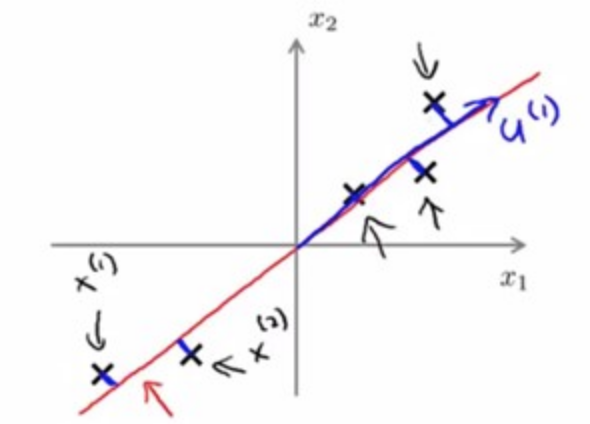
CSCI 5150 Midterm Questions

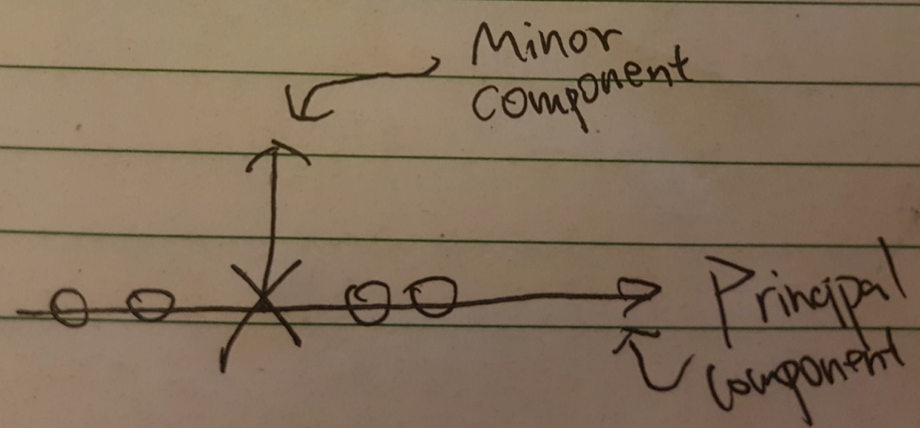
Zhang,Tuxin

1155101156

2-1.

If we want to use lines to cluster data points instead of “centroids” in K-means. We need to find a way to represent a cluster. Recall in K-means we use centroid to represent a cluster.

In PCA (or MCA) , let p be a point and v be a unit vector. Then p.v gives the distance from the origin to the projection of p on v. We can think of the vector v is the line, and the distance between a point p and v is that  the euclidean distance of position of x and after projection the position of x’. But we don’t actually want the principal component to represent my data since after projection the data points are far away. We actually want Minor component (the unit eigenvector corrsponding to the smallest eigenvalue in PCA) to represent our data. We can see below, we only use 4 points to illustrate. If we use Minor component, the X in the image is actually 1 point, all 4 data points are project to one single point X in the image, this is very similar to K-means we use 1 point to represent the cluster. After projection the data points are very close, which is what a cluster need.



K-Lines Algorithm:

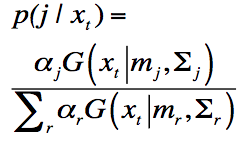
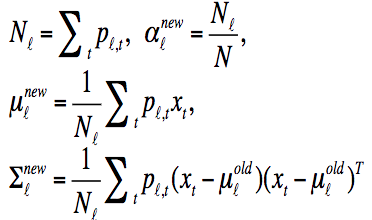
Notice: Same as K-means algorithm, this algorithm also highly depends on initialization. For simplicity we will just assume the initialization will be relatively fine.

1. Randomly initialized k lines j. (For j in 1…k)
2. For all points p, find the euclidean distance(vertical distance) from p the line j. Assign p to cluster j where p is closest among all k lines.
3. Computer MCA inside each cluster, make the minor component the vector representing the line.
4. Re-assign the all points p base on the distance(p,p’) where p’ is the position of p after projection onto minor component.
5. See if the minor component converge(stable). If not, go back to step 2.

