Homework 3
K-means, Hierarchical clustering, Mixture model and PCA

CMU 10-601: Machine Learning (Fall 2015)
http://www.cs.cmu.edu/~10601b/
OUT: Oct. 1, 2015
DUE: Oct 13, 2015, 10:30 AM

START HERE: Instructions

- The homework is due at 10:30 am on Tuesday October 13, 2015. Each student will given three late days that can be spent on any homeworks but not on projects. Once you have used up your late days for the term, late homework submissions will receive 50% of the grade if they are one day late, and 0% if they are late by more than one day.

- ALL answers will be submitted electronically through the submission website: https://autolab.cs.cmu.edu/courses/10601b-f15/assessments. You can sign in using your Andrew credentials. You should make sure to edit your account information and choose a nickname/handle. This handle will be used to display your results for any competition questions (such as the class project) on the class leaderboard.

- Some questions will be *autograded*. Please make sure to carefully follow the submission instructions for these questions. A template submission can be downloaded at http://www.cs.cmu.edu/~10601b/assignments/hw3template.tar

- Collaboration on solving the homework is allowed (after you have thought about the problems on your own). When you do collaborate, you should list your collaborators! You might also have gotten some inspiration from resources (books or online etc...). This might be OK only after you have tried to solve the problem, and couldn't. In such a case, you should cite your resources.

- If you do collaborate with someone or use a book or website, you are expected to write up your solution independently. That is, close the book and all of your notes before starting to write up your solution. You should also state your collaborations in your short-answer writeup. Specifically, please write down the following:

  1. Did you receive any help whatsoever from anyone in solving this assignment? Yes / No. If you answered yes, give full details: (e.g., "Jane explained to me what is asked in Question 3.4").
  2. Did you give any help whatsoever to anyone in solving this assignment? Yes / No. If you answered yes, give full details: (e.g., "I pointed Joe to section 2.3 to help him with Question 2").

  Collaboration without full disclosure will be handled severely, in compliance with CMU's Policy on Cheating and Plagiarism.

1 Short Answer Question [Pengcheng Zhou; 30 points]

- **Submission Instructions:** For each question you will be required to write your answers in a single text file. This file needs to have the same name as the question, and may have the extension .txt. For example, "1.1" or "1.1.txt". On each line, write your answer **and only your answer, no computations**, as a single floating point number, to at least 3 decimal places. You may number the lines of the file if you like, for example "1. <answer>", but be consistent.

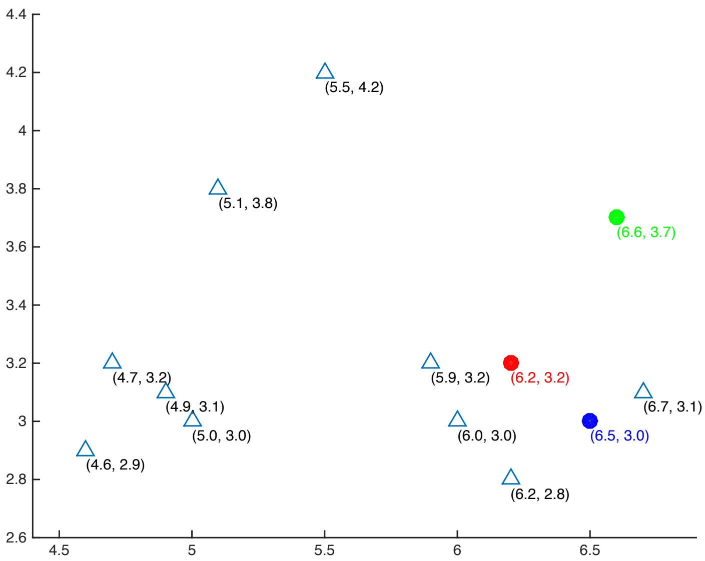

Figure 1: Scatter plot of datasets and the initialized centers of 3 clusters

1.1 Implement k-means manually [8 pts]

Given the matrix $\mathbf{X}$ whose rows represent different data points, you are asked to perform a k-means clustering on this dataset using the Euclidean distance as the distance function. Here k is chosen as 3. The Euclidean distance d between a vector $\mathbf{x}$ and a vector $\mathbf{y}$ both in $\mathcal{R}^p$ is defined as $d = \sqrt{\sum_{i=1}^{p}(x_i - y_i)^2}$. All data in $\mathbf{X}$ were plotted in Figure 1. The centers of 3 clusters were initialized as $\mu_1 = (6.2, 3.2)$ (red), $\mu_2 = (6.6, 3.7)$ (green), $\mu_3 = (6.5, 3.0)$ (blue).

