

# ICS273A: HW4

Due: March 5

**Problem I:** Learning theory. One of the basic PAC assumptions is that the training and test data pairs  $(x, y)$  are generated IID from some distribution  $D = p(x, y)$ . In this problem, we'll consider a case where the training and test distributions are different. Specifically, we'll look at the case where the training labels  $y$  are noisy, but the test labels are not.

Let  $y \in 0, 1$  and  $D$  be the clean, uncorrupted distribution. Define  $D_\tau$  to be a corrupted distribution over  $(x, y)$ , which is the same as  $D$ , except that the labels  $y$  have some probability  $0 \leq \tau \leq 0.5$  of being flipped. Thus, to sample from  $D_\tau$ , we would first sample  $(x, y)$  from  $D$ , and then with probability  $\tau$  (independent of the observed  $(x, y)$ ), replace  $y$  with  $1 - y$ . Thus  $D_0 = D$ .

The distribution  $D_\tau$  captures the setting in which a human is labeling the training data, but has a probability  $\tau$  of mislabeling a particular example. Ultimately, we are interested in evaluating a hypothesis  $h$  with respect to the original, uncorrupted distribution  $D$ .

We define the generalization error with respect to  $D_\tau$  to be

$$\varepsilon_\tau(h) = P_{x,y \sim D_\tau}[h(x) \neq y]$$

Note that  $\varepsilon_\tau(h)$  is the generalization error with respect to the original distribution; this is error we would like to minimize.

- A For any hypothesis  $h$ ,  $\varepsilon_0(h)$  can be calculated as a function of  $\varepsilon_\tau(h)$  and  $\tau$ . Derive a formula for  $\varepsilon_0(h)$  in terms of  $\varepsilon_\tau(h)$  and  $\tau$ .
- B Let  $|H|$  be finite, and suppose our training set  $S = \{(x^{(i)}, y^{(i)})\}$  is obtained by drawing  $m$  IID samples from the corrupted distribution  $D_\tau$ . Suppose we pick  $h \in H$  that minimizes the empirical risk, or  $\hat{h} = \arg \min_{h \in H} \hat{\varepsilon}_S(h)$ . We have written  $\hat{\varepsilon}_S(h)$  for the empirical risk measured on the dataset  $S$ . Also let  $h^* = \arg \min_{h \in H} \varepsilon_0(h)$ . Let any  $\delta, \gamma$  be given. Prove that for

$$\varepsilon_0(\hat{h}) \leq \varepsilon_0(h^*) + 2\gamma$$

to hold with probability at least  $1 - \delta$ , it suffices that

$$m \geq \frac{1}{2(1 - 2\tau)^2 \gamma^2} \log \frac{2|H|}{\delta}$$

Note. This suggests that roughly  $m$  examples that have been corrupted at a noise level of  $\tau$  are worth about as much as  $(1 - 2\tau)^2 m$  uncorrupted training examples.

C What happens as  $\tau$  approaches .5?

**Problem II: Kernels.** Consider a set of vectors  $S = \{x^{(1)}, \dots, x^{(m)}\}$ . Let  $k(x, y) = \phi(x)^T \phi(y)$ , the inner product in some feature space. Let  $K$  be a matrix where the  $i, j$  element is  $k(x^{(i)}, x^{(j)})$ . Let  $C$  denote the center of mass of the vectors in feature space:  $C = \frac{1}{m} \sum_{i=1}^m \phi(x_i)$ .

A Show that one can compute the distance  $\|\phi(x^{(i)}) - \phi(x^{(j)})\|^2$  using  $K$ .

B Show how to compute the squared norm of the center vector  $\|c\|^2$  using  $K$ .

C Suppose we define a new feature space  $\phi_c(x) = \phi(x) - C$ . Show that the center of mass of the points in this space is at the origin.

