11-755: MLSP

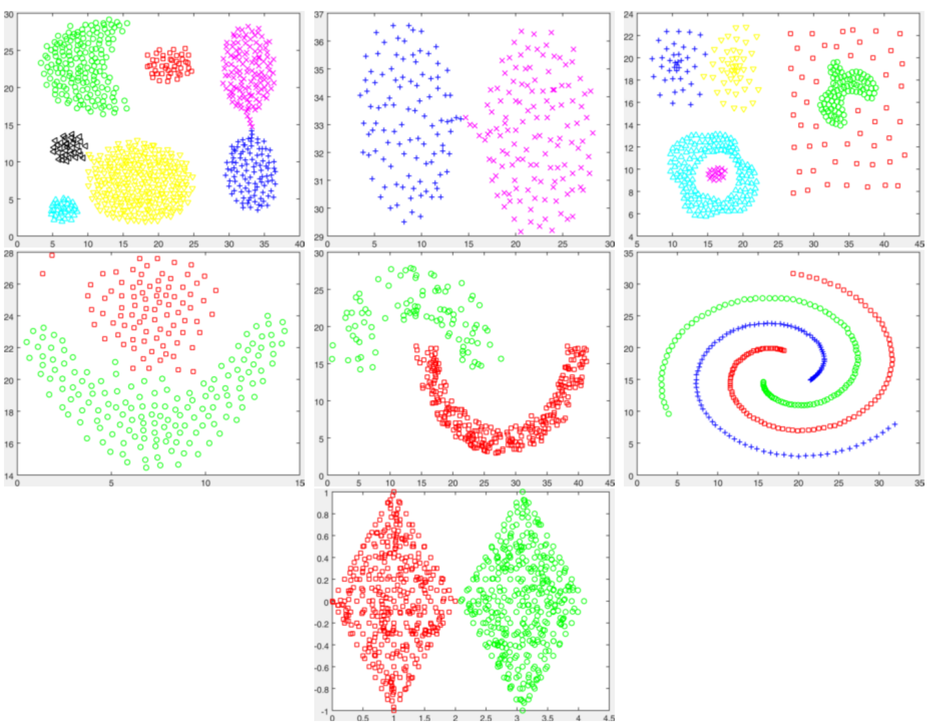
HW #3: Clustering and EM

Nikolas Wolfe

December 15, 2015

Problem 1: K-Means & Spectral Clustering

You are given a number of toy datasets. The visulized groud truth clustering configuration of these data: Aggregation (K=7), Bridge (K=2), Compound (K=6), Flame (K=2), Jain (K=2), Spiral (K=3) and TwoDiamond (K=2) are shown as follows:



Answer:

I implemented this code in Java, using the Efficient Java Matrix Library (EJML) package to do matrix multiplication and Eigen/Singular Value decomposition. In general, spectral clustering works a lot better than K-means and is able to determine a lot of structure (such as the spiral pattern) that k-means is unable to find.

In terms of evaluation, I ran the clustering algorithms and generated the confusion matrix between the most common label that emerges in each cluster and their actual labels. A diagonal matrix in this case is of course perfect performance. The accuracy can be determined from looking at the counts of labeled points for each class.

Dataset	Optimal K	Group Counts
Aggregation	7	$[K_1, K_2, K_3, K_4, K_5, K_6, K_7] = [45, 170, 102, 273, 34, 130, 34]$
Compound	6	$[K_1, K_2, K_3, K_4, K_5, K_6] = [50, 92, 38, 45, 158, 16]$
Spiral	3	$[K_1, K_2, K_3] = [101, 105, 106]$
Bridge	2	$[K_1, K_2] = [102, 13]$
Flame	2	$[K_1, K_2] = [87, 153]$
Jain	2	$[K_1, K_2] = [276, 97]$
Two Diamonds	2	$[K_1, K_2] = [400, 400]$

With this information, we can directly evaluate the quality of the Spectral and K-means clustering algorithms by simply comparing their diagonal of their confusion matrices (most common group label vs. actual label) to the vectors listed above above. To run the java package, simply run the following from the root of this project:

```
java -jar nwolfe_hw3.jar
```

This will run the main method in the class `ClusterDriver.java`, which will then iterate through a small range of σ values which are known to produce generally good spectral clusters, though it is often the case that several random restarts are required in order for the spectral clustering algorithm to settle into the optimal cluster groupings. In this report the best clusters have been chosen, though it is the case that when *you* run it, it might produce an optimal grouping.

Questions

In your report, answer the following questions:

1. Is the objective function of k-means a convex one?

   Answer: The objective function which we are trying to minimize is the sum of Euclidean distances from each cluster centroid to each cluster datapoint. However, the number of clusters K is discrete, which means we cannot take the derivative of this function and determine the optimal grouping (i.e. the minimum distance sum over all clusters) either analytically or through gradient descent or some other minimization technique. (This is also the case for spectral clustering.) Furthermore, the 2nd derivative of the Euclidean distance function is not zero. Because there is no way to find the optimal grouping (and perhaps no grouping actually exists), this is not a convex optimization problem.

