4. Comparing K-means and EM

Extension Note: Project 4 due date has been extended by 1 more day to **August 22 23:59UTC** .

Generate analogous plots to K-means using your EM implementation. Note that the EM algorithm can also get stuck in a locally optimal solution. For each value of K, please run the EM algorithm with seeds $0, 1, 2, 3, 4$ and select the solution that achieves the highest log-likelihood. Compare the K-means and mixture solutions for $K = [1, 2, 3, 4]$. Ask yourself when, how, and why they differ.

Reporting log likelihood values

1.0/1.0 point (graded)

Report the maximum likelihood for each K using seeds $0, 1, 2, 3, 4$:

$\text{Log-likelihood}|_{K=1} =$ -1307.2234317600935 ✔ **Answer:** -1307.2234

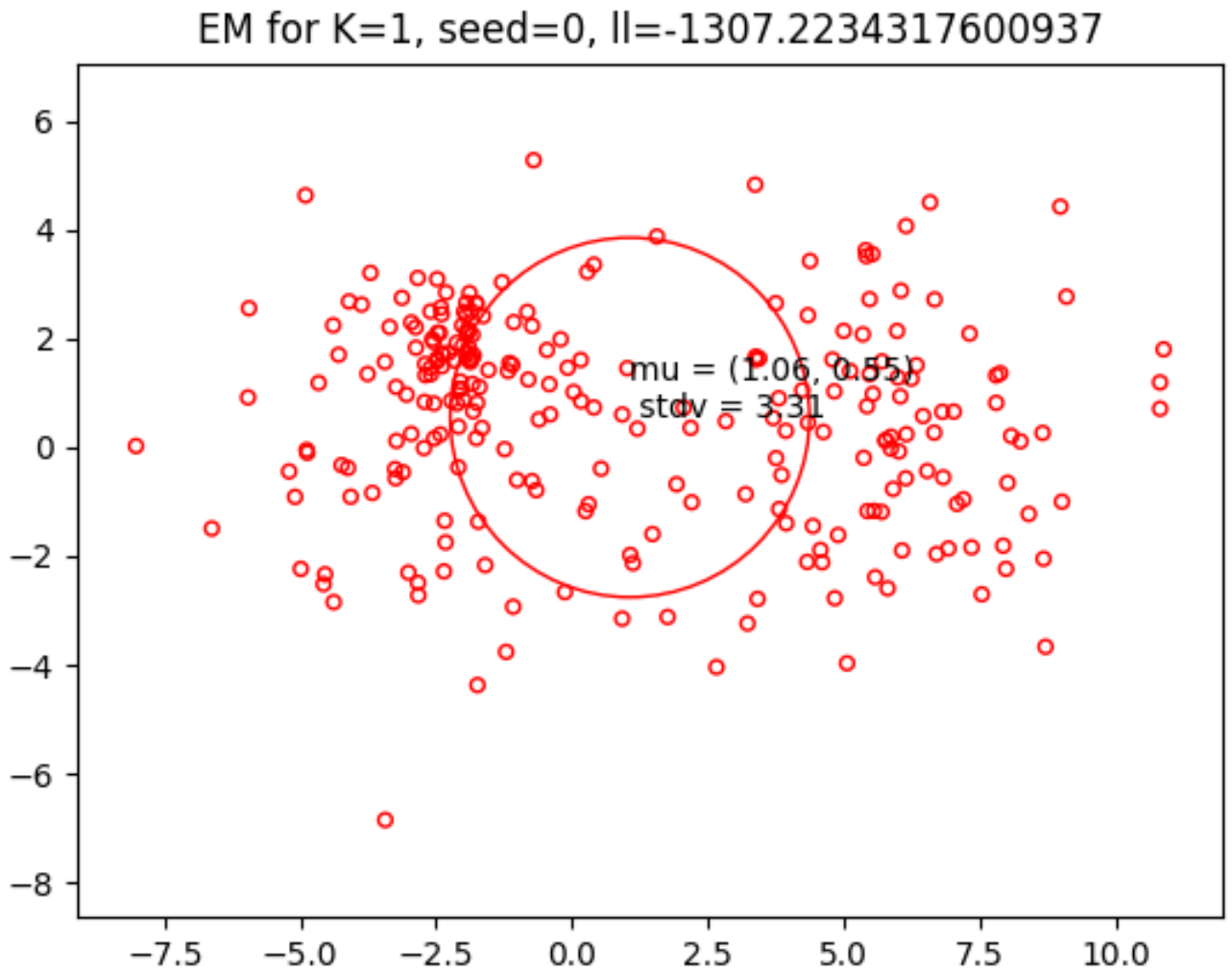
$\text{Log-likelihood}|_{K=2} =$ -1175.7146293666797 ✔ **Answer:** -1175.7146

