10-601 Machine Learning: Homework 6

Due 5 p.m. Friday, April 3, 2015

Instructions

- **Late homework policy:** Homework is worth full credit if submitted before the due date, half credit during the next 48 hours, and zero credit after that. You *must* turn in at least $n-1$ of the n homeworks to pass the class, even if for zero credit.
- **Collaboration policy:** Homeworks must be done individually, except where otherwise noted in the assignments. "Individually" means each student must hand in their own answers, and each student must write and use their own code in the programming parts of the assignment. It is acceptable for students to collaborate in figuring out answers and to help each other solve the problems, though you must in the end write up your own solutions individually, and you must list the names of students you discussed this with. We will be assuming that, as participants in a graduate course, you will be taking the responsibility to make sure you personally understand the solution to any work arising from such collaboration.
- **Online submission:** You must submit your solutions online on autolab. We recommend that you use LaTeX to type your solutions to the written questions, but we will accept scanned solutions as well. On the Homework 6 autolab page, you can download the template, which is a tar archive containing a blank placeholder pdf for the written questions. Replace each pdf file with one that contains your solutions to the written questions. When you are ready to submit, create a new tar archive of the top-level directory and submit your archived solutions online by clicking the "Submit File" button. You should submit a single tar archive identical to the template, except with the blank pdfs replaced by your solutions for the written questions. You are free to submit as many times as you like. ***DO NOT*** change the name of any of the files or folders in the submission template. In other words, your submitted files should have exactly the same names as those in the submission template. Do not modify the directory structure.

Problem 1: Expectation Maximization

Mixture models are helpful for modeling unknown subpopulations in data. If we have a collection of data points $X = \{X_1, ..., X_n\}$, where each X_i is drawn from one of K possible distributions, we can introduce a discrete-valued random variable $Z_i \in \{1, ..., K\}$ that indicates which distribution X_i is drawn from. In lecture, we discussed Gaussian mixture models, where each sample X_i is drawn from a Gaussian distribution according to the value of its mixture component. This question deals with the *categorical* mixture model, where each observation X_i is a discrete value drawn from a categorical distribution, rather than a continuous value drawn from a Gaussian distribution. The parameter for a categorical distribution is a K-dimensional vector π that lists the probability of each of K possible values (therefore, $\sum_k \pi_k = 1$). For example, suppose our categorical mixture model has 3 underlying distributions. Then, each Z_i could take on one of three values, $\{1, 2, 3\}$, with respective probabilities $\{\pi_1, \pi_2, \pi_3\}$; equivalently, $Z_i \sim \text{Categorical}(\pi)$, where $\pi = [\pi_1, \pi_2, \pi_3]^T \in \mathbb{R}^3$. The observation X_i is then generated from another categorical distribution, depending on the value of Z_i. Equation (1) summarizes the generative process for a categorical mixture model.

$$\begin{aligned} Z_i &\sim \text{Categorical}(\pi) \\ X_i &\sim \text{Categorical}(\theta_{Z_i}) \end{aligned} \tag{1}$$

For this model, where we observe X but not Z, we want to learn the parameters of the K categorical components $\Theta = \{\pi, \theta_1, ..., \theta_K\}$, where each $\theta_k \in \mathbb{R}^M$ is the parameter for the categorical distribution associated with the k-th mixture component (this means that each X_i can take on one of M possible values). We can use the EM algorithm to accomplish this.

A note on notation and a hint: it is helpful to use indicator variables when working with categorical distributions. The indicator function $\mathbb{1}\{x = j\}$ has value 1 when $x = j$ and 0 otherwise. With this notation, we can express the probability that a random variable drawn from a categorical distribution (e.g., $Y \sim$ Categorical(ϕ), where $\phi \in \mathbb{R}^N$) takes on a particular value as

$$P(Y) = \prod_{i=1}^{N} \phi_i^{\mathbb{1}\{Y=i\}}.$$

(a) **[6 points]** What is the joint distribution $P(X, Z; \Theta)$?

(b) **[6 points]** What is the posterior distribution of the latent variables, $P(Z|X; \Theta)$?

(c) **[8 points]** What is the expectation of the log-likelihood, $Q(\Theta'|\Theta) := \mathbb{E}_{Z|X;\Theta} \log P(X, Z; \Theta')$?