2. Can the iterative updating scheme in k-means return a solution which achieves the global minimum of the k-means objective function?

   Answer: Newp. No. Sorry! As stated above, there is no guaranteed minimum for the objective function because it isn't convex. There may be an infinite number of viable ways to cluster the data.

3. If yes, why? If not, why not and are there any practical ways to partially fix this?

   Answer: One of the ways to fix this is, first off, to attempt to pick the initialization of the cluster centroids according to some logic. In this case we happen to know the optimal value of K from the labeled data. This may result in a better teasing apart of the datapoints and less moving around before the algorithm converges. Another way might be to use bottom up or top down clustering, i.e. to start with a single cluster containing all the data, perturb the centroid and then continually break the clusters apart until an optimal value of K is reached. Or we could start with a large number of clusters and then continually merge them according to some criteria until we reach a desired value of K.

Results

For each problem I have printed the output of the algorithm after a single run...

Aggregation: Spectral Clustering

```
Reading data values in data/Aggregation.csv...
```

```
Reading data values in data/Bridge.csv...
Reading data values in data/Compound.csv...
Reading data values in data/Flame.csv...
Reading data values in data/Jain.csv...
Reading data values in data/Spiral.csv...
Reading data values in data/TwoDiamonds.csv...

=============== AGGREGATION: SPECTRAL CLUSTERING ===============
With sigma = 0.47

label  1    2    3    4    5    6    7
-----------------------------------------------------------
1 |    45   0    0    0    0    0    0
2 |    0    170  0    0    0    0    0
3 |    0    0    1    0    0    0    0
4 |    0    0    0    270  0    0    0
5 |    0    0    0    0    34   0    0
6 |    0    0    101  0    0    130  0
7 |    0    0    0    3    0    0    34

With sigma = 0.48

label  1    2    3    4    5    6    7
-----------------------------------------------------------
1 |    45   0    0    0    0    0    0
2 |    0    170  0    0    0    0    0
3 |    0    0    102  0    0    2    0
4 |    0    0    0    270  0    0    0
5 |    0    0    0    0    34   0    0
6 |    0    0    0    0    0    128  0
7 |    0    0    0    3    0    0    34

With sigma = 0.49

... Output truncated ...
```

In the second run above, the diagonal is almost perfect, save for a few items misplaced in groups 3 and 7.

Aggregation: K-Means Clustering

```
==================== AGGREGATION: K MEANS ====================
```

```
label  1    2    3    4    5    6    7
------------------------------------------------------------
1 |    0    0    0    0    0    0    0
2 |    17   170  0    0    0    0    0
3 |    0    0    102  0    0    3    0
4 |    0    0    0    273  34   0    34
5 |    0    0    0    0    0    0    0
6 |    28   0    0    0    0    127  0
7 |    0    0    0    0    0    0    0

-----------------------------------------------------------
```

In the k-means clustering it's clear that several groups did not even emerge from the data (though this does not mean they were empty.)

Bridge: Spectral Clustering

```
=============== BRIDGE: SPECTRAL CLUSTERING ===============
With sigma = 0.42

... Output truncated ...

With sigma = 0.44

label    1    2
-------------------
1 |      102  2
2 |      0    128

With sigma = 0.45

label    1    2
-------------------
1 |      58   58
2 |      44   72
```

In one case the grouping was nearly perfect, and in another the data was more or less spread around a different kind of grouping.

Bridge: K-Means Clustering

```
==================== BRIDGE: K MEANS =====================

label    1    2
-------------------
1 |      0    0
2 |      102  130

----------------------------------------------------------
```

Again, K-means was not able to find the right clustering...

Compound: Spectral Clustering

```
=============== COMPOUND: SPECTRAL CLUSTERING ===============

With sigma = 0.154

label  1    2    3    4    5    6
---------------------------------------------------
1 |    21   0    1    1    0    0
2 |    6    92   0    0    0    0
3 |    12   0    33   0    0    0
4 |    3    0    1    44   0    0
5 |    5    0    2    0    158  0
6 |    3    0    1    0    0    16

With sigma = 0.155

label  1    2    3    4    5    6
---------------------------------------------------
1 |    20   0    2    1    0    0
2 |    6    92   1    44   0    0
3 |    11   0    35   0    0    0
4 |    0    0    0    0    0    0
5 |    7    0    0    0    158  0
6 |    6    0    0    0    0    16

... Output truncated ...
```

This grouping is particularly interesting because the groups in the compound picture were very difficult to tease apart, and yet we see here a diagonal clearly taking shape, except for some overlap with group 1.

