CSCI567 Fall 2013 - Assignment 5
Due date/time: Wed., Nov. 26, 2013, 3:15pm

November 12, 2013

Algorithm Component

1 Clustering

Q1. (2 points) In the lectures, we discussed K-means. Given a set of data points $\{\boldsymbol{x}_n\}_{n=1}^N$, the method minimizes the following distortion measure (or objective or clustering cost)

$$D = \sum_{n=1}^{N} \sum_{k=1}^{K} r_{nk} \|\boldsymbol{x}_n - \boldsymbol{\mu}_k\|_2^2$$

where $\boldsymbol{\mu}_k$ is the prototype of the k-th cluster. r_{nk} is a binary indicator variable. If $\boldsymbol{x}_n$ is assigned to the cluster k, r_{nk} is 1 otherwise r_{nk} is 0.

For each cluster, $\boldsymbol{\mu}_k$ represents all the data points being assigned to that cluster. In the lecture, we showed but did not prove that, $\boldsymbol{\mu}_k$ is the mean of all such data points. That is why the method has MEANS in its name and we keep referring to $\boldsymbol{\mu}_k$ as MEANS, CENTROIDS, etc. You will prove this rigorously next.

Specifically, assuming all r_{nk} are known (that is, you know the assignments of all N data points), show that if $\boldsymbol{\mu}_k$ is the mean of all data points assigned to the cluster k, for any k, then the objective D is minimized. This justifies the iterative procedure of K-means[1].

Q2. (5 points) We now change the distortion measure to

$$D = \sum_{n=1}^{N} \sum_{k=1}^{K} r_{nk} |\boldsymbol{x}_n - \boldsymbol{\mu}_k|_1$$

In other words, the measurement of "closeness" to the cluster prototype is now using L_1 norm ($|\boldsymbol{z}|_1 = \sum_d |z_d|$).

Under this new cost function, show the $\boldsymbol{\mu}_k$ that minimizes D can be interpreted as the elementwise median of all data points assigned to the k-th cluster. (The elementwise median of a set of vectors are defined as a vector whose d-th element is the median of all vectors' d-th elements.)

[1] More rigorously, one would also need to show that if all $\boldsymbol{\mu}_k$ are known, then r_{nk} can be computed by assigning $\boldsymbol{x}_n$ to the nearest $\boldsymbol{\mu}_k$. You are not required to do so.

Furthermore, sketch an iterative algorithm, analogous to the K-means clustering, to minimize this cost function.

Q3. (2 points) Consider a Gaussian mixture model with the hidden variable $z \in \{1, 2, \ldots, K\}$. The model is given by the following conditional probability

$$p(\boldsymbol{x}|z) = \frac{1}{\sqrt{(2\pi)^D |\boldsymbol{\Sigma}_k|}} e^{-\frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu}_k)^{\mathrm{T}} \boldsymbol{\Sigma}_k^{-1} (\boldsymbol{x}-\boldsymbol{\mu}_k)}$$

for the observed data $\boldsymbol{x} \in \mathbb{R}^D$, and the mixture weight $\omega_k = p(z = k)$.

In using Gaussian mixture models for modeling data, an important quantity is the posterior distribution of z

$$p(z = k|\boldsymbol{x})$$

for all k, which are then used by the EM procedure to re-estimate parameters of the Gaussians $\boldsymbol{\mu}_k$ and $\boldsymbol{\Sigma}_k$.

Prove that if $\boldsymbol{\Sigma}_k = \sigma^2 \boldsymbol{I}$ for all k, where σ is a scalar and $\boldsymbol{I}$ is the identity matrix, then

$$\lim_{\sigma \to 0} p(z = k|\boldsymbol{x}) = \begin{cases} 1 & \text{if } k = \arg\min_j \|\boldsymbol{x} - \boldsymbol{\mu}_j\| \\ 0 & \text{others} \end{cases}$$

The above result links the EM algorithm for Gaussian mixture model

2 Gaussian mixture models and EM Algorithm

Q4. Gaussian mixture models (10 points)

The Gaussian mixture model is defined as

$$p(\boldsymbol{x}) = \sum_{k=1}^{K} \omega_k \mathcal{N}(\boldsymbol{x}; \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$$

