CSCI567 Fall 2015

Homework #5

Vaibhav Behl

3123054410

vbehl@usc.edu

1. Back Propagation Through Time

We are given the loss function as follows:

$$L(y_{1,..,T}, \hat{y}_{1,..,T}) = \frac{1}{2}\sum_{i=1}^{T} \left|| y_i - \hat{y}_i ||\right|_2^2$$

a) We need gradient of L with y_t:

$$\nabla_{y_t} L = (y_t - \hat{y}_t)$$

b)

$$\nabla_{s_T} L = \nabla_{y_T} L \,.\, \nabla_{s_T} y_T$$

$$= W_{HO}^T (\mathrm{y_T} - \hat{y}_T)$$

Also,

$$\boldsymbol{\nabla_{s_t} L = \nabla_{s_{t+1}} L \,.\, \nabla_{s_t} s_{t+1}}$$

Now,

$$s_{t+1} = \sigma(W_{IH} x_t + W_{HH} s_t)$$

$$\nabla_{s_t} s_{t+1} = \sigma(W_{IH} x_t + W_{HH} s_t)[1 - \sigma(W_{IH} x_t + W_{HH} s_t)].\, diag(W_{HH})$$

$$\nabla_{s_t} s_{t+1} = s_{t+1}(1 - s_{t+1}).\, diag(W_{HH})$$

Therefor,

$$\nabla_{s_t} L = \nabla_{s_{t+1}} L \,.\, s_{t+1}.\,(1 - s_{t+1}).\, diag(W_{HH})$$

(Note: the multiplication done at many places in this question is element wise, and final output is Mx1)

c)

$$\nabla_{W_{IH}} L = \sum \nabla_{s_t} L \, \nabla_{W_{IH}} s_t$$

$$= \sum \nabla_{s_t} L.\, s_t.\,(1 - s_t).\, x_t^T$$

$$\nabla_{W_{HH}} L = \sum \nabla_{s_t} L \, \nabla_{W_{HH}} s_t$$

$$= \sum \nabla_{s_t} L.\, s_t (1 - s_t) s_{t-1}^T$$

$$\nabla_{W_{HO}} L = \sum \nabla_{y_t} L \, \nabla_{W_{HO}} y_t$$

$$= \sum (y - \hat{y}_t) s_t^T$$

d) For new s_t function:

$$s_t = (1 - \tau).\, s_{t-1} + \tau.\, \sigma(W_{IH} x_t + W_{HH} s_{t-1})$$

$$\nabla_{W_{IH}} L = \sum \nabla_{s_t} L \, \nabla_{W_{IH}} s_t$$

$$= \tau \sum \nabla_{s_t} L.\, s_t.\, (1 - s_t).\, x_t^T$$

$$\nabla_{W_{HH}} L = \sum \nabla_{s_t} L \, \nabla_{W_{HH}} s_t$$

$$= \tau \sum \nabla_{s_t} L.\, s_t (1 - s_t) s_{t-1}^T$$

The $\nabla_{W_{HO}} L$, does not change.

$$\nabla_{W_{HO}} L = \sum \nabla_{y_t} L \, \nabla_{W_{HO}} y_t$$

$$= \sum (y - \hat{y}_t) s_t^T$$

2. Kernel K-Means

The objective function for Kernel K-means is as follows:

$$D = \sum_{n=1}^{N}\sum_{k=1}^{K} r_{nk}\left|\left|\phi(x_n) - \mu_k\right|\right|_2^2 \qquad (1)$$

a) Now we take the inner term of the summation and try to represent it in the kernel form:

$$\left|\left|\phi(x_n) - \mu_k\right|\right|_2^2 = \phi(x_n)^T\phi(x_n) - 2\mu_k^T\phi(x_n) + \mu_k^T\mu_k$$

Now, this μ_k, which is mean in new feature space, can be written as:

$$\mu_k = \frac{\sum_{i=1}^{N} r_{nk}\ \phi(x_i)}{\sum_{i=1}^{N}\gamma_{nk}}$$

Putting this value back, we get:

$$\left|\left|\phi(x_n) - \mu_k\right|\right|_2^2 = \phi(x_n)^T\phi(x_n) - 2\,\frac{\sum_{i=1}^{N} r_{nk}\ \phi(x_i)\phi(x_n)}{\sum_{i=1}^{N}\gamma_{nk}} + \frac{\sum_{i=1}^{N}\sum_{i=1}^{N} r_{nk}\ \phi(x_i)\phi(x_j)}{\left(\sum_{i=1}^{N}\gamma_{nk}\right)^2} \qquad (2)$$

