CS 383 - Machine Learning

Assignment 2 - Clustering
Summer 2017

Introduction

In this assignment you'll work on clustering data.

You may not use any functions from machine learning library in your code, however you may use statistical functions. For example, if available you **MAY NOT** use functions like

- kmeans

however you **MAY** use basic statistical functions like:

- std
- mean
- cov
- eig

Grading

Although all assignments will be weighed equally in computing your homework grade, below is the grading rubric we will use for this assignment:

Part 1 (Theory)	0pts
Part 2 (Basic k-Means Clustering)	30pts
Part 3 (Flexible k-Means Clustering)	10pts
Report	5pts
TOTAL	45pts

Table 1: Grading Rubric

DataSets

Pima Indians Diabetes Data Set In this dataset of 768 instances of testing Pima Indians for diabetes each row has the following information (1 class label, 8 features).

1. Class Label (-1=negative,+1=positive)
2. Number of times pregnant
3. Plasma glucose concentration
4. Diastolic blood pressure (mm Hg)
5. Triceps skin fold thinkness (mm)
6. Insulin (mu U/ml)
7. Body mass index (kg/m^2)
8. Diabetes pedigree function
9. Age (yrs)

Data obtained from: https://archive.ics.uci.edu/ml/datasets/Pima+Indians+Diabetes

1 Theory

None

2 Clustering

Let's implement our own version of k-means to cluster data!

Write a script that:

1. Reads in the data
2. Standardizes the data
3. Performs k-means clustering **using just the 6th and 7th feature of the data (BMI and Pedigree, respectively) with k=2** and using the Euclidean (L2) distance.

In addition, in your k-means code you'll want to visualize the progress of the algorithm (this will be part of your report):

1. Plot the initial setup
   (a) Data points are red 'x'
   (b) Cluster centers are blue 'o'
2. Plot the initial cluster assignments
   (a) Cluster 1 = red
   (b) Cluster 2 = blue
   (c) Data points are as 'x' (according the their assigned color)
   (d) Cluster centers are as 'o' (according the their assigned color)
3. Plot the final cluster assignments
   (a) Cluster 1 = red
   (b) Cluster 2 = blue
   (c) Data points are as 'x' (according the their assigned color)
   (d) Cluster centers are as 'o' (according the their assigned color)
   (e) Title should indicate how many iterations it took to get there

Your figures should end up similar to Figures 1-3.

Implementation Details

1. Seed your random number generator with zero (do this right before running your k-means).
2. Randomly select two data instances and use them for the initial seeds (since we'll do $k = 2$). I suggest you use randomize the indices and use the first two to select the observations you will use to initialize your reference vectors/cluster centers.
3. Use the L2 distance to measure the distance between observations and reference vectors.
4. Terminate the EM process when the sum of magnitude of change of the cluster centers (from the previous iteration to the current one) is less than $\epsilon = 2^{-23}$. That is, when $\sum_{i=1}^{k} d(a_i(t-1), a_i(t)) < \epsilon$ where k is the number of clusters, $a_i(t)$ is the reference vector for cluster i at time t and $d(x, y)$ is the L1 distance between vectors x and y (as defined in the *Similarity and Distance Functions* link on BBlearn).
5. Write your code in such a way that it could work for any value of positive integer k, and any number of features, D. However you only have to plot the first two features and only have to plot *two* clusters.