$$\mathbf{X} = \begin{bmatrix} 5.9 & 3.2 \\ 4.6 & 2.9 \\ 6.2 & 2.8 \\ 4.7 & 3.2 \\ 5.5 & 4.2 \\ 5.0 & 3.0 \\ 4.9 & 3.1 \\ 6.7 & 3.1 \\ 5.1 & 3.8 \\ 6.0 & 3.0 \end{bmatrix}$$

1. What's the center of the first cluster (red) after one iteration? (Answer in the format of **[x1, x2]**, round your results to three decimal places, same as problems 2 and 3) [2 pts]

2. What's the center of the second cluster (green) after two iteration? [2 pts]

3. What's the center of the third cluster (blue) when the clustering converges? [2 pts]

4. How many iterations are required for the clusters to converge? [2 pts]

1.2 Application of $k-$means [6 pts]

There are 6 different datasets noted as A,B,C,D,E,F. Each dataset is clustered using two different methods, and one of them is K-means. All results are shown in Figure 2. You are required to determine which result

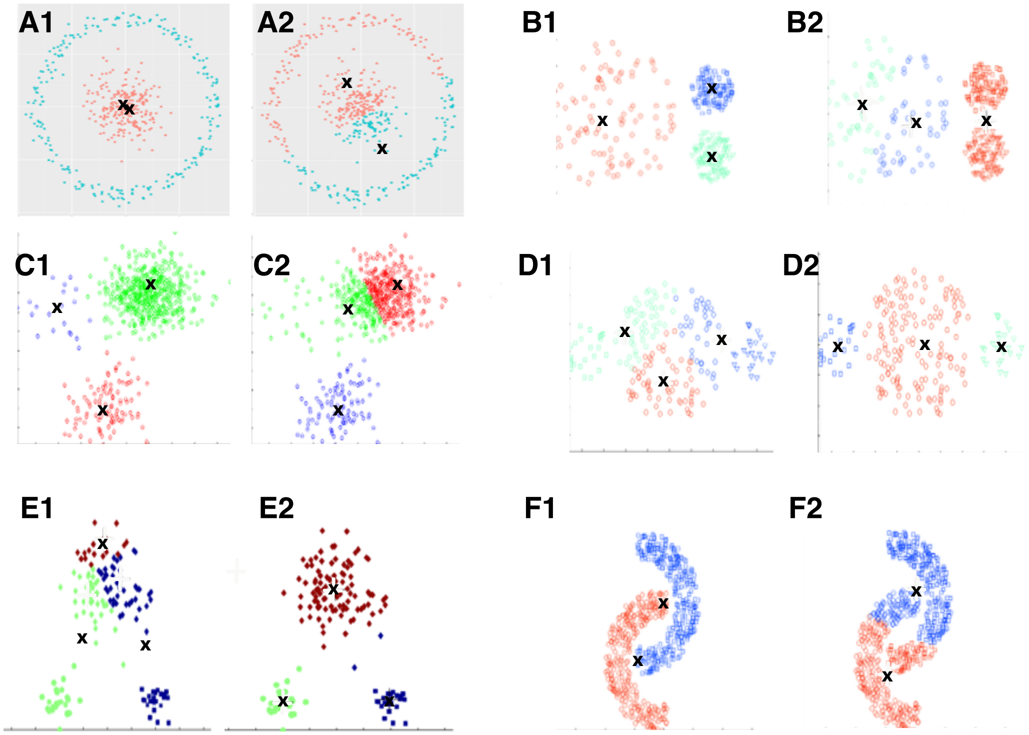

Figure 2: Clustered results for 6 datasets

is more likely to be generated by K-means method. (Hint: check the state when K-means converges; Centers for each cluster have been noted as **X**; Since x and y axis are scaled proportionally, you can determine the distance to centers geometrically). The distance measure used here is the Euclidean distance.