Compound: K-Means Clustering

```
==================== COMPOUND: K MEANS ====================

label  1    2    3    4    5    6
--------------------------------------------------
1 |    0    0    0    0    0    0
2 |    48   92   0    0    0    0
3 |    0    0    38   1    0    0
4 |    2    0    0    44   0    0
5 |    0    0    0    0    158  16
6 |    0    0    0    0    0    0

---------------------------------------------------------
```

Aggregation: Flame Clustering

```
=============== FLAME: SPECTRAL CLUSTERING ===============
With sigma = 0.735

label    1    2
-------------------
1 |      0    0
2 |      87   153
```

Flame: K-Means Clustering

```
==================== FLAME: K MEANS ====================

label    1    2
-------------------
1 |      0    0
2 |      87   153

---------------------------------------------------------
```

Jain: Spectral Clustering

```
=============== JAIN: SPECTRAL CLUSTERING ===============

With sigma = 0.303

label    1    2
-------------------
1 |      276  0
2 |      0    97

With sigma = 0.304

label    1    2
-------------------
1 |      276  97
2 |      0    0

... Output truncated ...
```

Jain: K-Means Clustering

```
===================== JAIN: K MEANS =====================

label    1    2
-------------------
1 |      276  97
2 |      0    0

-----------------------------------------------------------
```

Spiral: Spectral Clustering

```
=============== SPIRAL: SPECTRAL CLUSTERING ===============
With sigma = 0.2

label  1    2    3
---------------------------
```

```
1 |     101  0    0
2 |     0    105  0
3 |     0    0    106

With sigma = 0.30000000000000004

label  1    2    3
---------------------------
1 |     101  0    0
2 |     0    105  0
3 |     0    0    106

With sigma = 0.4

label  1    2    3
---------------------------
1 |     101  0    0
2 |     0    105  0
3 |     0    0    106

... Output truncated ...
```

Given the shape in this case we can really see the difference between spectral clustering and k-means. This grouping was found perfectly, all zeros in the off-diagonal elements, with even a variable number of settings for σ. Looking at the k-means result below, it's clear that there's a real distinction here.

Spiral: K-Means Clustering

```
===================== SPIRAL: K MEANS =====================

label  1    2    3
---------------------------
1 |     66   54   53
2 |     0    0    0
3 |     35   51   53

-----------------------------------------------------------
```

Two Diamonds: Spectral Clustering

```
============== TWO DIAMONDS: SPECTRAL CLUSTERING ==============
With sigma = 20.0

label    1    2
-------------------
1 |      400  0
2 |      0    400

With sigma = 21.0

label    1    2
-------------------
1 |      204  198
2 |      196  202

... Output truncated ...
```

Two Diamonds: K-Means Clustering

```
==================== TWO DIAMONDS: K MEANS ====================

label    1    2
-------------------
1 |      203  196
2 |      197  204

----------------------------------------------------------
```

Questions & Conclusion

Compare your spectral clustering results with k-means. It is natural that on certain hard toy examples, both method won't generate perfect results? In your report, briefly analyze what is the advantage or disadvantage of spectral clustering over k-means. Why it is the case?

Answer:

It is natural in the sense that the Gaussian kernel function used in spectral clustering is good at separating certain kinds of data and not others. Clearly separable or asymmetric groups, even if they have complicated boundaries (such as the spiral) are easy to break apart through projection into a higher dimensional space. In the case of something like the flame or the two diamonds, neither algorithm could solve the problem well, because the spaces are contiguous, and it is not clear (at least from the perspective of the algorithm, which cannot see the coloring or labels) where to draw the boundaries. In these cases, different initializations will result in entirely different groupings. Furthermore in many cases it required a lot of random restarts before the groups coalesced into something reasonable.

The data you're working with is the best way to decide which algorithm to use for clustering. If you're worried about contiguous spaces, rather than absolute distance from a centroid, then spectral clustering is the way to go. On the other hand, it can be computationally expensive to use spectral clustering given the large matrix multiplication involved. K-means is very quick, easy, and fast. Furthermore, it can be difficult to try something like bottom-up or top-down clustering with spectral clustering, which is highly dependent on one's initialization of k and fairly rigid afterwards. K-means is flexible in the sense that it's very easy to change the number of clusters, recompute distances, and use certain ad-hoc mechanisms for getting a better estimate for the right value of k after initialization.

With spectral clustering I also found that SVD was better than eigen decomposition at finding the basis vectors for the projection of the data points into k dimensional space.

Problem 2: Expectation Maximization

Let Z be the sum of two random variables X and Y, $(Z = X + Y)$, where X and Y are drawn independently from below discrete probability distributions with probability mass functions defined as:

$$P(X = n) = P1(1 - P1)^{n-1} \tag{1}$$

$$P(Y = n) = P2(1 - P2)^{n-1} \tag{2}$$

Given samples of Z, derive an EM algorithm to estimate $P1$ and $P2$.

Answer: We have two discrete probability distributions. Samples of Z are always the sum of X and Y, so Z is a multinomial distribution. We don't know $P1$ and $P2$ from observing Z, so we have to *fractionate* the observations we get for Z.