In the class, I derived the Q-function, which is the expectation of the complete likehood

$$Q(\boldsymbol{\theta}, \boldsymbol{\theta}^{old}) = \sum_n \sum_{\boldsymbol{z}_n} p(\boldsymbol{z}_n|\boldsymbol{x}_n, \boldsymbol{\theta}^{old}) \log p(\boldsymbol{x}_n, \boldsymbol{z}_n|\boldsymbol{\theta})$$

where $\boldsymbol{z}_n$ is the hidden variable corresponding to the observed data $\boldsymbol{z}_n$. For Gaussian mixture model, this Q-function becomes

$$Q(\boldsymbol{\theta}, \boldsymbol{\theta}^{old}) = \sum_n \sum_k \gamma_{nk} \log p(\boldsymbol{x}_n|z_n = k, \boldsymbol{\theta}) p(z_n = k|\boldsymbol{\theta})$$

where z_n is the "color" (i.e., the mixture component that $\boldsymbol{x}_n$ comes from), γ_{nk} is the responsibility (i.e., the posterior probability of $\boldsymbol{x}_n$ comes from $z_n = k$ under the old parameters). $\boldsymbol{\theta}$ is then all the means and covariance matrices, as well as ω_k. $p(\boldsymbol{x}_n|z_n = k, \boldsymbol{\theta})$ is the k-th Gaussian distribution and $p(z_n = k|\boldsymbol{\theta})$ is the mixture weight ω_k.

Computing the Q-function is called E-step in the EM algorithm. The M-step then computes the new parameters that maximize the Q-function, which gives rise to updated estimates of the parameters.

Prove by maximizing the Q-function that the new parameters are

$$\begin{aligned}
\omega_k &\leftarrow \frac{N_k}{\sum_k N_k} \\
\boldsymbol{\mu}_k &\leftarrow \frac{1}{N_k} \sum_n \gamma_{nk} \boldsymbol{x}_n \\
\boldsymbol{\Sigma}_k &\leftarrow \frac{1}{N_k} \sum_n \gamma_{nk} (\boldsymbol{x}_n - \boldsymbol{\mu}_k)(\boldsymbol{x}_n - \boldsymbol{\mu}_k)^{\mathrm{T}}
\end{aligned} \tag{1}$$

where $N_k = \sum_n \gamma_{nk}$. Note that, in deriving $\boldsymbol{\Sigma}_k$, you will need the following equality

$$\frac{\partial}{\partial \boldsymbol{A}} \log |\boldsymbol{A}| = (\boldsymbol{A}^{-1})^{\mathrm{T}}$$

where $\boldsymbol{A}$ is a matrix.

Q5. Guassian mixture models with special structures (5 pts)

You are now asked to develop an algorithm for fitting data to a special type of Gaussian mixture model. Specifically, the model takes the form

$$p(\boldsymbol{x}) = \sum_{k=1}^{K} \omega_k \mathcal{N}(\boldsymbol{x}; \boldsymbol{\mu}_k, \boldsymbol{\Sigma})$$

where $\boldsymbol{\Sigma}$ is the covariance matrix shared by all the Gaussian mixture components. Show how the parameters should be updated in the M-Step.

Programming Component

You will be experimenting K-means in addition to other methods. You are free to use either others' implementation or your own implementation of K-means.

1. (10 points) In the Question 2 of the Algorithm Component of this assignment, you have derived a new clustering algorithm using the new cost function defined with L_1 norm. In this question, you will *generate* some data (this is often called synthetic data) to demonstrate the advantage of the new cost function.

Specifically, using L_1 norm has the benefit of being less sensitive to outliers — we mentioned this when we discussed about linear regression and stated that squared errors can amplify errors that are often large for outliers.

How to demonstrate the new cost function is meritorious? On a high-level, since the new cost function claims it can deal with outliers better, we should make up a dataset where there are outliers to highlight the advantage of the new cost function. With that in mind, we will do the following:

- Create a data set that has 90 two-dimensional data points that can be clustered into 3 clusters. We can do this by randomly sampling from 3 Gaussian distributions that are separated from each other. For each cluster, we make sure that these data points are "tightly" centering around the means. (You will find Matlab's function randn() is especially useful for this kind of purpose. You should use Matlab's plotting function scatter() to visualize where data points are placed. In particular, for each cluster, plot the points in that cluster with a different color.)
- Now, add 10 points to each cluster. We are free to add but we should add them in such a way that after visualizing all data points, we can point to those newly added points and say "Hmm.. they kind of belong to the 3 clusters but they seem a bit far from the means/centroids... Ie. they are outliers".
- With 120 points at our disposal, we are ready to "prove" we are doing a smart thing in proposing the new cost function
    1. We run the K-means clustering algorithm on the 90 data points. We should get centroids/cluster centers that are very close to the means of the 3 Gaussian distributions.
    2. We run the K-means clustering algorithm on the 120 data points. Now since the squared distances used in the K-means algorithm are more sensitive to outliers, the centroids will be shifted significantly.
    3. We run the new algorithm on the 90 data points. We should get centroids/cluster centers that are very close to the means of the 3 Gaussian distributions.
    4. We run the new algorithm on the 120 data points. We should get centroids/cluster centers that are shifted away from the means of the 3 Gaussian distributions, but *not as badly as* the original K-means algorithm. We declare *Success* as our algorithm robustly fights back the adverse effect of the outliers.

Report Following the above experimental design and obtaining results, you should report

- Plot 1. The 90 data points and the 30 outliers. Mark the 90 data point with $\circ$ markers, and fill them with the gray color, and the outliers with $\diamond$ markers and fill them with the black color. Mark the locations of the 3 means of the Gaussian distributions with $\times$ markers in the green color.

- Plot 2. (K-means results on 90 data points) Same as Plot 1 but with 30 outliers removed. Fill the markers with solids colors: red, blue, black, for 3 clusters respectively. Note that you should color each data point according to the cluster they are *assigned* to. Mark the means of the clusters identified by the K-means algorithm on the 90 data points with □ markers (do not fill them with colors). The plot should be titled as "K-means results on 90 data points", also in the caption, you should report the Euclidean distances from the Gaussians' means to the means of the clustering.

- Plot 3. (K-means results on 120 data points) Same as Plot 1. Fill the markers with solids colors: red, blue, black, for 3 clusters respectively. Note that you should color each data point according to the cluster they are *assigned* to. Mark the means of the clusters identified by the K-means algorithm on the 120 data points with □ markers (do not fill them with colors). The plot should be titled as "K-means results on 120 data points", also in the caption, you should report the Euclidean distances from the Gaussians' means to the means of the clustering.

- Plot 4. (New clustering results on 90 data points) Same as Plot 1 but with 30 outliers removed. Fill the markers with solids colors: red, blue, black, for 3 clusters respectively. Note that you should color each data point according to the cluster they are *assigned* to. Mark the means of the clusters identified by the new clustering algorithm on the 90 data points with □ markers. The plot should be titled as "New clustering results on 90 data points", also in the caption, you should report the Euclidean distances from the Gaussians' means to the means of the clustering.

- Plot 5. (New clustering results on 120 data points) Same as Plot 1. Fill the markers with solids colors: red, blue, black, for 3 clusters respectively. Note that you should color each data point according to the cluster they are *assigned* to. Mark the means of the clusters identified by the new clustering algorithm on the 120 data points with □ markers. The plot should be titled as "New clustering results on 120 data points", also in the caption, you should report the Euclidean distances from the Gaussians' means to the means of the clustering.

You should also comment on whether empirical studies have supported the claim that the new method is better in terms of resisting outliers. In particularly, the distances in Plot 5 should be smaller than the distances in Plot 3. The distances in Plot 4 should be about the same as distances in Plot 2. What the relationships between the distances in Plot 5 and Plot 4?

Submission

- Submit your plots/writeup in a single PDF file.

- Submit your data in a Matlab file called clustering.mat. The file should contains the following variables: i) good data: it should be a 2×90 matrix, storing the 90 data points; ii) outliers: it should be a 2×30 matrix, storing the outliers; iii) GaussianMeans: it should be a 2×3 matrix, storing the Gaussian means; iv) clustering90means: it should be a 2×3 matrix, storing the cluster means, computed using K-means on 90 data points; v) clustering120means: it should be a 2×3 matrix, storing the cluster means, computed using K-means on 120 data points; vi) newclustering90means: it should be a 2×3 matrix, storing the cluster means, computed using the new method on 90 data points; vii) newclustering120means: it should be a 2×3 matrix, storing the cluster means, computed using the new method on 120 data points.