Now we can substitute this back into the original distance eq.(1). Thus we need to call eq.(2) in a loop for all data points and all centers.

b) Point will be assigned by to the cluster which has the minimum distance in the feature space. We can write that equation using (1) and (2) from above.

$$D_n = \arg\min_{D_n} \sum_{k=1}^{K} r_{nk}\left|\left|\phi(x_n) - \mu_k\right|\right|_2^2$$

$$= \arg\min_{D_n} \sum_{k=1}^{K} r_{nk}\left[\phi(x_n)^T\phi(x_n) - 2\,\frac{\sum_{i=1}^{N} r_{nk}\ \phi(x_i)\phi(x_n)}{\sum_{i=1}^{N}\gamma_{nk}} + \frac{\sum_{i=1}^{N}\sum_{i=1}^{N} r_{nk}\ \phi(x_i)\phi(x_j)}{\left(\sum_{i=1}^{N}\gamma_{nk}\right)^2}\right]$$

c) Pseudo code for complete kernel K-means algo:
   1. First we pre-compute the kernel matrix according to the kernel we want to use. E.g. For polynomial kernel: $K_{ij} = x_i^T x_j$.
   2. Next we randomly pick 'K' data points to be 'K' cluster centers. These will help when we want to compute their distance(dot product) to other points in the kernel space, as that data would already be in Kernel matrix.
   3. Now we start the iteration over all 'N' points and calculate each points distance to the $'\mu_k'$ in the kernel space. This is easily calculated using the dot products from the kernel matrix.
   4. Next, we assign the point $'x_n'$ to the cluster $'k'$ having the minimum distance $'D_k'$. We also set a flag 'changed' here to denote that an assignment was made.
   5. We continue this algorithm till 'changed' is not set or we reach some preset number of iterations, like 100.

3. EM Algorithm

The probability expression is gives as:

$$P(x_i) = \begin{cases} \pi + (1-\pi)e^{-\lambda} & if \;\; x_i = 0 \\ \dfrac{(1-\pi)\lambda^{x_i} e^{-\lambda}}{x_i!} & if \quad x_i > 0 \end{cases}$$

Next, the hidden variable z_n can be defined as:

$$z_n = 1 \,;\, x_n \; comes \; from \; poisson \; (\; x_i > 0)$$
$$z_n = 0 \,;\, x_n \; comes \; from \; zero - inflated (x_i = 0)$$

Now the posterior probability can be written as, combining$z_n = 0 \; and \; z_n = 1$:

$$P(z_n|\lambda, \pi) = \pi^{1-z_n}(1-\pi)^{z_n}$$

Now, the conditional becomes, combining$x_n|z_n = 0\ and\ x_n|z_n = 1$:

$$P(x_n|z_n, \lambda, \pi) = 1^{1-z_n}\left(\frac{\lambda^{x_n}e^{-\lambda}}{x_n!}\right)^{z_n}$$

The joint distribution using baye's thm:

$$P(x_n, z_n|\lambda, \pi) = \prod P(z_n|\lambda, \pi)P(x_n|z_n, \lambda, \pi)$$

$$= \prod\ \pi^{1-z_n}(1-\pi)^{z_n}1^{1-z_n}\left(\frac{\lambda^{x_n}e^{-\lambda}}{x_n!}\right)^{z_n}$$

Taking complete log likelihood of this:

$$l = \log\left(\prod\ \pi^{1-z_n}(1-\pi)^{z_n}1^{1-z_n}\left(\frac{\lambda^{x_n}e^{-\lambda}}{x_n!}\right)^{z_n}\right)$$

$$= \sum(1-z_n)\log(\pi) + z_n\log(1-\pi) + z_n(x_n\log\lambda - \lambda - \log x_n!)$$

Now starting the EM steps:

E-Step: Here we calculate the posteriors (responsibility) given some parameters at (t).

$$P(z_n = 1|\theta_t) = P(z_n = 1|x_n, \lambda_t, \pi_t)$$

$$= \left.(1-\pi_t)\left(\frac{\lambda_t^{x_n}e^{-\lambda_t}}{x_n!}\right) + \pi_t\right/(1-\pi_t)\left(\frac{\lambda_t^{x_n}e^{-\lambda_t}}{x_n!}\right) + \pi_t = \mu_{n1}$$

$$P(z_n = 0|\theta_t) = P(z_n = 0|x_n, \lambda_t, \pi_t)$$

$$= \left.\pi_t\right/(1-\pi_t)\left(\frac{\lambda_t^{x_n}e^{-\lambda_t}}{x_n!}\right) + \pi_t = \mu_{n0}$$

M-Step: Here we keep the posteriors same, and get the max. θ.