D Let  $K_c$  be the kernel matrix corresponding to the mapping  $\phi_c(x)$ . Write  $K_c$  in terms of  $K$ .

**Problem III: [MATLAB] SVMs.** We will be using a quadratic programming package to train a hard-margin kernel SVM. We will be using the classification data from hw4Train.dat and hw4Test.dat. We will be using MATLAB's `quadprog` function to optimize the dual formation.

A To use the optimization packages, we need to write the dual hard-margin SVM formulation as minimizing a quadratic function  $\frac{1}{2}x^T Hx + f^T x$  subject to the single linear inequality  $Ax \leq b$  and the single linear equality  $A_{eq}x = b_{eq}$ . Explicitly define the  $H, f, A, b, A_{eq}, b_{eq}, x$  in terms of our variables  $\{\alpha_i, x^i, y^i\}$ .

B Attempt to train a linear hard-margin SVM from hw4Train.dat using the above formulation. Explain why this cannot be done.

C Use a Gaussian kernel  $k_\lambda(x, y) = \exp(-\frac{1}{\lambda}\|x - y\|^2)$  with  $\lambda = .1$ . Visualize the decision boundary by plotting the test points with an 'x' if its predicted to be 0, and 'o' if its predicted to be 1. Compute the test error rate.

**Problem IV: [MATLAB] Face recognition and dimensionality reduction.** We will build a face recognizer using faceTrain.mat and faceTest.mat with a nearest neighbor classifier. We will use principle component analysis (PCA) to reduce the dimensionality of the raw image features. This face recognition algorithm is known as eigenfaces, and is competitive with state-of-the-art systems. We will also experiment with performing nonlinear component analysis with kernel PCA.

- A You can visualize the faces using the same code visualizing digit images. There are 40 different people in this dataset, so this is a multiclass classification task with 40 classes. Construct a  $k$ -nearest neighbor classifier using euclidean distance and score the miss-classification rate on the test data. Select the optimal  $k$  by cross-validation. When  $k$  is even, you may need to break ties randomly at test time. This will be our baseline algorithm.
- B Reduce the dimensionality of the training data by performing PCA. Because the dimensionality of the data is quite high, use the matlab function `svds` to compute the eigenvectors iteratively. At first, project the data down to 10 dimensions. Visualize the first, second, and third principle components as images.
- C Pick a particular face from the training set and visualize its *reconstruction* using 1, 2, and 10 principle components.
- D Perform classification on the test data by also projecting the test data into the 10 dimensional space, and then performing nearest neighbor classification. Experiment with the optimal number of dimensions through cross-validation.
- E Use kernel-PCA to perform the dimensionality reduction. Recall that PCA, in one dimensions, computes the direction  $v$  that maximizes the variance of a linear projection  $v^T x^{(i)}$ . We derived in class that this can be solved with an eigenvector problem  $\frac{1}{m} X^T X v = \lambda v$ , where  $X$  is the design matrix for centered datapoints  $x^{(i)}$ . Assume that the vector  $v$  lies in the span of the data, or  $v = \sum_i \alpha_i x^{(i)}$ . We can compute the same direction in terms of the vector of coefficients  $\alpha$ . Show that this yields an eigenvector problem of the form  $XX^T \alpha = \tilde{\lambda} \alpha$ . What is  $\tilde{\lambda}$  in terms of  $\lambda$ ?
- F (i) Show how the above “ $\alpha$ -based” version of PCA can be kernelized, so one that is computing the spanning coefficients of a vector  $v$  in the feature space  $\phi(x)$ . (ii) Show how to compute the projection of data point  $x^{new}$  onto the principle component vector in the feature space.

Extra credit Implement kernel PCA for your eigenface classifier. Note that the data needs to be centered in the feature space. You derived the expression for centering a kernel matrix in Problem II. Project down to 1,2, and 5 dimensions in the feature space and build a nearest neighbor classifier. Does the nonlinear dimensionality reduction reduce the test error?