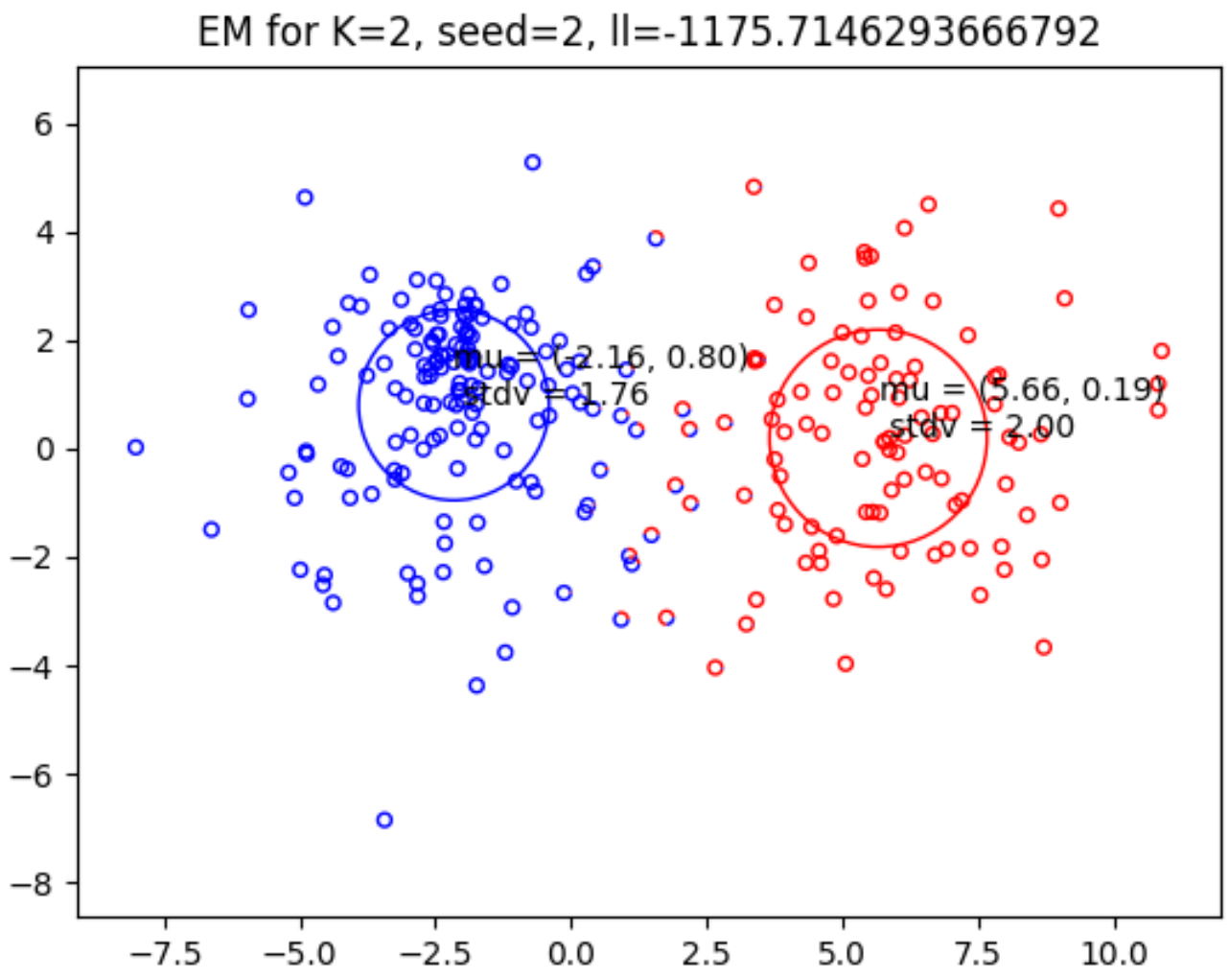
$\text{Log-likelihood}|_{K=3} =$ -1138.890899687267 ✔ **Answer:** -1138.8908