(d) **[Extra Credit, 5 points]** What is the update step for Θ? That is, what Θ maximizes $Q(\Theta'|\Theta)$? (Remember that Θ includes the categorical parameter π for Z and the categorical parameters $\theta_1, ..., \theta_K$ for X, from the generative process described in (1). Make sure your solution enforces the constraint that parameters to categorical distributions must sum to 1—Lagrange multipliers are a great way to solve constrained optimization problems.)

Problem 2: AdaBoost

Consider the following dataset, plotted in Figure 1:

$$X_1 = (-1, 0, +), X_2 = (-0.5, 0.5, +), X_3 = (0, 1, -), X_4 = (0.5, 1, -),$$
$$X_5 = (1, 0, +), X_6 = (1, -1, +), X_7 = (0, -1, -), X_8 = (0, 0, -).$$

In this problem, you'll run through $T = 3$ iterations of AdaBoost with decision stumps (axis-aligned half-planes) as weak learners.

(a) **[20 points]** For each iteration $t = 1, 2, 3$, compute $\epsilon_t, \alpha_t, Z_t, D_t(i)\ \forall i$ (in Table 1) and draw the decision stump (on Figure 1). Recall that Z_t is the normalization factor to ensure that the weights D_t sum to one.

(b) **[5 points]** What is the training error of AdaBoost? Give a one-sentence reason for why AdaBoost outperforms a single decision stump.

Table 1: Values of AdaBoost parameters at each timestep.

t	ϵ_t	α_t	Z_t	$D_t(1)$	$D_t(2)$	$D_t(3)$	$D_t(4)$	$D_t(5)$	$D_t(6)$	$D_t(7)$	$D_t(8)$
1											
2											
3											

Problem 3: Perceptron

Consider running the Perceptron algorithm on some sequence of examples S (an example is a data point and its label). Let S' be the same set of examples as S, but presented in a different order.

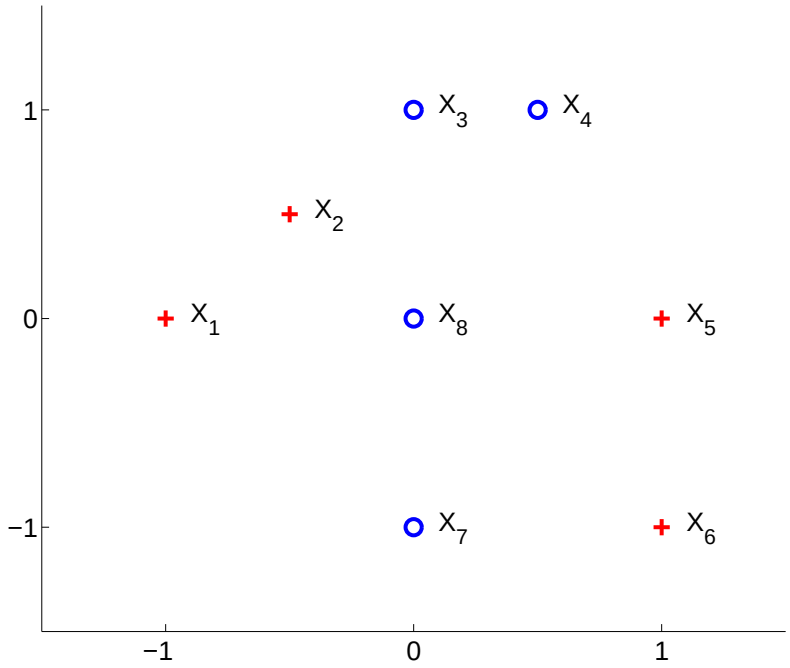

Figure 1: A small dataset, for binary classification with AdaBoost.

(a) **[4 points]** Does the Perceptron algorithm necessarily make the same number of mistakes on S as it does on S'?

(b) **[8 points]** If so, why? If not, show such an S and S' where the Perceptron algorithm makes a different number of mistakes on S' than it does on S.

(c) **[8 points]** We know that in $\mathbb{R}^d$ we can shatter $d+1$ points with linear separators, but not $d+2$ points (because the VC-dimension of linear separators is $d+1$). But what if we require that the points be separated by margin γ? Show that you can have at most $(R/\gamma)^2$ points inside the ball of radius R that can be "shattered at margin γ," meaning that every labeling is achievable by a separator of margin γ. **Hint:** Suppose for contradiction you had $M = (R/\gamma)^2 + 1$ such points. What would happen if you gave them in some order to the Perceptron algorithm, with labels exactly opposite of what Perceptron predicts?