1. Dataset A (write A1 or A2, [1 pt], same in the following question);

2. Dataset B

3. Dataset C

4. Dataset D

5. Dataset E

6. Dataset F

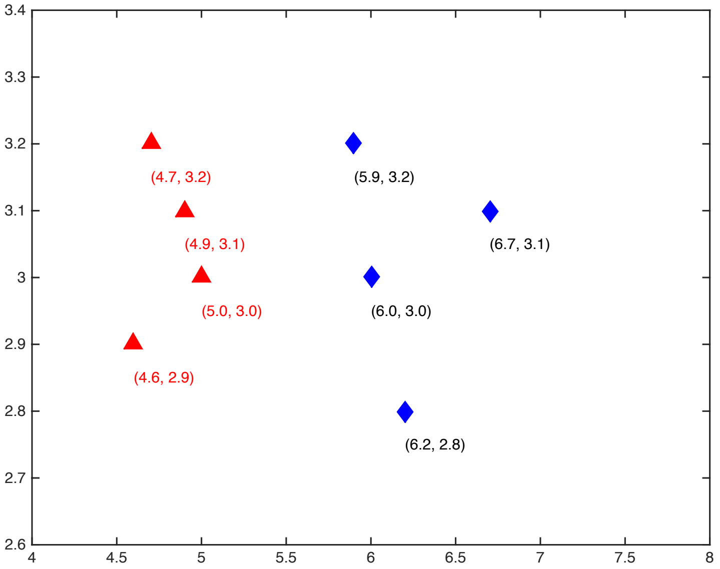

Figure 3: Scatter plot samples in two clusters

1.3 Hierarchical clustering [9 pts]

In Figure 3 there are two clusters A (red) and B (blue), each has four members and plotted in Figure 3. The coordinates of each member are labeled in the figure. Compute the distance between two clusters using Euclidean distance.

1. What is the distance between the two farthest members? (complete link) (round to **four** decimal places here, and next 2 problems); [2 pt]

2. What is the distance between the two closest members? (single link) [2 pt]

3. What is the average distance between all pairs? [4 pt]

4. Among all three distances above, which one is robust to noise? Answer either "complete", "single", or "average". [1 pt]

1.4 PCA [7 pts]

Consider 4 data points in the 2−d space: $(-1,-1),(0.5,-0.5),(1,1),(-0.5,0.5)$.
1. What is the first principal component? (Answer in the format of **[a, b]**, round to **4** decimal places, use positive values in the case of roots); [2 pt]

2. If we project all points into the 1−d subspace by the second principal components. What is the variance of the projected data? [2 pt]

3. For a given dataset of $\mathbf{X}$, all the eigenvalues of its covariance matrix $\mathbf{C}$ are $\{2.2, 1.7, 1.4, 0.8, 0.4, 0.2, 0.15, 0.02, 0.001\}$, if we want to explain more than 90% of the total variance using the first k principal components, what is the least number of k? [3 pt]

2 Programming [Joe Runde; 70 points]

This is a coding question. The focus this week is on dimensionality reduction and unsupervised learning, or clustering. You will see how it is possible to learn a model of the data, and the associated class labels without access to training labels. Some general instructions:

- **Octave:** You must write your code in Octave. Octave is a free scientific programming language, with syntax identical to that of MATLAB. Installation instructions can be found on the Octave website. (You can develop your code in MATLAB if you prefer, but you must test it in Octave before submitting, or it may fail in the autograder.)
- **Autograding:** This problem is autograded using the CMU Autolab system. The code which you write will be executed remotely against a suite of tests, and the results used to automatically assign you a grade. To make sure your code executes correctly on our servers, you should avoid using libraries which are not present in the basic Octave install.
- **Submission Instructions:** For each sub-question you will be given a single function signature. You will be asked to write a single Octave function which satisfies the signature. In the code handout linked at the top of the assignment, we have provided you with a single folder containing stubs for each of the functions you need to complete. Do not rename these files. Complete each of these functions, then compress the files as a tar archive and submit to Autolab online. You may submit code as many times as you like. When you download the files, you should confirm that the autograder is functioning correctly by compressing and submitting the directory of stubs provided. This should result in a grade of zero for all questions.
- **READ THIS:**
  When writing your program, ensure that it both accepts input matrices and outputs matrices with the dimensions given in the problem. If it doesn't, **you will not receive any points.**