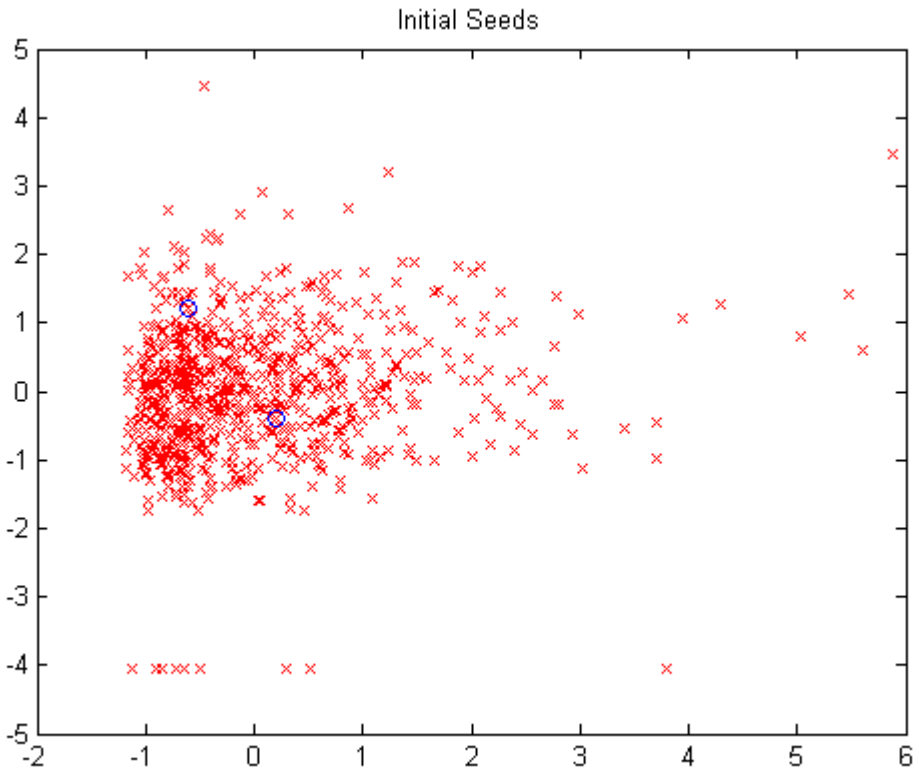

Figure 1: Seeds

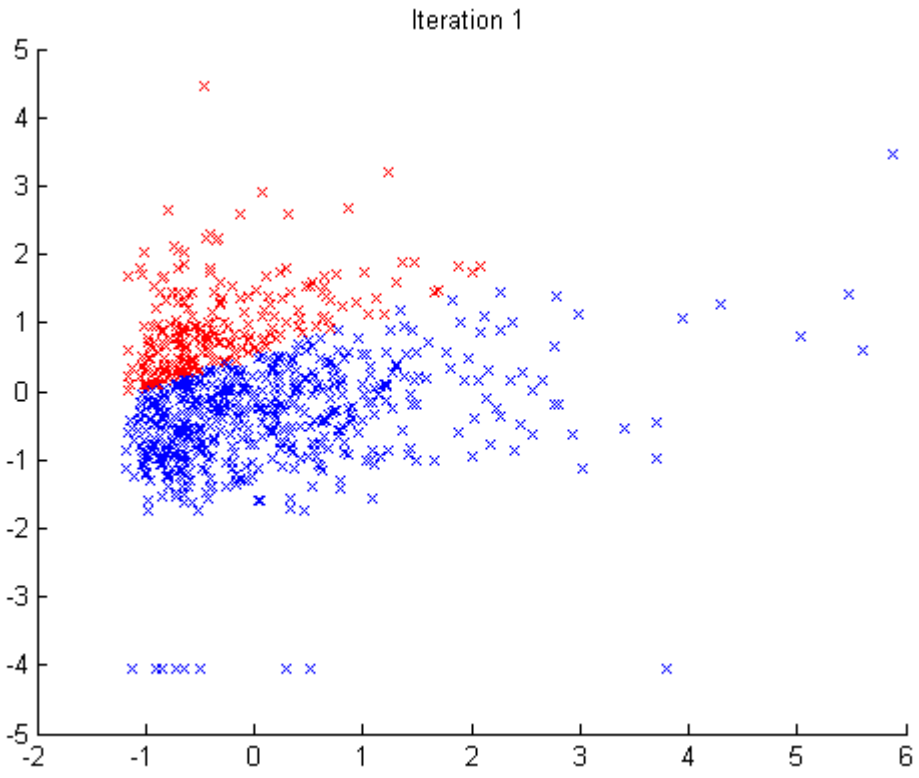

Figure 2: Initial Clustering

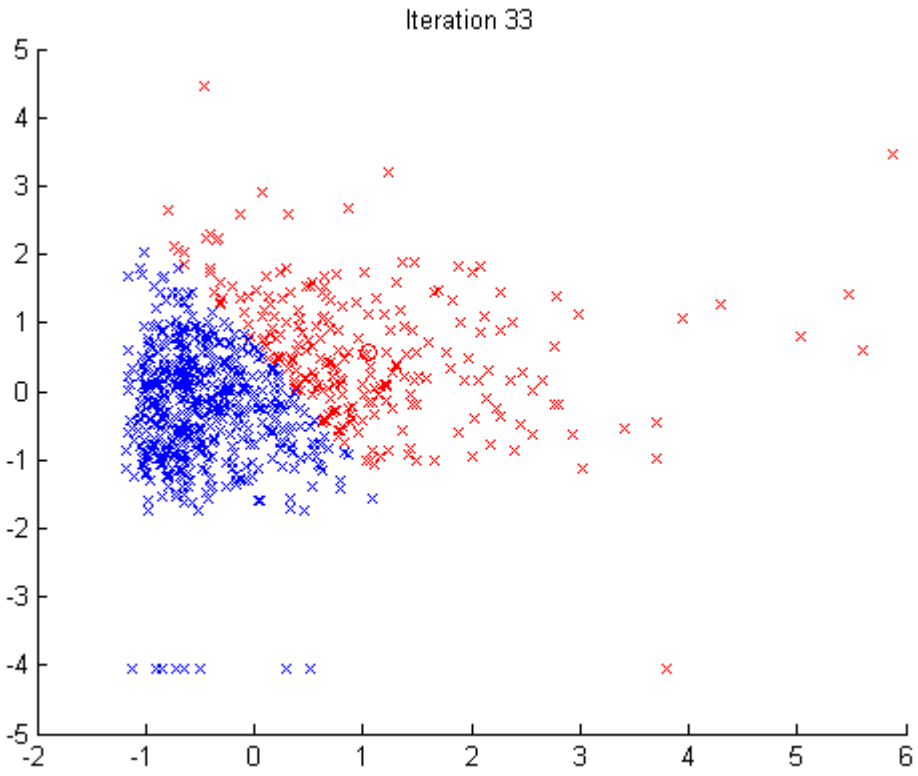

Figure 3: Final Clustering after 33 iterations

3 Clustering Part 2

Let's make sure that our code is flexible.

Re-do the clustering problem, but now allow for $1 \leq k \leq 7$ and for the user to choose which two features to use for plotting. We'll use the following color codes:

Cluster 1	Yellow
Cluster 2	Magenta
Cluster 3	Cyan
Cluster 4	Red
Cluster 5	Green
Cluster 6	Blue
Cluster 7	Black

Table 2: Cluster Colors

Depending on your implementation perhaps you created a function called *kmeans*. Would could then pass information to it like:

kmeans(data,k,xcol,ycol)

where

- *data* is our data matrix to cluster
- k is the number of clusters
- *xcol* is the column from *data* to use for plotting on the x-axis
- *ycol* is the column from *data* to use for plotting on the y-axis.

To reproduce the results from the previous section we could call this kmeans function as:

kmeans(data(:,7:-1:6), 2, 1, 2)

If we wanted to use all the data for clustering, have $k = 4$ clusters, and plot the 2nd and 3rd column then we would make the call:

kmeans(data, 4, 2, 3)

Example runs can be seen in Figures 4-6.