- Submit your code for the new clustering algorithm. Name your file **newclustering.m**. It should be a Matlab function such that when we type newclustering(Data, K), it would give us back the clustering results (i.e., means) of running the algorithm on the Data, assuming K clusters.

2. (5 points) Let us have some fun! Use your favorite device (mobile phones, cameras etc) to take a picture of your face. That picture is your data to use in this experiment.

Make sure your picture is colored, has a size of 200 times 200 pixels. You can convert this picture with Matlab's image processing functions as a matrix of 200×200 where each element has 3 RGB values (this is called an RGB image).

In the lecture, I demonstrate the K-means clustering as a technique for vector quantization. You will experiment this. Specifically, for $K = 10, K = 100, K = 200, K = 400$, you will run the clustering algorithm to cluster those 40000 pixels, where each pixel is a 3-dimensional vector. (You need to make sure that each dimension's value ranges from 0 to 1. If they are not, you can scale or linearly transform them from the existing range to between 0 and 1.). Then for each pixel, replace the original RGB values with the mean of the cluster to which it is assigned. This will give you back an image with lost details.

Report You should report the following

- The original image.
- The 4 images as a result of vector quantization (lossy compression).

Submission You should submit all the 5 images in a single PDF file. Clearly mark which image is the original one and the K for the other images. Note that, qualitatively, the larger K is the more similar the image is as the original one.

3 PCA for Preprocessing (25 points)

We will use the handwritten digit dataset from Homework 2. You will be using the task to understand the effect of PCA as an important preprocessing method.

Important! Read This! When you run PCA, you should always use the training data only to compute the mean, the covariance matrix, and then the projection matrix. The validation/test data should NEVER be used to compute these quantities. If you do, you are "cheating" and your results will be skewed.

For this reason, do not mix the validation and training data.

Once the mean/projection matrix computed, they can be applied to the validation/test data to obtain low-dimensional coordinates. Do not forget to subtract the mean from the validation/test images. Otherwise, you will get very bad results.

Experiment procedures You will first compute the mean, the covariance matrix and the projection matrix with PCA. Suppose your projection matrix is $\boldsymbol{U} \in \mathbb{R}^{D \times M}$ where D is the dimensionality of the original data (in this case $D = 256$) and M is the dimensionality of the subspace you need to project to.

The low-dimensional coordinates for a data point $\boldsymbol{x} \in \mathbb{R}^D$ would be

$$\boldsymbol{y} = \boldsymbol{U}^T(\boldsymbol{x} - \boldsymbol{x}_0)$$

where $\boldsymbol{x}_0$ is the mean of the training data.

- We will experiment different M: 5, 20, 40, 60 and 100. (You should always choose the top M eigenvalues and eigenvectors to compose $\boldsymbol{U}$).

- The $\boldsymbol{y}$ computed above is called "diagonalized" data. We will also experiment whitening the data, ie, trying to make the data has an identity matrix as its covariance matrix. The whitening step is

$$\boldsymbol{y} = \boldsymbol{\Lambda}^{-1/2}\boldsymbol{U}^T\boldsymbol{x}$$

  where $\boldsymbol{\Lambda}$ is the diagonal matrix of eigenvalues as diagonal elements.

- Once $\boldsymbol{y}$ is computed, we will use it for classification. Note that, the training, validation and testing data all will be transformed into the corresponding $\boldsymbol{y}$ and we build classifiers and test them in the space of $\mathbb{R}^M$.

  Classifiers You should use only support vector machines as classifiers in this homework. Your SVM should use Gaussian RBF kernels. For each different M, you should use the 5-fold cross-validation data to choose the best regularization coefficients (C) and the bandwidths of Gaussian RBF kernels.

3.1 Learn to use SVM toolbox

The libSVM package, available at http://www.csie.ntu.edu.tw/∼cjlin/libsvm/ is a well known and widely used package for SVM. It supports both binary and multiclass classification in various variants of the SVM problem. You are required to experiment with report classification results using libSVM. This part of the assignment does not involve any numerical optimization your behalf. You should:

- **Download** the Matlab compatible source files available at:http://www.csie.ntu.edu.tw/∼cjlin/libsvm/
- **Installation and Preparation** - The archive file you download will contain a 'matlab' folder. Carefully **READ the README file** within that folder.
  - The installation is operating system dependent and required a C compiler. (use 'mex -setup' in your matlab environment to see if you have it configured.)
    - On a linux\mac you should already have gcc installed.
    - On windows based machine - There are many available compilers. You may want to look for a version of Microsoft's visual studio express.
  - Note the **'Examples'** section in the README file. It contains a short code snippet that shows how to train a model and then use if for classification.
- **Read documentation!** Libsvm is flexible: it implements several types of SVM (you should choose the SVM implementation of hinge-loss, and L2-regularized weight parameters).