2-2.

(a)

For convenience, Let’s first take a look at the formula in PPT-CSCI5150.18b-P10.

1. 2. 

Notice that K-means uses hard classification, Gaussian mixture model uses soft classification (We can view as each point is divide into some parts and they belongs to different clusters) .

K-Gaussian Algorithm.

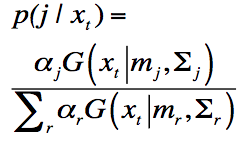
Introduction:

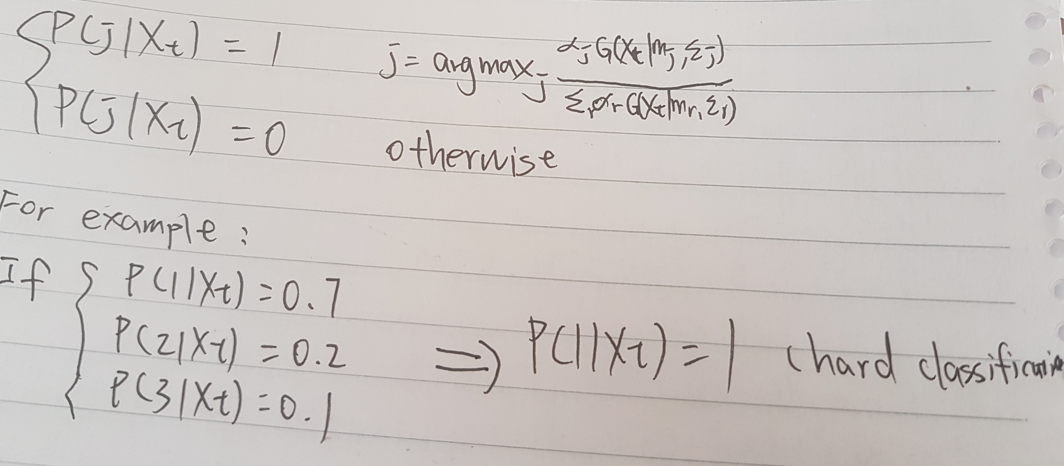
1.Use K Gaussian to represent data points (In K-means we use K centroids to represent data points).

2.Use hard classification in K-Gaussian (Like K-means).

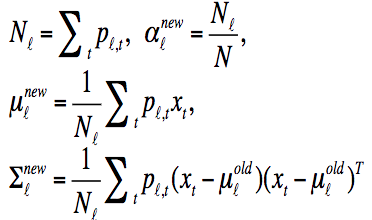
We can still using EM algorithm to estimate the parameters in K-Gaussian but with come changes.

By referring to the formulas we can get K Gaussian algorithm:

1.Since K-Gaussian use hard classification,  need to be change to



In this way, we are using hard classification like K-means.

2. Next we need use K Gaussian to represent data instead of centroids. Actually we can use the same formula, notice that the Pl,t is changed as in step 1(Hard classification). Nl is really how many points in cluster l (no probability involved, hard classification). αl is the percentage of points in this cluster with respect to the total points. μl is “true mean” in that cluster (without any probability since Pl,t is just 1 ). Same for the co-variance matrix **Σ.** Overall, the formulas in this step are not changing but the meaning of some of the elements inside the formula changed.

3. Just like EM algorithm, repeat step 1 and step 2 until it is converge(parameters are all stable). Then stop the algorithm.

(b) When all αj are equal, what is the algorithm now?

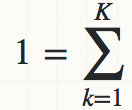
First αj are all equal means the probability that an observation comes from any population k are all equal.

Case 1:

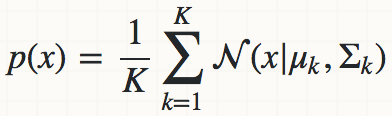
If αj are just set to be equal at the initialization stage of EM algorithm, then as EM algorithm runs, αj will be changed until it is converge for all clusters.

Case 2:

If αj are truly equal from the beginning to the end of the algorithm. Then there are some modification can be done to the algorithm.