Problem 4: Kernels

4.1: A proposed kernel

Consider the following kernel function:

$$K(x, x') = \begin{cases} 1, & \text{if } x = x' \\ 0, & \text{otherwise} \end{cases}$$

(a) **[8 points]** Prove this is a legal kernel. That is, describe an implicit mapping $\Phi : X \to \mathbb{R}^m$ such that $K(x, x') = \Phi(x) \cdot \Phi(x')$. (You may assume the instance space X is finite.)

(b) **[6 points]** In this kernel space, any labeling of points in X will be linearly separable. Justify this claim.

(c) **[6 points]** Since all labelings are linearly separable, this kernel seems perfect for learning any target function. Why is this actually a bad idea?

4.2: Composition of kernels

It is possible to use previously-defined kernels to construct new ones. For this problem, you will prove why the following proposed functions are or are not valid kernels The notation $< x, z >$ indicates the dot product $x^T z$).

(a) **[3 points]** $K(x, z) = 5 < x, z >$

(b) **[4 points]** $K(x, z) =< x, z >^3 +(< x, z > +1)^2$

(c) **[8 points]** $K(x, z) =< x, z >^2 + \exp(-||x||^2) \exp(-||z||^2)$

Problem 5: Extra Credit

Support Vector Machines

Suppose we have a set $S = \{(x_1, y_1), ..., (x_m, y_m)\}$ of labeled examples in $\mathbb{R}^n$, and assume $||x_i|| = 1, \forall i$. It is NP-hard to find a linear separator that minimizes the number of points misclassified, so learning algorithms optimize other related quantities that are easier to solve. SVM solves the optimization problem

$$\begin{aligned} \min_{w} \quad & ||w||_2^2 + C \sum_{i=1}^{m} \xi_i \\ \text{subject to} \quad & y_i(w^T x_i) \geq 1 - \xi_i \quad \forall i \\ & \xi_i \geq 0 \quad \forall i \end{aligned} \tag{2}$$

(a) **[5 points]** Suppose that S has the property that the total distance one would need to move the points in order to make them separable by margin γ is d_γ. (By "total distance" we mean $\sum_i d_i$ where d_i is the distance that point x_i is moved. By "separable by margin γ" we mean that, for some hyperplane through the origin, all points are on the correct side and at distance at least γ from it.) Show that, for an appropriate value of C, the number of misclassifications made on S by the separator produced by SVM is at most $\frac{1}{2} + d_\gamma/\gamma$.
Hint: We can find the value of d_γ with the following optimization problem:

$$\begin{aligned} \min_{w, \tilde{\gamma}_i, \tilde{\gamma}} \quad & \sum_{i=1}^{m} \tilde{\gamma}_i \\ \text{subject to} \quad & ||w|| = 1 \\ & y_i(w^T x_i) \geq \tilde{\gamma} - \tilde{\gamma}_i \quad \forall i \end{aligned} \tag{3}$$

where γ_i indicates the distance between point x_i and the appropriate margin.

Boosting

Suppose that, instead of computing $\alpha_t = \frac{1}{2} \ln\left(\frac{1-\epsilon_t}{\epsilon_t}\right)$, we instead fixed $\alpha > 0$ ahead of time and set $\alpha_t = \alpha$ for each iteration t in this Boost(α) algorithm. Assume that on every round t of boosting, we know that ϵ_t will be at most $1/2 - \gamma$, for some $\gamma > 0$. We choose to set $\alpha = \frac{1}{2} \ln\left(\frac{1+2\gamma}{1-2\gamma}\right)$.

(a) **[5 points]** Show how to modify the training error analysis of AdaBoost to derive an upper bound on the training error of the final hypothesis H produced by Boost(α) after T rounds. Your bound should be in terms of γ and T only (it should not depend on α,ϵ_t, etc.).
Hint: Begin by proving, as we did in class, the bound on the error $\text{err}_S(H_{\text{final}})$ in terms of the normalization factors Z_t.

(b) **[5 points]** Use the previous result to show that the final hypothesis H will be consistent with m training examples (i.e., have zero training error) after T rounds if $T > \frac{\ln m}{2\gamma^2}$.

(c) **[5 points]** Assume that the weak learning algorithm generates hypotheses h_t which belong to a finite class $\mathcal{H}$. Use error bounds from the class to show that if we choose T as in part (b), then with probability $1 - \delta$, the generalization error of H is at most $\frac{T \ln |\mathcal{H}| + \ln(1/\delta)}{m}$.