2. K-means

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

K-means

1.0/1.0 point (graded)

For this part of the project you will compare clustering obtained via K-means to the (soft) clustering induced by EM. In order to do so, our K-means algorithm will differ a bit from the one you learned. Here, the means are estimated exactly as before but the algorithm returns additional information. More specifically, we use the resulting clusters of points to estimate a Gaussian model for each cluster. Thus, our K-means algorithm actually returns a mixture model where the means of the component Gaussians are the K centroids computed by the K-means algorithm. This is to make it such that we can now directly plot and compare solutions returned by the two algorithms as if they were both estimating mixtures.

Read a 2D toy dataset using `X = np.loadtxt('toy_data.txt')` . Your task is to run the K-means algorithm on this data using the implementation we have provided in `kmeans.py` Initialize K-means using `common.init(X, K, seed)` , where K is the number of clusters and `seed` is the random seed used to randomly initialize the parameters.

Note that `init(X,K)` returns a K-component mixture model with means, variances and mixing proportions. The K-means algorithm will only care about the means, however, and returns a mixture that is retrofitted based on the K-means solution.

Try $K = [1, 2, 3, 4]$ on this data, plotting each solution using our `common.plot` function. Since the initialization is random, please use seeds $0, 1, 2, 3, 4$ to and select the one that minimizes the total cost. Save the associated plots (best solution for each K). The code for this task can be written in `main.py` .

Report the lowest cost for each K:

$\text{Cost}|_{K=1} =$ 5462.297452340002 ✔ **Answer:** 5462.2974

$\text{Cost}|_{K=2} =$ 1684.9079502962372 ✔ **Answer:** 1684.9079

$\text{Cost}|_{K=3} =$ 1329.59486715443 ✔ **Answer:** 1329.5948

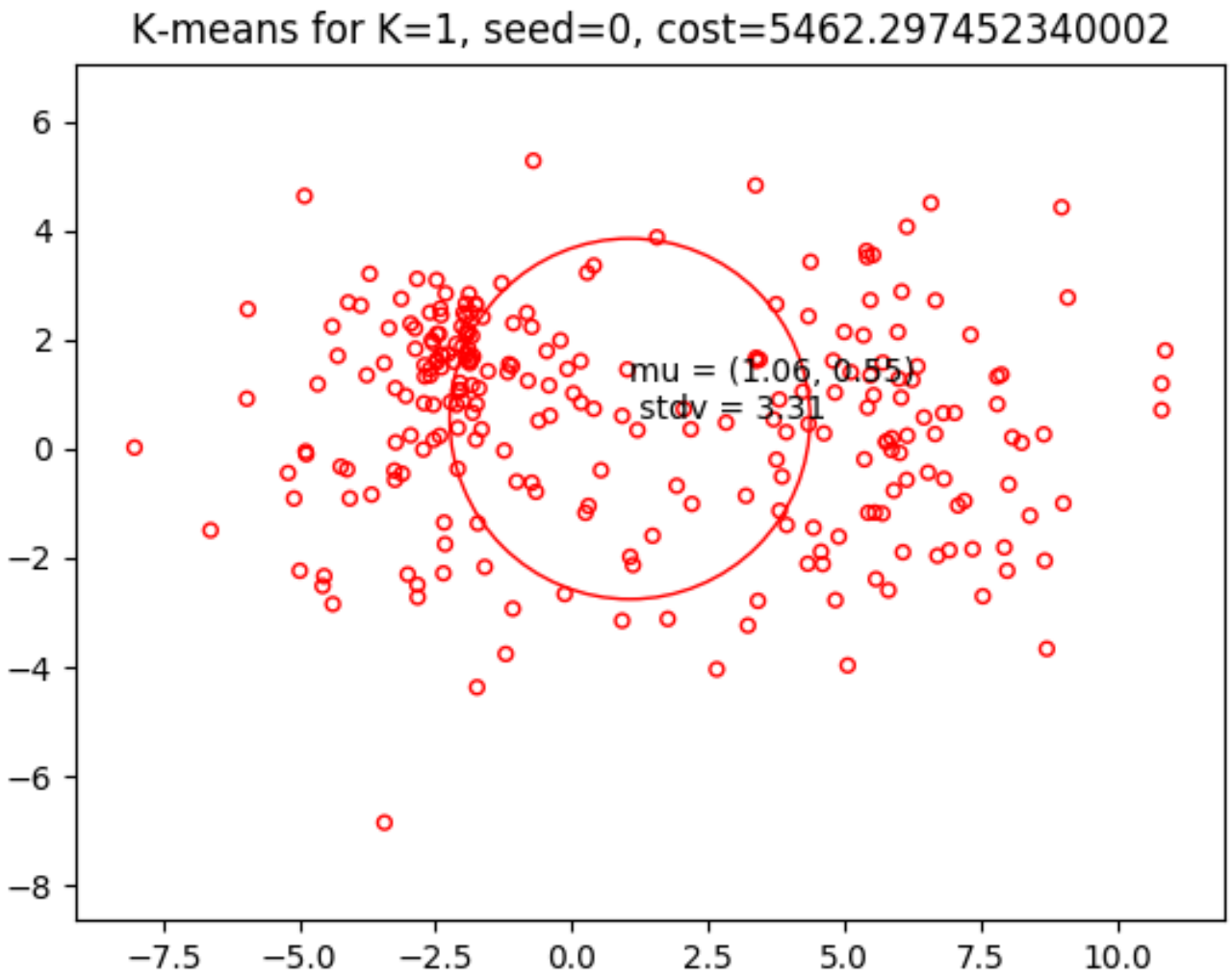
$\text{Cost}|_{K=4} =$ 1035.499826539466 ✔ **Answer:** 1035.4998