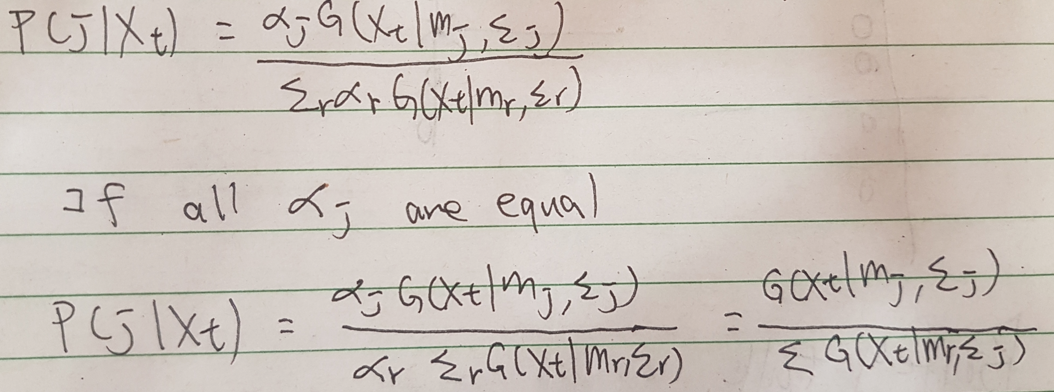
Since all αj are all equal, j = 1…K since total K clusters. And by definition  αk .

Thus for all αj = .



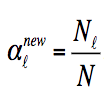
Initialization step: All αj are equal.

E step of EM algorithm:



The P(j|Xt) will become only the Gaussian probability of Xt of Gaussian j divide by the sum all Gaussian probability of Xt. This in some sense means that the probability of Xt is belongs to cluster j depends on the position of Xt from μl with respect to**Σl.** (Base on the **Σl.** the closer you are from μl, the more likely you are belongs to cluster js)

M step of EM algorithm:

 since αj are all equal, implies that all Nl are equal, which means the number of points in each cluster will be the same. Actually we don’t even need to estimate this one, because since αj are all equal, αj = 1/K (K : Total number of clusters)

μl = 1/K 

**Σl** = 1/K

Finally, see if μl and**Σl** converge. If not run EM again with the newly computed parameters.

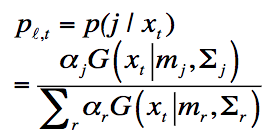
Overall, set all αj to be equal means the probability that an observation comes from any population k are all equal (αj = 1/K). Lead to number of points in each clusters are equal. Then the algorithm runs purely depends on the Gaussian distribution of data points, not related to mixing proportion αj anymore since they are all equal. And we can still use EM algorithm to estimate μl and**Σl,** (No need to estimate αj since they are all equal).

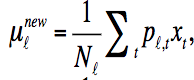
(c)

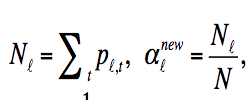
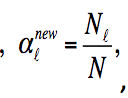
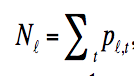
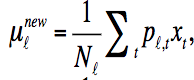
First:

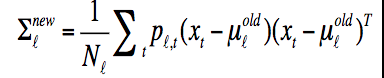
Gaussian mixture model soft assigns a point to clusters (so it give a probability of any point belonging to any centroid).

K means Hard assign a data point to one particular cluster on convergence. Since there are no probability involved, we need to delete all things related to probability.

So,  needed to be change to =1 if  has the least squared Euclidean distance to the cluster’s centroid j which is

(argmin\_j ||j - ||), otherwise it is equal to 0 , where the centroid is the mean.

From , we can also delete since we don’t need probability. Now,  should simply means the number of points in cluster l. Now  should means for all points ,  = 1 if  belongs to cluster l, otherwise 0. And this cluster has points. add all points in this cluster together,  means divide by the numbers of points which is exactly the centroid. Compute it for all l (all clusters k). Again and again under it is stale (converge).

Finally, delete all variance and co-variance  since it is K-means so we won’t need it.