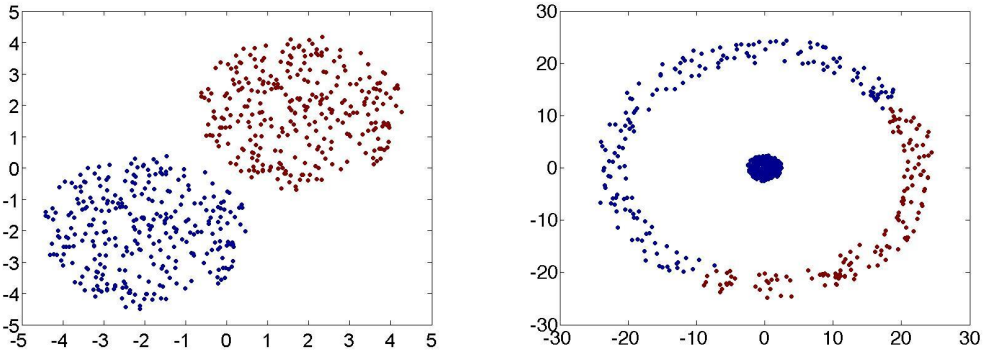

Figure 1: Sample clusters from k-emans

10-701/15-781, Machine Learning: Homework 3

Eric Xing, Tom Mitchell, Aarti Singh
Carnegie Mellon University

- The assignment is due at 10:30am (beginning of class) on **Wed, Jan 20, 2010**.
- Separate you answers into three parts, one for each TA, and put them into 3 piles at the table in front of the class. Don't forget to put both your name and a TA's name on each part.
- If you have question about any part, please direct your question to the respective TA who design the part.

1 Clustering [50 pt, Field Cady]

Clustering means partitioning your data into "natural" groups, usually because you suspect points in a cluster have something in common. The EM algorithm and k-means are two common algorithms (there are many others). This problem will have you implement these algorithms, and explore their limitations.

The datasets for you to use are available online, along with a Matlab script for loading them. Ask me if you're having any trouble with it. Instructions for submitting code will be posted later.

You can use any language for your implementations, but you may not use libraries which already implement these algorithms (you can, however, use fancy built-in mathematical functions, like Matlab or Mathematica provide).

1.1 K-means : implementation [15 pt]

In k-means clustering, the goal is to pick your clusters such that you minimize the sum, over all points x, of $|x - c_x|^2$, where c_x is the mean of the cluster containing x; this should remind you of least-squares line fitting. K-means clustering is NP-hard, but in practice Lloyd's algorithm, also called the "k-means algorithm", works extremely well.

Implement Lloyd's algorithm, and apply it to the datasets provided. Plot each dataset, indicating for each point which cluster is was placed in.

Answer :

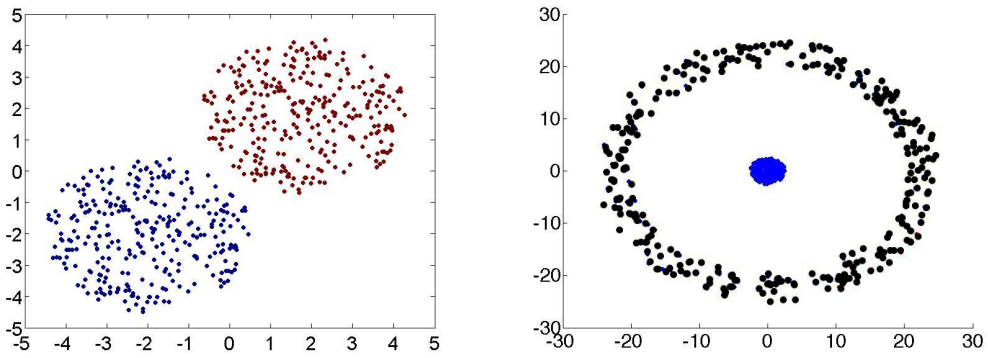

Figure 2: Sample clusters from the EM algorithm

Your outcome for dataset 1 should look very clean; two nice round clusters. For dataset 2, it should be messy in some way, and might be highly sensitive to yoru starting values. Figure 1 shows sample outputs.

1.2 K-means : evaluation [5 pt]

How well do you think k-means did for each dataset? Explain, intuitively, what (if anything) went badly and why.

Answer : K-means should do well on dataset 1 and poorly on dataset 2. This is because k-means looks for nice round clusters, and there aren't two of them in dataset 2.

