cx + C' \). Thus, also taking into account that \( f(1) = 0 \), \( f \) must have the form \( f(x) = ax \ln x + b(x - 1) \). The term \( b(x - 1) \) makes no contribution when used in the formula for \( f \) divergence, and can be dropped. Therefore, the given Bregman/\( f \)-divergence is equivalent to \( f \) divergence with \( f(x) = ax \ln x \) for some constant \( a \), or \( d_f \) is proportional to the KL divergence.

5. [Getting started on Python]

Suppose the \( k \) nearest neighbor (KNN) classifier (not to be confused with the \( K \)-means clustering algorithm) is used for classification of the classic Iris flower data set, with uniform weighting of \( k \) nearest neighbors. Use only the first two measurements for each sample, i.e. the sepal length and width. Suppose the 150 samples are randomly, uniformly divided into 100 training samples, used to train the classifier, and 50 samples to test the classifier. The scores on the training set and on the test set are defined to be the fraction of correct classifications for those sets. The scores are random because of the random splitting of the samples into two groups. Here is what you need to compute: (a) Determine the number of neighbors \( k \) that maximizes the expected score of the trained classifier on the training sample. (b) Determine the number of neighbors \( k \) that maximizes the expected score of the trained classifier on the test data. Use 100 repetitions of the training/testing experiment to estimate these average scores. To get started, see http://nbviewer.jupyter.org/urls/courses.engr.illinois.edu/ece543/sp2019/ece543_PythonProblem1.ipynb?flush_cache=true.