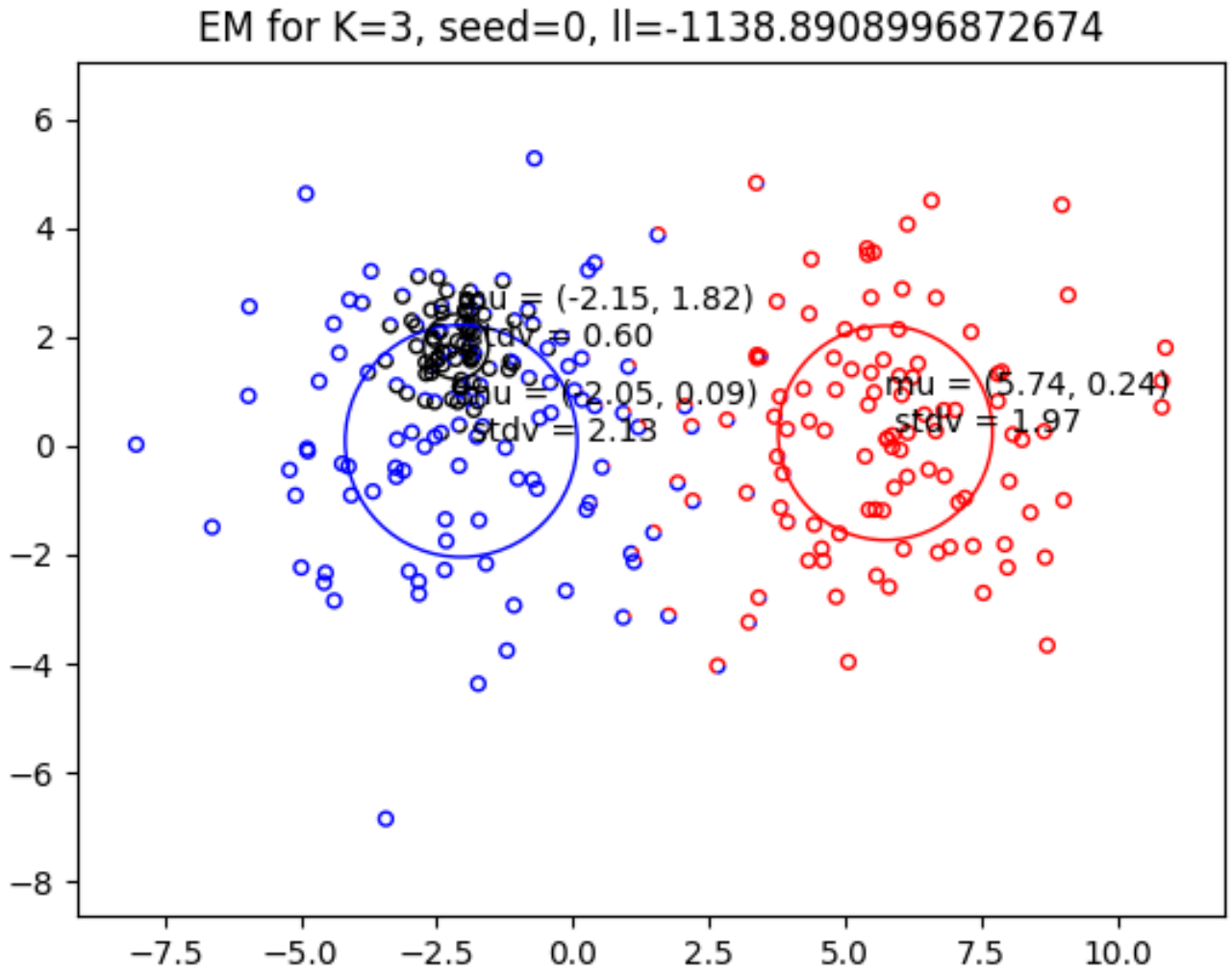
$\text{Log-likelihood}|_{K=4} =$ -1138.601175699485 ✔ **Answer:** -1138.6011

