K-means and EM-GMM

Introduction to Machine Learning (CSCI 1950-F), Summer 2011

In these exercises, we use the K-means and EM-GMM algorithms to cluster data gathered from the Old Faithful geyser. These instructions (this file) can be downloaded from the course website along with code and data for use in the exercises below.

Old Faithful is a hydrothermal geyser in Yellowstone National Park in the state of Wyoming, USA. Its name stems from the supposed regularity of its eruptions. (It is typically around 60-90 minutes between eruptions.)

`http://en.wikipedia.org/wiki/Old_Faithful`

The data set `faithful.txt` consists of 272 observations, each of which represents a single eruption and contains 2 variables: one corresponding to the duration in minutes of the eruption, and the other equal to the time until the next eruption (also in minutes). The data is arranged in a matrix `x` in which each row corresponds to an observation.

(1) Implement and run the K-means algorithm

(a) The script `main.m` loads the data and calls the function `kmeans.m`, which is intended to implement the K-means algorithm. However, the function is incomplete. Your task is to fill in the missing parts. **Print out and submit the code you wrote.**

(b) With `part = 1`, run the script `main.m` to cluster the data set (using the code you wrote) and plot the evolution of the algorithm over time. Run the script several times to get a feel for the behavior. (Since the random number generator is not reset, each run gives you a different initialization of the algorithm.) **Print out and submit the final plot.**

(2) Compare with EM-GMM

The function `EM_GMM.m` implements the Expectation-Maximization (EM) algorithm on a Gaussian mixture model (GMM). In other words, it uses EM to estimate the parameters of this probabilistic model for cluster-like data.

With `part = 2`, run the script `main.m` to cluster the data set using both K-means and EM-GMM,

and to plot the evolution of each of the algorithms over time. Run the script several times to get a feel for the behavior.

A note on terminology: we say "cluster" to mean "assign each point to a group". There is not a universally accepted measure of the quality of a given clustering (that is, a given assignment of points). So in the following questions, I'm asking you to rate the algorithms based on your intuitive perception of what constitutes a group.

Submit your answers to the following questions. How well does each algorithm do at clustering the data? Would you say that one is better than the other (in terms of the clustering) on this data set?

(3) Compare on a more difficult data set

In `part = 3`, some additional points are appended to the data set. They form a "cluster" crossing directly over one of the existing "clusters", roughly in the orthogonal direction.

With `part = 3`, run the script `main.m` several times to cluster the data set using both K-means and EM-GMM.

Submit your answers to the following questions. How well does each algorithm do at clustering the data? Would you say that one is better than the other (in terms of the clustering) on this data set?

(4) Compare the computation time

In `kmeans.m`, there is a line `pause(0.5)` that slows down the script so you can see the evolution of the algorithm. (Otherwise it is too fast.) Comment (or delete) this line, and then run the script `main.m` several times. Which algorithm is typically faster?