Defining a Q function:

$$Q(\theta, \theta_{old}) = Q(\lambda, \pi, \lambda_t, \pi_t)$$

$$= \sum \mu_{n0} \log \pi + \sum \quad \mu_{n1}(x_n \log \lambda + \log(1-\pi) - \lambda - \log x_n!)$$

Now taking derivatives wrt. Params:

$$\nabla_\pi Q = \frac{\sum \mu_{n0}}{\pi} - \frac{\sum \quad \mu_{n1}}{1-\pi} = 0$$

$$\pi = \sum \mu_{n0} / \sum (\mu_{n1} + \mu_{n0)}$$

$$\nabla_\lambda Q = \sum \mu_{n1} x_n / \lambda \; - \mu_{n1} \; = 0$$

$$\lambda = \sum \mu_{n1} x_n \Big/ \sum \mu_{n_1}$$

Update Steps:

1. Fix params and calculate posteriors using the E.step.
2. Fix posteriors and maximize for params using M.step.
3. Go back to 1., until convergence.

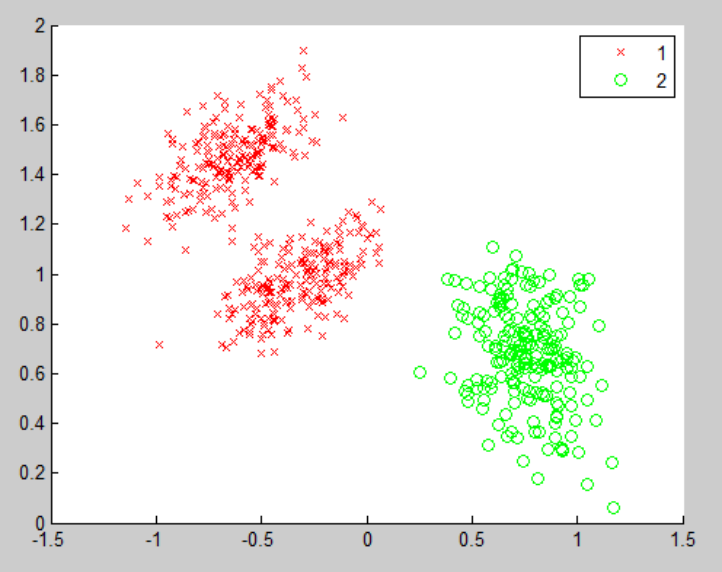
4. Programming

4.1 **Data**: loaded data with Matlab.

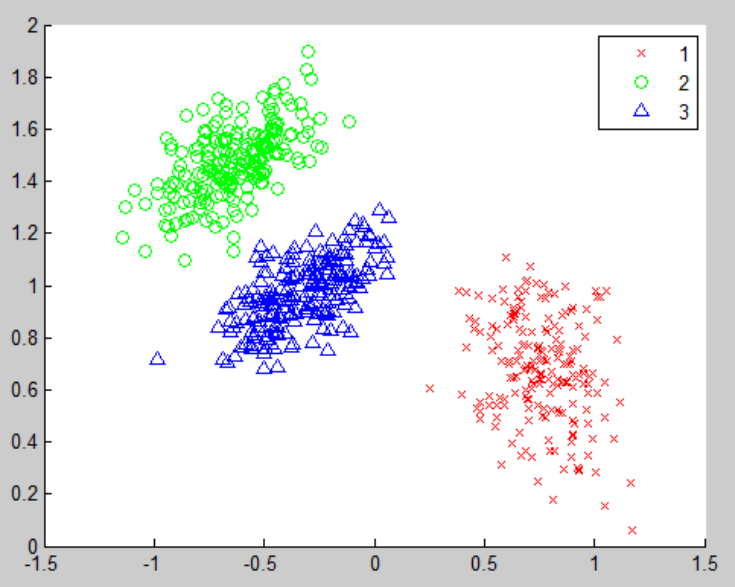
4.2 **Implement k-means**:

a) Plots for hw5_blob.mat:

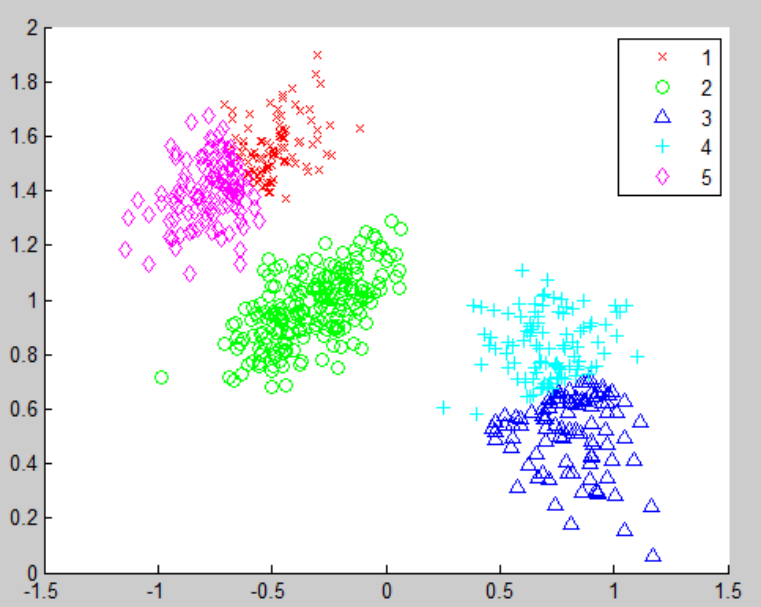
(k=2)



(k=3)

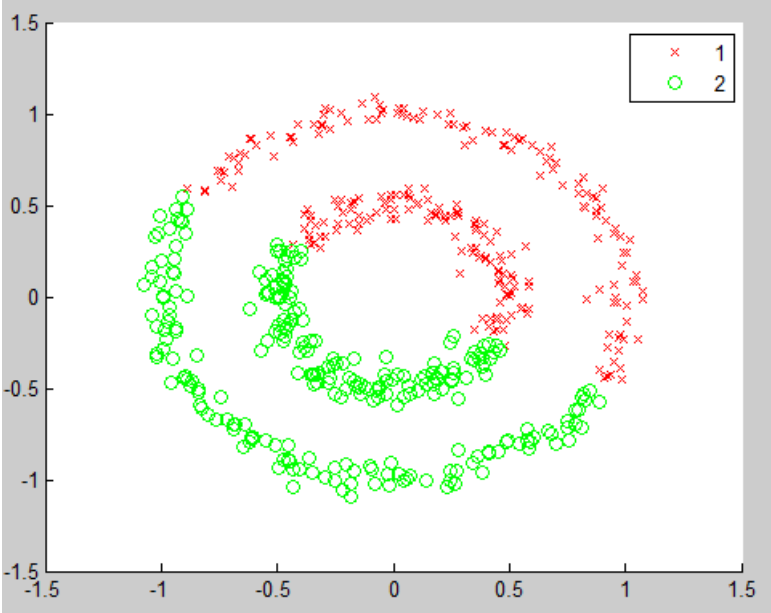


(k=5)

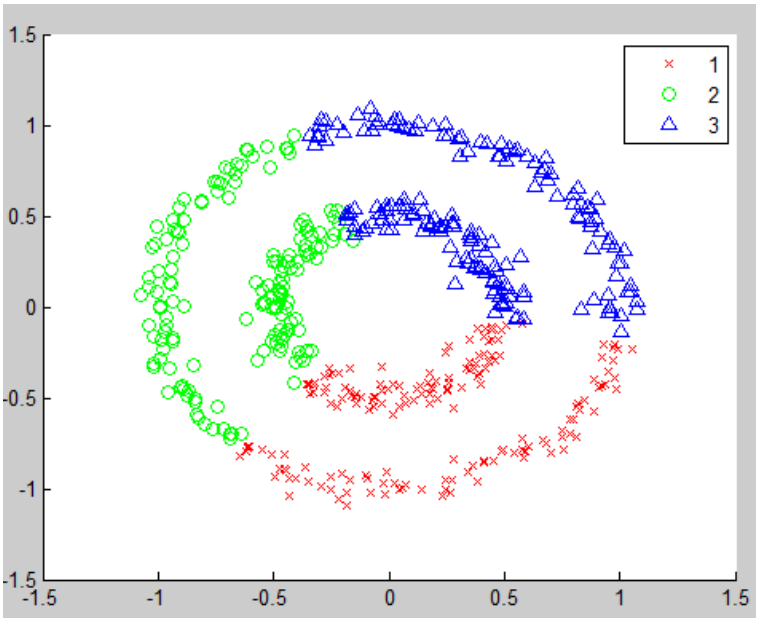


Plots for hw5_circle.mat:

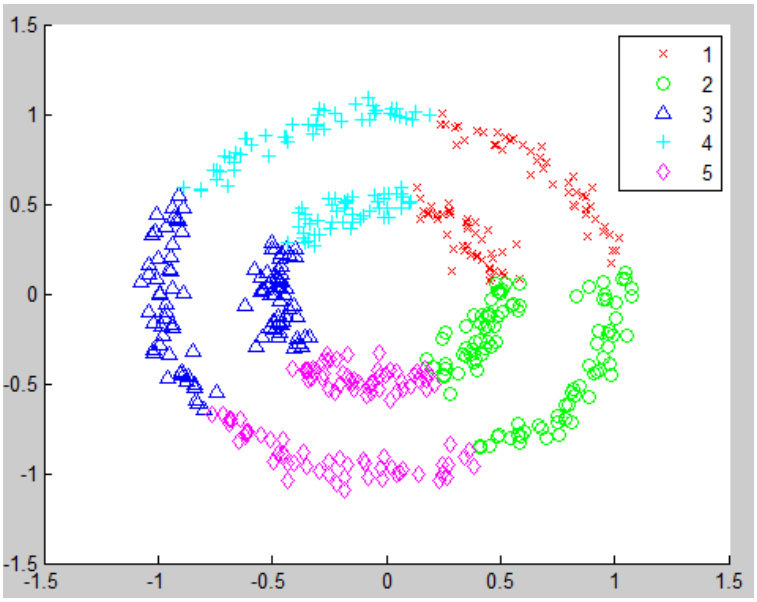
(k=2)



(k=3)



(k=5)



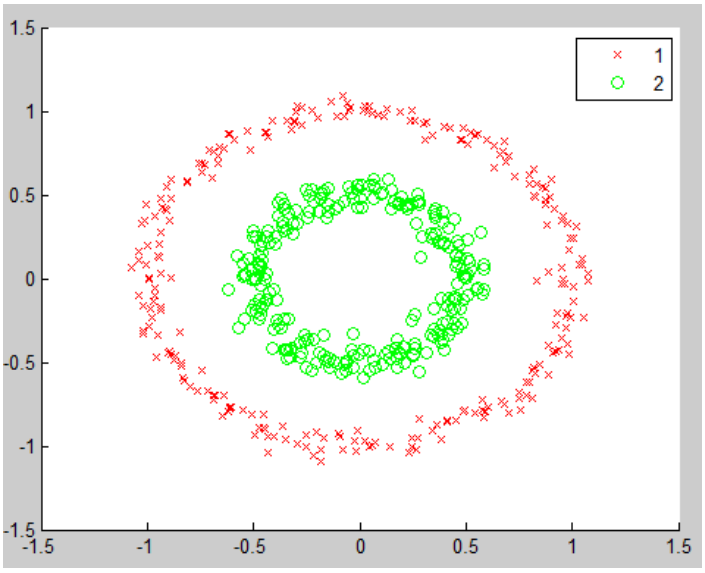
.

b) The K-means algorithm fails to separate hw5_circle.mat because it is not linearly separable.

4.3 Implement kernel k-means:

a) Choice of kernel: Polynomial. I convert the dataset from 2-D to 1-D using the transformation, $(x, y) \rightarrow (x^2 + y^2)$.