3.2 Writeup Requirement

Submission to Blackboard: Predictions. What you need to submit is a Matlab file with a variable which is a 2000-dimensional column vector, recording the predictions of your algorithm on the test data, using the best combination of preprocessing and classifier. Name the variable **pca_diagonize_prediction** and **pca_whiten_prediction**, respectively, one for each procedure of preprocessing data. Name the Matlab file **hw5_prediction.mat**.

Submission to Blackboard: Writeup report in PDF format. (a) for each preprocessing procedure, the optimal M^* as well as its corresponding optimal C and the Gaussian kernel bandwidth. (b) the plots of accuracies on the test data (vertical axis: accuracy. horizontal axis: M. The M^* should be in the plots so we can clearly see they give the best accuracies on the test dataset.).

3.3 Contrast to another preprocessing method (15 points)

Another popular method for preprocessing data is to standardize. Specifically, for every feature dimension d, we compute the mean and standard deviation of this feature in the *training data*

$$a_d = \frac{\sum_{n=1}^{N} x_{nd}}{N}, \quad b_d = \sqrt{\frac{\sum_{n=1}^{N}(x_{nd} - a_d)^2}{N-1}}$$

where x_{nd} is the d-th dimension of the n-th training sample, and there are N training samples. Once a_d and b_d are computed, we transform any data $\boldsymbol{x}$ with

$$y_d = \frac{x - a_d}{b_d}$$

for every dimension. The transformation is applied to the training, validation and test data. The classifiers will then be built using $\boldsymbol{y}$, just like in PCA. However, unlike in PCA where you choose a M to be smaller than D, we will use all dimensions.

But you do need to choose the best C and the bandwidth for the Gaussian RBF kernel, when you construct the classifier.

Submission to Blackboard: Predictions. What you need to submit is the same Matlab file **hw5_prediction.mat** with another variable. The variable is a 2000-dimensional column vector, recording the predictions of your algorithm on the test data, by your best performing model on the cross-validation. Name the variable **standard_prediction**.

Submission to Blackboard: Writeup report in PDF format. State (a) the optimal C and the bandwidth of your SVM classifier; (b) the accuracy of your best performing model on the test data. (c) Which performs the best, the standardization procedure or your best performing model from the experiments on PCA? (Generally, it is hard to say a prior; one has to do the experiments in order to figure out).

Due Date and Time

The due date/time is **Nov. 26, 2013, 3:15pm**. A single two-day extension or two one-day extensions are allowed, combining with your previous requests. Other late submissions will get at most half the credit.

Collaboration

For the algorithm part of the assignment, you can collaborate. However, each has to turn in his/her own write-up.

For the coding part of the assignment, you may collaborate in groups of up to 4. Each group needs to submit to blackboard only once. Group members share the same credits.

Blackboard Submission Instructions

You are required to submit both runnable Matlab code files, result mats (.mat) and a formatted report (PDF) via the blackboard system.

Pack your (programming) writeup report (in PDF format, do not forget listing all your team members), and the Matlab file storing the predictions (hw5_prediction.mat) and clustering data (clustering.mat) into a single zipped file.

Upload the single zip archive file *containing a single folder*. **The name of the zip file should be your abcd_hw5_AB.zip** where abcd is the last 4 digits of your USC ID and AB is your initials. **The folder should be named as results**. If you are working in a group, only one student from the group should upload the files. Please use the uploading student's ID number for the folder name.

The folder should contain all your .m files and also an PDF report named *HW5Algorithm.pdf.* (for the algorithm part)Additionally, if you are submitting the coding part for the group, your folder should also contain *HW5Programming.pdf.* The report should contain the names and USC student IDs of all the group members. It should answer all programming questions — make you mark clearly each question you are answering. Since you will have plots, you will need to refer to the plots (such as "Please see Fig. 1" — LaTeX does not place plots next to texts so you need to number the plots and refer that way.)