CSE 546 Machine Learning, Autumn 2013
Homework 4

Due: Monday, December 2, beginning of class

1 Learning Theory [30 points]

Consider a noise-free binary classification problem for which we learn a function that maps binary feature vectors of length d, $\mathbf{x} \in \{0,1\}^d$, to output values $y \in \{0,1\}$. We are given N noise-free, i.i.d. training examples $(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_N, y_N)$.

1. *[8 points]* Suppose our learning method makes no restrictions on the hypothesis space. That is, we potentially learn any classification function h that satisfies

$$h : \{0,1\}^d \rightarrow \{0,1\}$$

   For this case, what is the size of our hypothesis space?
2. *[4 points]* Using a PAC bound, if we would like to guarantee a generalization error less than ϵ with high probability $1-\delta$, how many samples must we observe?
3. *[4 points]* Interpret the bound you derived above. Is it useful? Why or why not?
4. *[10 points]* Now assume the true relationship between $\mathbf{x}$ and y can be expressed by a binary decision tree of depth 2. If we learn a decision tree with depth 2, what is the size of the hypothesis space? Provide a bound on the number of samples N required to achieve generalization error ϵ with probability $1-\delta$. (Note: you do not need to calculate the size of the hypothesis space exactly. Answers using Big-Oh notation are acceptable.)
5. *[4 points]* How does this bound compare to your initial bound? Comment on the structure we assumed for this problem and its effect on our PAC bounds.

2 PCA via Successive Deflation [35 points]

(Adapted from Murphy Exercise 12.7)

Suppose we have a set of n datapoints $x_1, \ldots, x_n$, where each x_i is represented as a d-dimensional column vector.

Let $\mathbf{X} = [x_1; \ldots; x_n]$ be the $(d \times n)$ matrix where column i is equal to x_i. Define $\mathbf{C} = \frac{1}{n}\mathbf{X}\mathbf{X}^T$ to be the covariance matrix of $\mathbf{X}$, where $\mathbf{C}_{ij} = \sum_n \mathbf{X}_{in}\mathbf{X}_{jn} = covar(i,j)$.

Next, let $v_1, v_2, \ldots v_k$ be the first k eigenvectors with larges eigenvalues of $\mathbf{C}$, i.e., the principal basis vectors. These satisfy

$$v_j^T v_k = \begin{cases} 0 & \text{if } j \neq k \\ 1 & \text{if } j = k \end{cases}$$

v_1 is the first principal eigenvector of $\mathbf{C}$ (the eigenvector with the largest eigenvalue), and as such satisfies $\mathbf{C}v_1 = \lambda_1 v_1$. Now define $\tilde{x}_i$ as the orthogonal projection of x_i onto the space orthogonal to v_1:

$$\tilde{x}_i = (\mathbf{I} - v_1 v_1^T) x_i$$

Finally, define $\tilde{\mathbf{X}} = [\tilde{x}_1; \ldots; \tilde{x}_n]$ as the **deflated matrix** of rankd $d-1$, which is obtained by removing from the d-dimensional data the component that lies in the direction of the first principal eigenvector:

$$\tilde{\mathbf{X}} = (\mathbf{I} - v_1 v_1^T)\mathbf{X}$$

1. *[11 points]* Show that the covariance of the deflated matrix, $\tilde{\mathbf{C}} = \frac{1}{n}\tilde{\mathbf{X}}\tilde{\mathbf{X}}^T$ is given by

$$\tilde{\mathbf{C}} = \frac{1}{n}\mathbf{X}\mathbf{X}^T - \lambda_1 v_1 v_1^T$$

   (Hint: Some useful facts: $(\mathbf{I} - v_1 v_1^T)$ is symmetric, $\mathbf{X}\mathbf{X}^T v_1 = n\lambda_1 v_1$, and $v_1^T v_1 = 1$. Also, for any matrices A and B, $(AB)^T = B^T A^T$.)
2. *[11 points]* Show that for $j \neq 1$, if v_j is a principal eigenvector of $\mathbf{C}$ with corresponding eigenvalue λ_j *(that is, $\mathbf{C}v_j = \lambda_j v_j$)*, then v_j is also a principal eigenvector of $\tilde{\mathbf{C}}$ with the same eigenvalue λ_j.
3. *[6 points]* Let u be the first principal eigenvector of $\tilde{\mathbf{C}}$. Explain why $u = v_2$. (You may assume u is unit norm.)
4. *[7 points]* Suppose we have a simple method for finding the leading eigenvector and eigenvalue of a positive-definite matrix, denoted by $[\lambda, u] = f(\mathbf{C})$. Write some pseudocode for finding the first K principal basis vectors of $\mathbf{X}$ that only uses the special f function and simple vector arithmetic.
   (Hint: This should be a simple iterative routine that takes 2-3 lines to write. The input is $\mathbf{C}, K$, and the function f, the output should be v_j and λ_j for $j \in [1:K]$.)