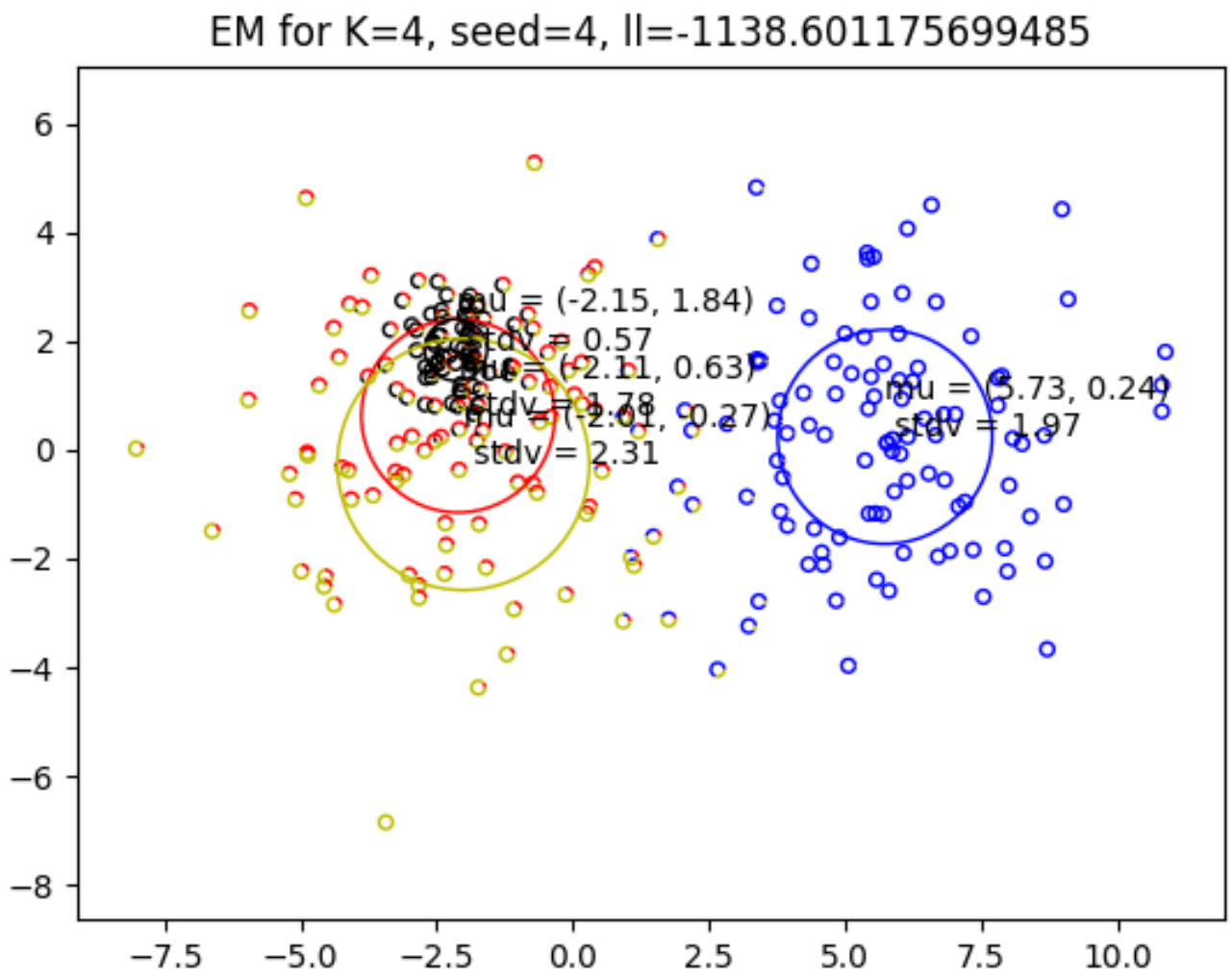
Solution:

Plots:


EM for K=1, seed=0, ll=-1307.2234317600937
mu = (1.06, 0.55)
stdv = 3.31
6
4
2
0
−2
−4
−6
−8
−7.5
−5.0
−2.5
0.0
2.5
5.0
7.5
10.0


EM for K=2, seed=2, ll=-1175.7146293666792
mu = (-2.16, 0.80)
stdv = 1.76
mu = (5.66, 0.19)
stdv = 2.00
6
4
2
0
−2
−4
−6
−8
−7.5
−5.0
−2.5
0.0
2.5
5.0
7.5
10.0





Submit You have used 3 of 20 attempts

Answers are displayed within the problem

Analysing plots

1.0/1.0 point (graded)

Which of the following sentences are true? (Check all that apply)

Note: This question is the multichoice version of the free-text question: “Compare the K-means and mixture solutions for $K = [1, 2, 3, 4]$. Ask yourself when, how, and why they differ."
In order to answer this, you should look at the plots side by side, either by adapting the code to plot them together or by simply saving the plots as you go. For each value of K, ask yourself whether the plots you see are similar or different. If they are different, why are they different?

Hint: What are we optimizing for in each case? What are we plotting? In the case of K-means, we have clusters, with EM, can these really be called clusters? What is EM optimizing for?

Now, write a descriptive paragraph of your observations as if it were part of a report for this project and you were going hand this back for us to grade. Try matching your paragraph with the options provided. If they don't match, then we wouldn't have given you full credit for this question.

Note: We have increased the attempt by 1.

- [x] In the case K=1, the mixture parameters and point assignments are the same for both methods ✔
- [x] In the case K=2, both methods have simlilar parameters and point assignments ✔
- [ ] In the case K=3, the k-means solution accounts for point density better than EM
- [x] In the case K=4, the k-means solution equally spaces the clusters to minimize distortion cost ✔

✔

Solution:

The $K = 1$ case is exactly the same since all the data is assigned to the same cluster and both methods compute the mean. The $K = 2$ case is very similar in terms of cluster mean and point assignment, but since EM uses a spherical Gaussian model with varying deviations the points midway between the two are assigned slightly differently. The $K = 3$ case equally spaces the clusters for k-means to minimize the overall distortion cost, but the left two clusters for EM are closely packed with very different variances because EM wants to account for the densely set of points on the left side. The $K = 4$ case is similar to the $K = 3$ case where all of the clusters in k-means are equally spaced, but the EM has significant overlap on the left side.

Submit You have used 2 of 4 attempts

Answers are displayed within the problem

Discussion

Hide Discussion

Topic: Unit 4 Unsupervised Learning (2 weeks) :Project 4: Collaborative Filtering via Gaussian Mixtures / 4. Comparing K-means and EM

Add a Post

All Posts

Re. explaining the point density

discussion posted 7 days ago by **ptressel** (Community TA)

This is a little different take on "account[ing] for the point density" from the other discussions here, that I hope will help with understanding how to evaluate clustering algorithms by eye.