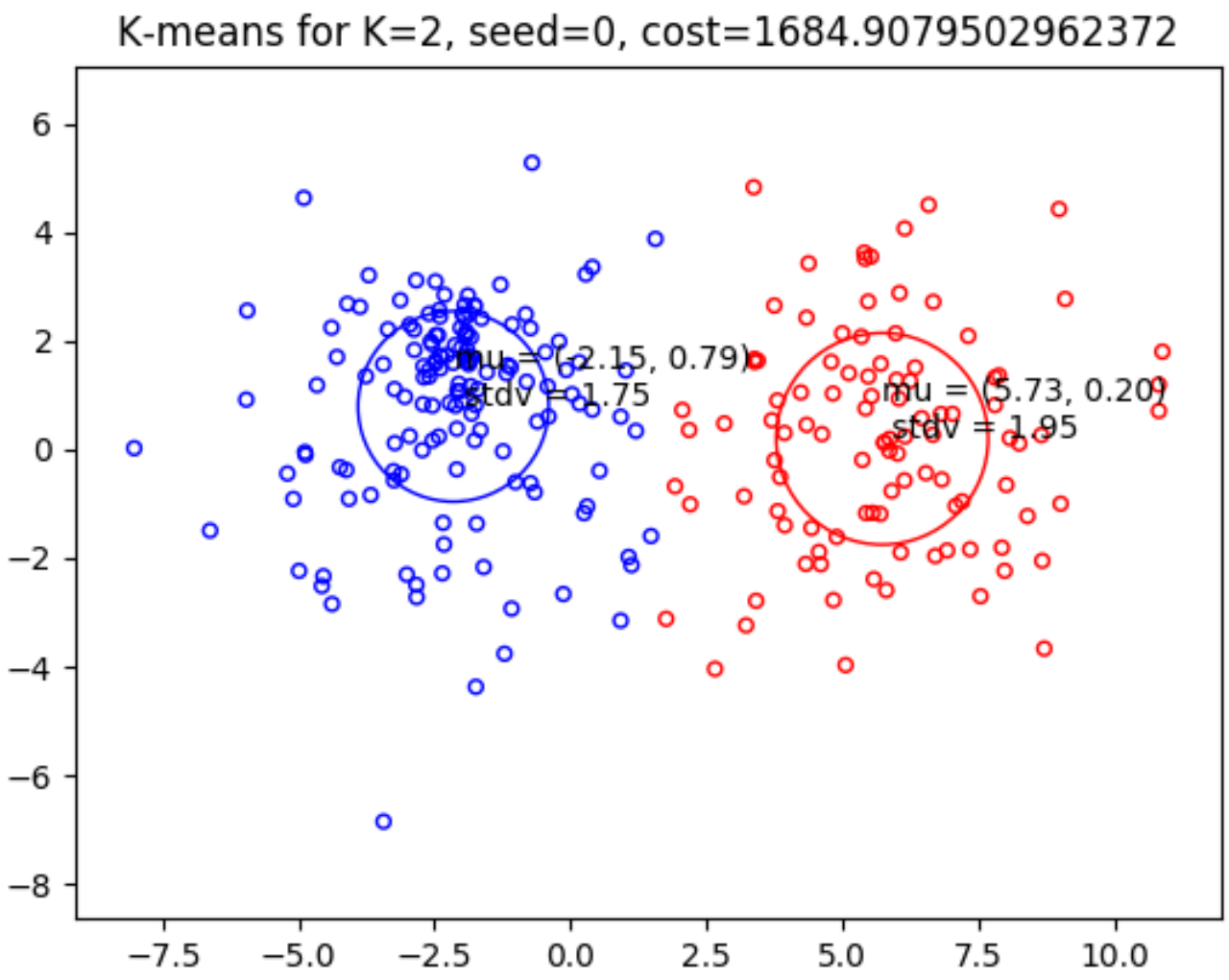
Solution:

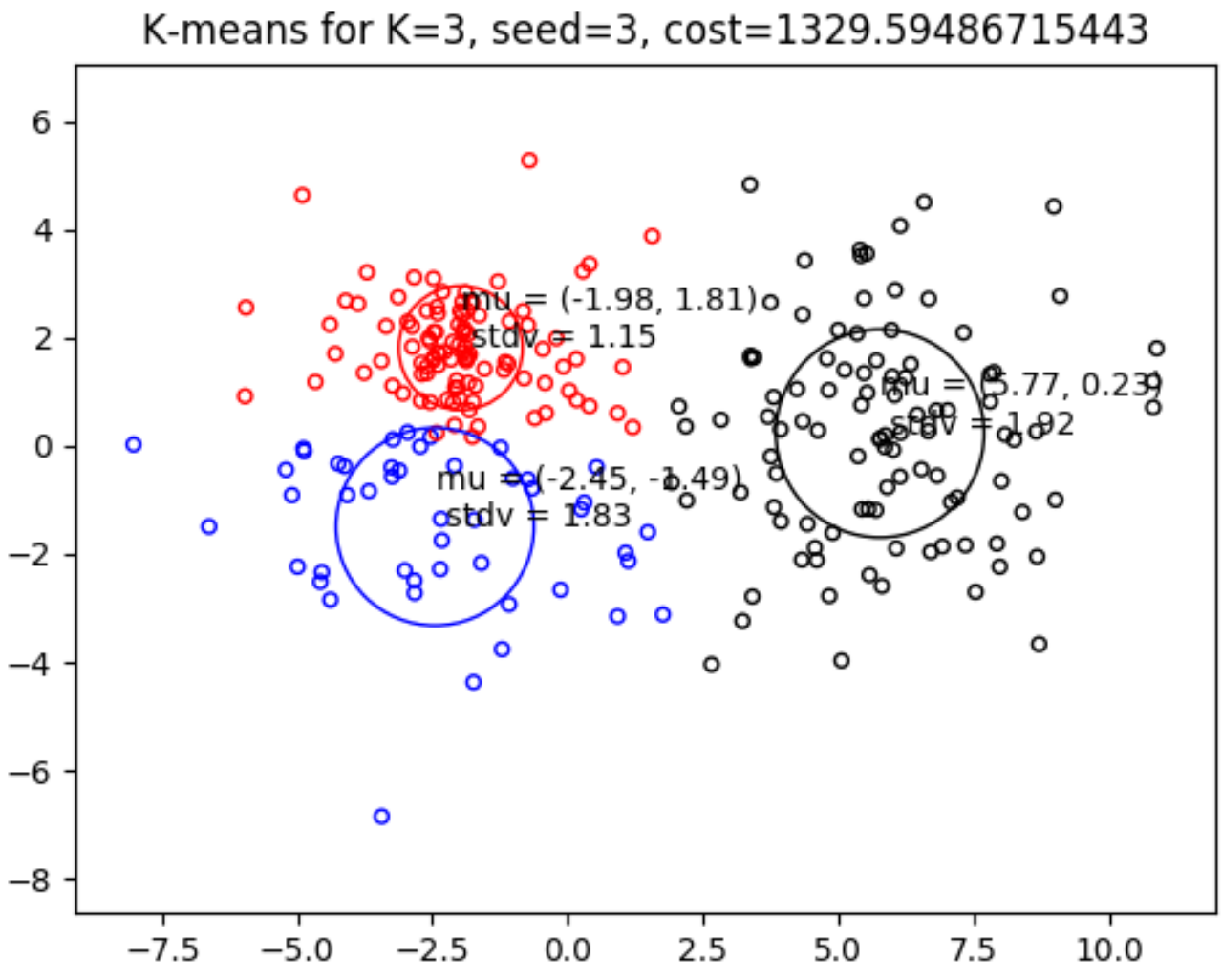
Code:

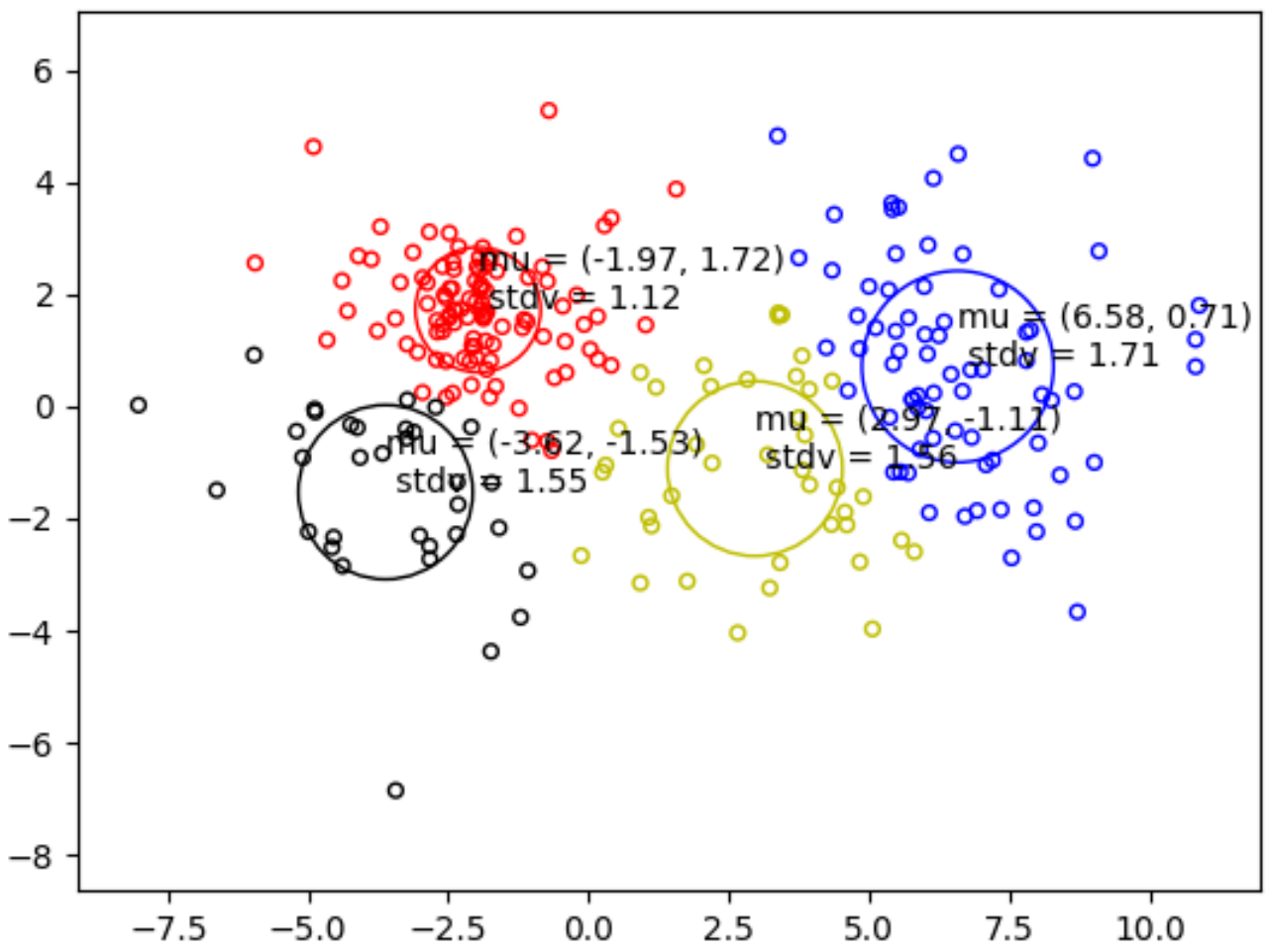
```
def run_kmeans():
    for K in range(1, 5):
        min_cost = None
        best_seed = None
        for seed in range(0, 5):
            mixture, post = common.init(X, K, seed)
            mixture, post, cost = kmeans.run(X, mixture, post)
            if min_cost is None or cost < min_cost:
                min_cost = cost
                best_seed = seed

        mixture, post = common.init(X, K, best_seed)
        mixture, post, cost = kmeans.run(X, mixture, post)
        title = "K-means for K=, seed=, cost=".format(K, best_seed, min_cost)
        print(title)
        common.plot(X, mixture, post, title)
```

Plots:


K-means for K=1, seed=0, cost=5462.297452340002
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


K-means for K=2, seed=0, cost=1684.9079502962372
mu = (-2.15, 0.79)
stdv = 1.75
mu = (5.73, 0.20)
stdv = 1.95
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 1 of 20 attempts

Answers are displayed within the problem

Discussion

Show Discussion

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

© All Rights Reserved