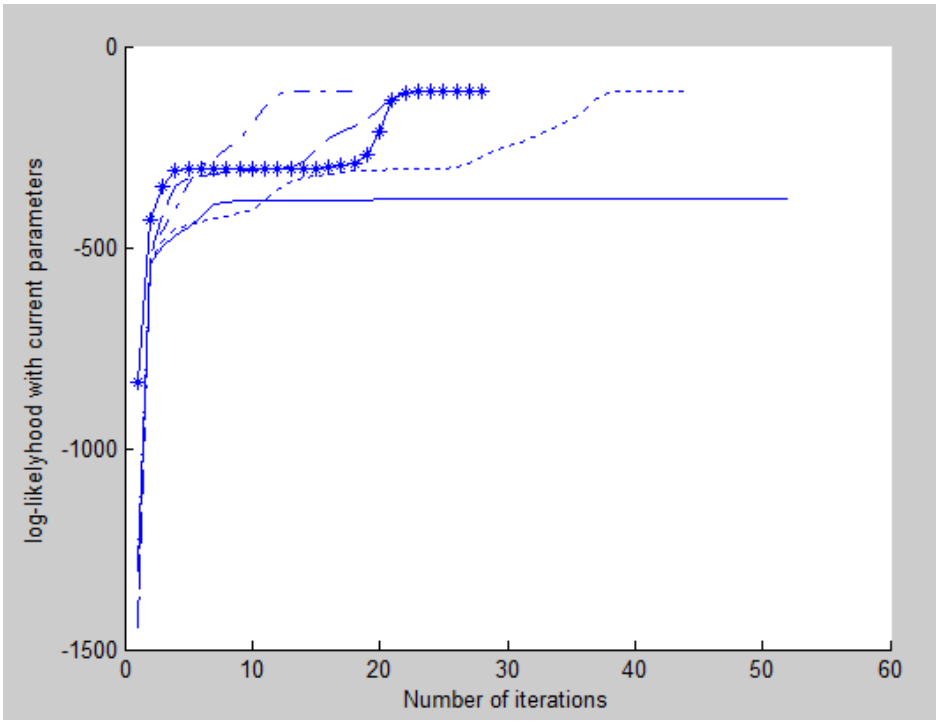
b) Plot:



4.4 Implement Gaussian Mixture Model:

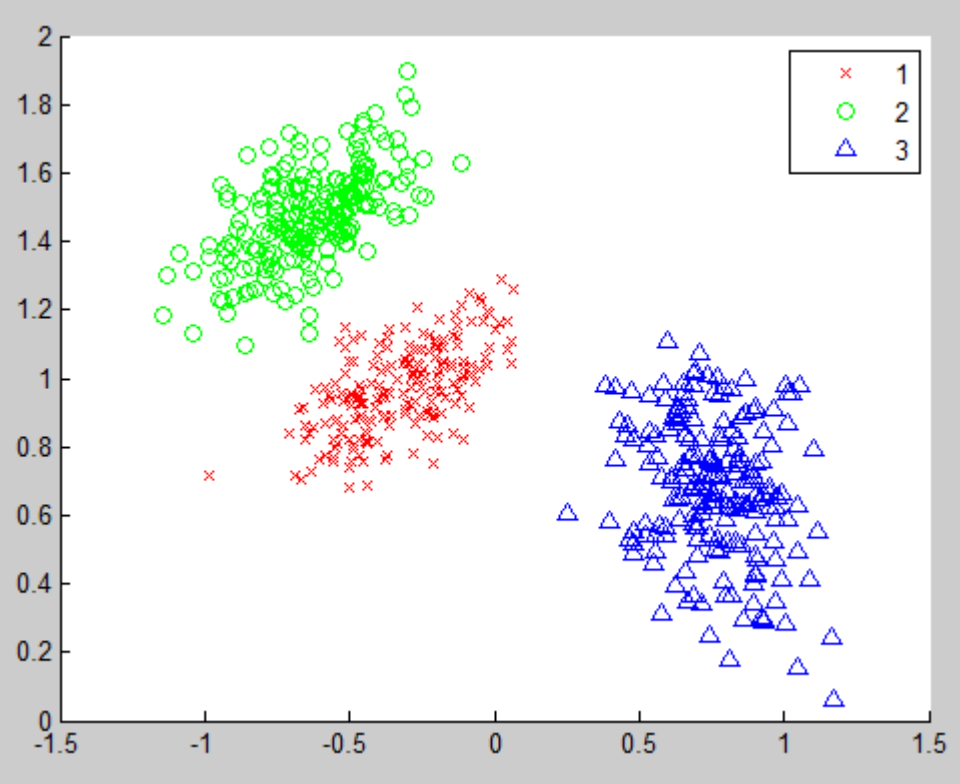
Using blob dataset.

a) For K=3:



b)

1. Most likely cluster assignment for the best run(k=3):



2. Mean and covariance matrix for best run:

mean:

-0.3259 0.9713
-0.6395 1.4746
0.7590 0.6798

covariance:

(:,:,1) =
0.0360 0.0146
0.0146 0.0163

(:,:,2) =
0.0360 0.0155
0.0155 0.0194

(:,:,3) =
0.0272 -0.0084
-0.0084 0.0404