3 Programming Question (clustering with K-means) [35 points]

In class we discussed the K-means clustering algorithm. Your programming assignment this week is to implement the K-means algorithm on digit data.

3.1 The Data

There are two files with the data. The first

`digit.txt`

contains the 1000 observations of 157 pixels (a subset of the original 785) concerning handwritten digits. The second file

`labels.txt`

contains the true digit label (either 1, 3, 5, or 7). You can read both data files in with

```
X = genfromtxt('digit.txt')
Y = genfromtxt('labels.txt', dtype=int)
```

Please note that there aren't IDs for the digits. Please assume the first line is ID 0, the second line is ID 1, and so on. The labels correspond to the digit file, so the first line of labels.txt is the label for the digit in the first line of digit.txt.

3.2 The algorithm

Your algorithm should be implemented as follows:

1. Select k starting centers that are points from your data set. You should be able to select these centers randomly or have them given as a parameter.
2. Assign each data point to the cluster associated with the nearest of the k center points.
3. Re-calculate the centers as the mean vector of each cluster from (2).
4. Repeat steps (2) and (3) until convergence or iteration limit.

Define convergence as no change in label assignment from one step to another **or** you have iterated 20 times (whichever comes first). Please count your iterations appropriately: after 20 iterations, you should have re-calculated the centers 20 times.

3.3 Within group sum of squares

The goal of clustering can be thought of as minimizing the variation within groups and consequently maximizing the variation between groups. A good model has low sum of squares within each group. We define sum of squares in the traditional way. Let C_k be the kth cluster and let μ_k be the empirical mean of the observations x_i in cluster C_k. Then the within group sum of squares for cluster C_k is defined as:

$$SS(k) = \sum_{i \in C_k} |x_i - \mu_{C_k}|^2$$

Please note that the term $|x_i - \mu_{C_k}|$ is the euclidean distance between x_i and μ_{C_k}, and therefore should be calculated as $|x_i - \mu_{C_k}| = \sqrt{\sum_{j=1}^{d} (x_{ij} - \mu_{C_{kj}})^2}$, where d is the number of dimensions. Please note that that term is squared in $SS(k)$. If there are K clusters total then the "sum of within group sum of squares" is just the sum of all K of these individual $SS(k)$ terms. .

3.4 Mistake Rate

Given that we know the actual assignment labels for each data point we can attempt to analyze how well the clustering recovered this. For cluster C_k let its assignment be whatever the majority vote is for that cluster. If there is a tie, just choose the digit that is smaller numerically as the majority vote. For example if for one cluster we had 270 observations labeled **one**, 50 labeled **three**, 9 labeled **five**, and 0 labeled **seven** then that cluster will be assigned value **one** and had $50 + 9 + 0 = 59$ mistakes. If we add up the total number of "mistakes" for each cluster and divide by the total number of observations (1000) we will get our total mistake rate, between 0 and 1.

3.5 Questions

When you have implemented the algorithm please report the following:

1. [10pts] The values of sum of within group sum of squares and mistake rates for $k = 2$, $k = 4$ and $k = 6$. Please start your centers with the first k points in the dataset. So, if $k = 2$, your initial centroids will be ID 0 and ID 1, which correspond to the first two lines in the file.
2. [5pts] The number of iterations that k-means ran for $k = 6$, starting the centers as in the previous item. Make sure you count the iterations correctly. If you start with iteration $i = 0$ and at $i = 3$ the the cluster assignments don't change, the number of iterations was 4, as you had to do steps 2 and 3 to figure this out.
3. [10pts] A plot of the sum of within group sum of squares versus k for $k = 1, 2, 3, 4, 5, 6, 7, 8, 9, 10$. Please start your centers randomly (choose k points from the dataset at random).
4. [10pts] A plot of total mistake rate versus k for $k = 1, 2, 3, 4, 5, 6, 7, 8, 9, 10$. Please start your centers randomly (choose k points from the dataset at random).

For the last two items, you should generate these graphs about 3 times, just to make sure you don't submit a plot where k-means got really unlucky centers in the beginning. Only submit one plot though. Also remember to submit your code.

3.6 Some useful functions

You should take a look at the "euclidean" function in "scipy.spatial.distance". You should also take a look at smart way of indexing, such as the "fancy stuff" in the python review.