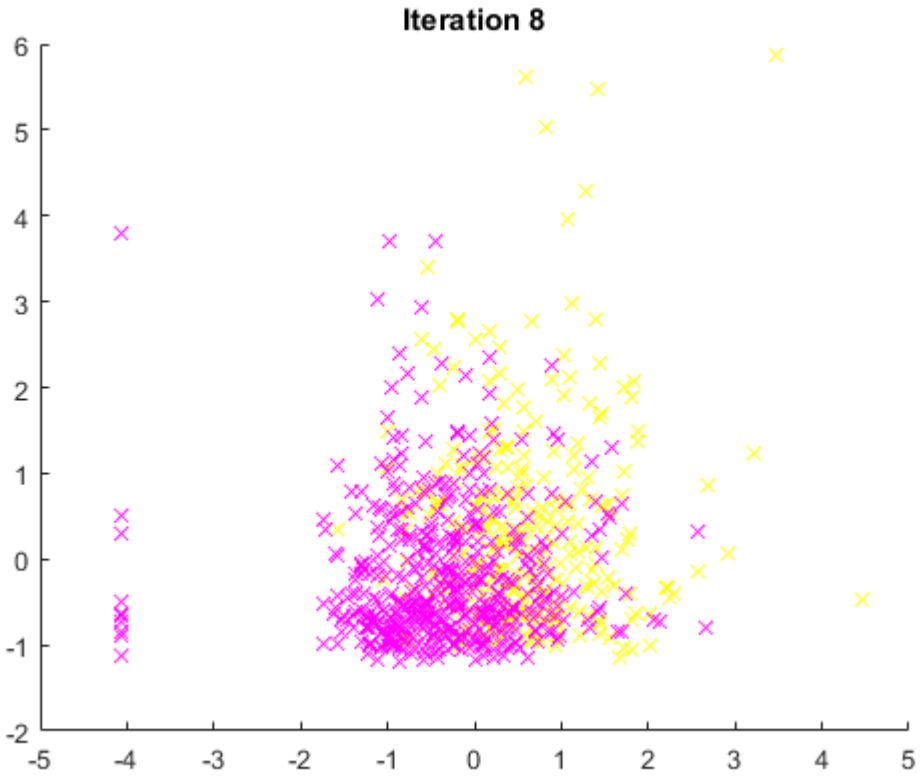

Figure 4: All features, k=2, plot features 6,7

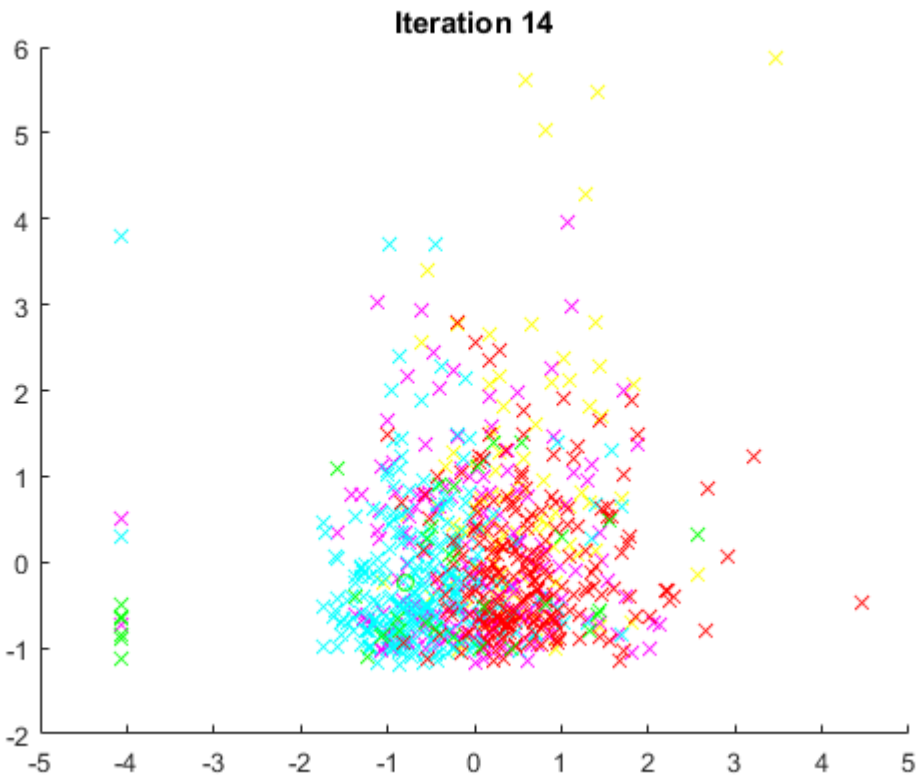

Figure 5: All features, k=5, plot features 6,7

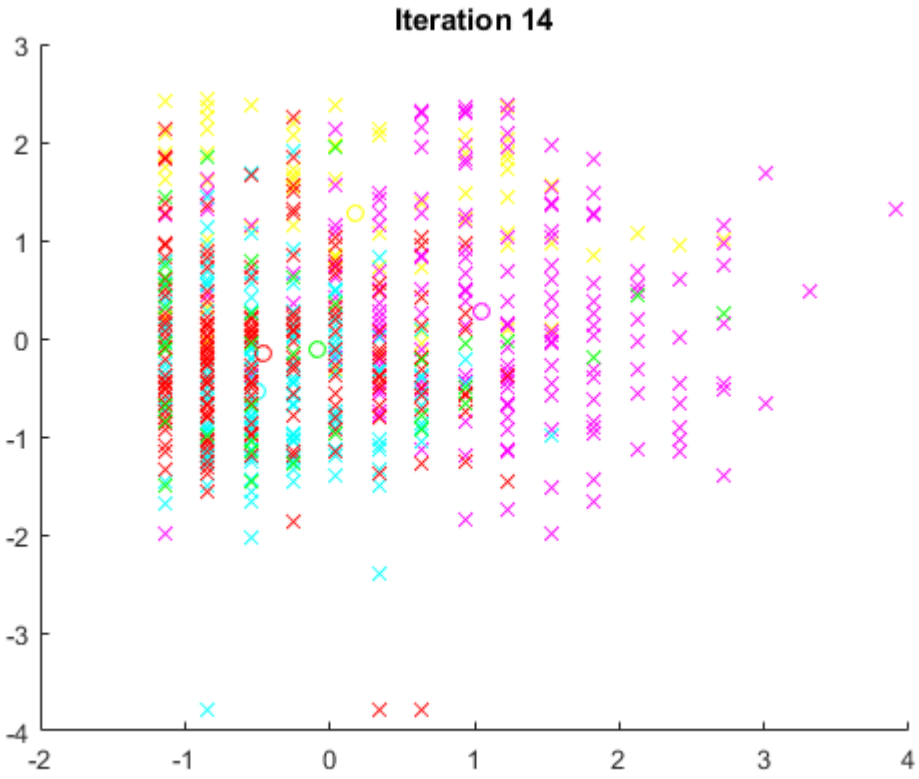

Figure 6: All features, k=5, plot features 1,2

Submission

For your submission, upload to Blackboard a single zip file containing:

1. PDF Writeup
2. Source Code
3. readme.txt file

The readme.txt file should contain information on how to run your code to reproduce results for each part of the assignment.

The PDF document should contain the following:

1. Part 1: None
2. Part 2: The visualization of the k-means clustering process including:
   (a) The initial setup visualization
   (b) The initial cluster assignment visualization
   (c) The final cluster assignment visualization

   and report how many iterations it took for your algorithm to terminate.
3. Part 3: Terminal cluster examples for at least three different runs, where the number of clusters (k), the number features to be clustered (D), and the chosen features to plot, varies between runs.