1.3 EM Algorithm : implementation [25 pt]

A disadvantage of k-means is that the clusters cannot overlap at all. Expectation maximization deals with this by only probabilistically assigning points to clusters.

The thing to understand about the EM algorithm is that it's a special case of MLE; you have some data, you assume a parameterized form for the probability distribution (a mixture of Gaussians is, after all, an exotic parameterized probability distribution), and then you pick the parameters to maximize the probability of your data. But the usual MLE approach, solving $\frac{\partial P(X|\theta)}{\partial \theta} = 0$, isn't tractable, so we use the iterative EM algorithm to find θ. The EM algorithm is guaranteed to converge to a local optimum (I'm resisting the temptation to make you prove this :)).

Implement the EM algorithm, and apply it to the datasets provided. Assume that the data is a mixture of two Gaussians; you can assume equal mixing ratios. What parameters do you get for each dataset? Plot each dataset, indicating for each point which cluster it was placed in.

Answer :

Your outcome for dataset 1 should look very clean; two nice round clusters.

For dataset 2, you should be able discern that the ring is different from the blob in the middle. This is because the means of your gaussians are close together, but one of them has very large variance to capture the circle. Figure 2 shows sample outputs.

1.4 EM Algorithm : evaluation [5 pt]

Modeling dataset 2 as a mixture of gaussians is unrealistic, but the EM algorithm still gives *an* answer. Is there anything "fishy" about your answers which suggests something is wrong?

We usually do the EM algorithm with mixed Gaussians, but you can use any distributions; a Gaussian and a Laplacian, three exponentials, etc. Write down the formula for a parameterized probability density suitable for modeling "ring-shaped" clusters in 2D; don't let the density be 0 anywhere. You don't need to work out the EM calculations for this density, but you would if this came up in your research.

Answer : The fact that the means of the two clusters are so close is suspicious; the mean of each cluster is located where the other cluster is almost its densest.

There are many distributions which give a ring shape. Here is one with 3 parameters x_0, y_0 and r_0, for the center and radius of the ring:

$$f(x,y) = \frac{1}{\gamma}\left(\frac{1}{1+(\sqrt{x^2+y^2}-r_0)^2}\right) \tag{1}$$

where γ, the normalization constant, is

$$\begin{aligned}
\gamma &= \int_x\int_y \left(\frac{1}{1+(\sqrt{x^2+y^2}-r_0)^2}\right) dydx \\
&= \int_\theta\int_r \left(\frac{1}{1+(r-r_0)^2}\right) rdrd\theta \\
&= 2\pi\int_r \left(\frac{1}{1+(r-r_0)^2}\right) rdr \\
&= 2\pi\int_{r=0}^{\infty} \left(\frac{r}{1+(r-r_0)^2}\right) dr \\
&= 2\pi\left[2\sqrt{3}(r_0+1)Arctan\left(\frac{-2r_0+2r-1}{\sqrt{3}}\right)(r_0-1)\left(\log((r-r_0)^2+r_0-r+1)-2\log(r-r_0+1)\right)\right]_0^{\infty} \\
&= 2\pi\left[\left(2\sqrt{3}(r_0+1)\pi/2\right)-\left(2\sqrt{3}(r_0+1)Arctan(-(1+2r_0)/\sqrt{3})-(r_0-1)\log(\frac{r_0^2+r_0+1}{r_0^2-2r_0+1})\right)\right]
\end{aligned}$$

Yeah, this is uglier than I intended, but it is straightforward.

1.5 Extra Credit [5 pt]

With high-dimensional data we cannot perform visual checks, and problems can go unnoticed if we assume nice round, filled clusters. Describe in words a clustering algorithm which works even for weirdly-shaped clusters with unknown mixing ration. However, you can assume that the clsuters do not overlap at all, and that you have a LOT of training data. Discuss the weaknesses of your algorithm. Don't work out the details for this problem; just convince me that you know the basic idea and understand its limitations. Please keep your answers concise.

Hint : Nothing we've covered so far in class will help you here; think about graphs...

Answer :

A hierarchical approach. Start each point as its own connected component. Then while there is more than 1 connected component, draw a new edge to conenct the closest connecting components (the shortest such edge possible), keeping track of what the components were and how long the edges you added were. The last few edges you added should have been much longer than the previous edges; the connected components before you added these edges were the correct clusters.

This algorithm can fail catastrophically if there are outlier point which cause 2 clusters to merge prematurely.