The core difference between supervised learning and clustering (unsupervised learning) is that for clustering, we don't have ground truth for categories that the data points belong to. If we have multiple categories of data, each producing feature vector points with, then for clustering, they are all dumped in the pot together. We don't have the categories, just the observed sample distribution of points in feature vector space. Those points will have higher density in places where some category (or categories, since they can overlap) dumps more of its points.

When we make a generative model, we're trying to *mimic* the natural data's point density. The better we can match the observed point density in the training set, the better our model is...except for overfitting and being too generous with the number of parameters. Just as with supervised learning, if we overfit the training data, we risk making a model that does

not match future, live, data as well. Hence "punishing" the number of free parameters in the model, with measures like the BIC.

But back on point density and "eyeball" evaluation...

Say we have some 2D Gaussian where we know the mean and variance, and we want to visualize its samples compared to its parameters. We make a nifty plot like the ones in this project: We plot the samples as points in the 2D space. Then we draw a circle (or ellipse in the general covariance case) at one standard deviation out from the mean. What do we expect to see inside that circle? Lots of points, no? That's where the peak of the density is, so we expect to see most points near the mean, and fewer as we go outward. We can even make that concrete -- we can compute the expected fraction of points that are less than one standard deviation away from the mean. (For a 1D Gaussian, you've memorized that fraction, right? Right??)

So, if that's what we guess is the model, but what we see instead, in some real sample of data, is that the points are over yonder, not inside our one-standard-deviation circle, then maybe our model doesn't account for the data very well...

Now repeat that for multiple categories. Say we have a mixture of Gaussians, with parameters we have chosen -- means, variances, mixture proportions. Again, let's first look at what we expect to see if our model is correct. We sample lots of points from our model and plot them. On top of that we plot one-standard-deviation circles for each Gaussian. Each Gaussian, we expect, will put more points inside its one-sd circle than outside of it. The difference from our single Gaussian case is that here, the mixture proportions affect how many points we draw from each Gaussian. For a Gaussian with a low weight in our model, it may be harder to see where their peaks are. In any case, now we know what to expect our plot looks like if our model is correct.

So now, say we have a sample of data that we didn't generate from our model. Now we want to "eyeball" how good the fit is. So we make the same sort of plot: Plot the sample points, then plot the one-standard-deviation circles from our model over them. Do we see what we expect, if this is the right model for the data?

We can get fancier than this, and also take into account the mixing proportions, and also handle non-Gaussian models, by instead plotting *contour lines* for the model's probability density. Our (generative) model is one big probability density from which we can sample points. That probability density is a function (in however many dimensions our feature vectors have). We can find where the contours are numerically -- we don't need to solve for them analytically. Lots of plotting packages have tools for this -- look for contour plot or level plot. If we have more than 2D, we can (for instance) pick out pairs of features that have high variance. Or make a grid of little plots with all pairs of features. (Some stats plotting packages will make that grid of scatter plots for you. The standard R plot for this puts tiny histograms of each feature along the diagonal of the grid of plots. With some effort, you can probably get contour lines on each little scatter plot. If you do this, you should post your code for it.)

So now we have contour lines for our model's PDF. We plot those over a scatterplot of the real data. And we ask, does the density of the points match the contours? Is the density between each pair of contour lines fairly uniform? Are the points bunched up where the peaks of our model's density contours are?

This post is visible to everyone.

Add a Response

2 responses

rrothorn
6 days ago

It was the only sentence I was not sure what was asked. I appreciate the way you explained this and make me understand.

Thank you, rrothorn!

posted 6 days ago by **ptressel** (Community TA)

Add a comment

abhishekdly
3 days ago

Thank you ptressel.The way you explain this is impressive.I am glad that we have TAs like you