2.1 K-means [30 pts]

In this question we'll explore the big problem with K-means: how do we choose K? Increasing K will always increase the likelihood of the model, but with N data points, we would like to choose some K < N. One method is to maximize the **Bayesion Information Criterion.** This criterion is similar to likelihood, but it places a penalty on the complexity of the model. Page 4 of this paper has a good explanation: `http://www.aladdin.cs.cmu.edu/papers/pdfs/y2000/xmeans.pdf`. A sample dataset for this problem can be downloaded at: `http://www.cs.cmu.edu/~10601b/assignments/multigauss.mat`.
For this problem:

- k is the number of centroids.
- f is the number of features.
- C is a $k \times f$ matrix of centroid centers.
- idx is an $n \times 1$ vector of centroid indices. X_i belongs to centroid idx_i.
- X is an $n \times f$ matrix of training data. This data is drawn from a mixture of spherical gaussians with variance σ

1. **BIC [20 pts]**
   Complete the function [bic] = BIC(X, C, idx, k). This function implements BIC as explained on Page 4 of the paper linked above, returning a single value. In this paper, R is the total number of data points, $R_j = \sum_i \delta(idx_i == j)$, and $p_k = k(f + 1)$.
2. **Choose K [10 pts]**
   Complete the function [k] = ChooseK(X, restarts, max_K). This function should apply K-Means to the dataset X, and return the value for $K < max_K$ which maximizes BIC.
   - Use the kmeans method in octave (also in matlab). Its usage is: [idx, C] = kmeans(X, k)

- kmeans uses random starting points, and may suffer from a bad initialization. For each potential value of k, run kmeans *restarts* number of times, and record the best BIC value.
- Since you'll need to use your BIC function, first ensure that it is correct.

2.2 Expectation Maximization (EM) [40 pts]

In this question you will implement the EM algorithm for Gaussian Mixture Models. A good read on gaussian mixture EM can be found at: http://www.ics.uci.edu/~smyth/courses/cs274/notes/EMnotes.pdf. A sample dataset for this problem can be downloaded at: http://www.cs.cmu.edu/~10601b/assignments/multigauss.mat.

For this problem:

- n is the number of training points.
- f is the number of features.
- k is the number of gaussians.
- X is an $n \times f$ matrix of training data.
- w is an $n \times k$ matrix of membership weights. $w(i,j)$ is the probability that x_i was generated by gaussian j
- t is a $k \times 1$ vector of mixture weights (gaussian prior probabilities). t_i is the prior probability that any point belongs to class i.
- mu is a $k \times f$ matrix containing the means of each gaussian.
- $sigma$ is an $f \times f \times k$ tensor of covariance matrices. $sigma(:,:,i)$ is the covariance of gaussian i

1. **Expectation [10 pts]**
   Complete the function [w] = Expectation(X, k, t, mu, sigma). This function takes in a set of parameters of a gaussian mixture model, and outputs the membership weights of each data point.

2. **Maximization of Means [5 pts]**
   Complete the function [mu] = MaximizeMean(X, k, w). This function takes in the training data along with the membership weights, and calculates the new maximum likelihood mean for each gaussian.

3. **Maximization of Covariances [15 pts]**
   Complete the function [sigma] = MaximizeCovariance(X, k, w, mu). This function takes in the training data along with membership weights and means for each gaussian, and calculates the new maximum likelihood covariance for each gaussian.

4. **Maximization of Mixture Weights [5 pts]**
   Complete the function [t] = MaximizeMixtures(k, w). This function takes in the membership weights, and calculates the new maximum likelihood mixture weight for each gaussian.

5. **EM [5 pts]**
   Put everything together and implement the function [t, mu, sigma] = EM(X, k, t_0, mu_0, $sigma_0$, nIter). This function runs the EM algorithm for nIter steps and returns the parameters of the underlying GMM. **Note:** Since this code will call your other functions, make sure that they are correct first. A good way to test your EM function offline is to check that the log likelihood, $logP(X|parameters)$ is increasing for each iteration of EM.