Anthony Tummillo

HW Report #9

Problem 1.

Part a.)
Mean of cluster S1: (0,2.5)
Mean of cluster S2: (6.5,3.5)

Datapoints:
(0,0) -> Cluster S1
(0,5) -> Cluster S1
(6,7) -> Cluster S2
(7,0) -> Cluster S2

Part b.)
Mean of cluster S1: (2,4)
Mean of cluster S2: (7,0)

Datapoints:
(0,0) -> Cluster S1
(0,5) -> Cluster S1
(6,7) -> Cluster S1
(7,0) -> Cluster S2

Part c.)
K-means clustering minimizes the **sum of squared center-point distances** for all clusters.

The squared center-point distances are found using the equation

$$\sum_{i=1}^{k} \sum_{x_j \in S_i} || x_j - u_i ||^2$$

so, the mathematical expression I would use to compare different clusterings and select the best one is

$$\arg\min_S \sum_{i=1}^{k} \sum_{x_j \in S_i} || x_j - u_i ||^2 \qquad u_i = \text{center of cluster } S_i$$

This expression states that you should select the clustering (S) whose clusters (S_1 ,…, S_k) gives the lowest sum of squared center-point distances.

Using Euclidean distance this expression determines that the Clustering from part a.) - starting with initial means (0,0) and (7,0) - is the best one.

Problem 2.

Part a.)

```
Clustering: 1
        Cluster 1 size: 98
        Cluster 2 size: 82
Clustering: 2
        Cluster 1 size: 144
        Cluster 2 size: 36
Clustering: 3
        Cluster 1 size: 1
        Cluster 2 size: 179
Clustering: 4
        Cluster 1 size: 60
        Cluster 2 size: 120
Clustering: 5
        Cluster 1 size: 124
        Cluster 2 size: 56
Clustering: 6
        Cluster 1 size: 179
        Cluster 2 size: 1
Clustering: 7
        Cluster 1 size: 135
        Cluster 2 size: 45
Clustering: 8
        Cluster 1 size: 110
        Cluster 2 size: 70
Clustering: 9
        Cluster 1 size: 179
        Cluster 2 size: 1
Clustering: 10
        Cluster 1 size: 88
        Cluster 2 size: 92
Clustering: 11
        Cluster 1 size: 95
        Cluster 2 size: 85
```

```
Clustering: 12
          Cluster 1 size: 86
          Cluster 2 size: 94
Clustering: 13
          Cluster 1 size: 161
          Cluster 2 size: 19
Clustering: 14
          Cluster 1 size: 105
          Cluster 2 size: 75
Clustering: 15
          Cluster 1 size: 2
          Cluster 2 size: 178
Clustering: 16
          Cluster 1 size: 178
          Cluster 2 size: 2
Clustering: 17
          Cluster 1 size: 80
          Cluster 2 size: 100
Clustering: 18
          Cluster 1 size: 97
          Cluster 2 size: 83
Clustering: 19
          Cluster 1 size: 168
          Cluster 2 size: 12
Clustering: 20
          Cluster 1 size: 1
          Cluster 2 size: 179
Clustering: 21
          Cluster 1 size: 132
          Cluster 2 size: 48
Clustering: 22
          Cluster 1 size: 1
          Cluster 2 size: 179
Clustering: 23
          Cluster 1 size: 179
          Cluster 2 size: 1
Clustering: 24
          Cluster 1 size: 29
          Cluster 2 size: 151
Clustering: 25
          Cluster 1 size: 172
          Cluster 2 size: 8
```

```
Clustering: 26
        Cluster 1 size: 12
        Cluster 2 size: 168
Clustering: 27
        Cluster 1 size: 134
        Cluster 2 size: 46
Clustering: 28
        Cluster 1 size: 175
        Cluster 2 size: 5
Clustering: 29
        Cluster 1 size: 130
        Cluster 2 size: 50
Clustering: 30
        Cluster 1 size: 167
        Cluster 2 size: 13
```

Using my formula from problem 1 part c.) I found that clustering #10 was the best with a sum of squared center-point distances of 8.8361e+03

Part b.)
In my method for finding initial seeds for the k-means algorithm I first found the maximum squared Euclidean distance between two points in the clustering_data dataset (735.1159). I then took half this distance (367.5580) and randomly selected two points from clustering_data until I found two points whose distance from one another was greater than or equal to 367.5580. Once a valid pair of points was found I used these points as my initial centers in the k-means algorithm.

Test Run:
Min_sum from a = 8.8361e+03
Min_sum from b = 8.8551e+03
For this run the clustering initialization in part a.) performed better.

This was just an initial test run to get an idea of what to expect before running the competition 100 times. I would hypothesize that my clustering initialization will tend to perform worse than the clustering initialization from part a.).

100 Run Results:
Run #1
Initialization from part a.) performed better
Minimum sum from part a.) implementation = 8.840627e+03
Run #2
Initialization from part a.) performed better
Minimum sum from part a.) implementation = 8.830180e+03
Run #3

```
Initialization from part a.) performed better
Minimum sum from part a.) implementation = 8.835818e+03
Run #4
Initialization from part a.) performed better
Minimum sum from part a.) implementation = 8.829875e+03
Run #5
Initialization from part a.) performed better
Minimum sum from part a.) implementation = 8.833348e+03
Run #6
Initialization from part a.) performed better
Minimum sum from part a.) implementation = 8.832149e+03
Run #7
Initialization from part a.) performed better
Minimum sum from part a.) implementation = 8.834853e+03
Run #8
Initialization from part a.) performed better
Minimum sum from part a.) implementation = 8.831610e+03
Run #9
Initialization from part a.) performed better
Minimum sum from part a.) implementation = 8.834534e+03
Run #10
Initialization from part a.) performed better
Minimum sum from part a.) implementation = 8.833245e+03
Run #11
Initialization from part a.) performed better
Minimum sum from part a.) implementation = 8.833959e+03
Run #12
Initialization from part a.) performed better
Minimum sum from part a.) implementation = 8.836852e+03
Run #13
Initialization from part a.) performed better
Minimum sum from part a.) implementation = 8.836359e+03
Run #14
Initialization from part a.) performed better
Minimum sum from part a.) implementation = 8.837277e+03
Run #15
Initialization from part a.) performed better
Minimum sum from part a.) implementation = 8.833789e+03
Run #16
Initialization from part a.) performed better
Minimum sum from part a.) implementation = 8.833937e+03
Run #17
Initialization from part a.) performed better
Minimum sum from part a.) implementation = 8.833684e+03
```

```
Run #18
Initialization from part b.) performed better
Minimum sum from part b.) implementation = 8.831461e+03
Run #19
Initialization from part a.) performed better
Minimum sum from part a.) implementation = 8.830476e+03
Run #20
Initialization from part a.) performed better
Minimum sum from part a.) implementation = 8.827882e+03
Run #21
Initialization from part a.) performed better
Minimum sum from part a.) implementation = 8.831553e+03
Run #22
Initialization from part a.) performed better
Minimum sum from part a.) implementation = 8.835942e+03
Run #23
Initialization from part a.) performed better
Minimum sum from part a.) implementation = 8.828805e+03
Run #24
Initialization from part a.) performed better
Minimum sum from part a.) implementation = 8.836648e+03
Run #25
Initialization from part a.) performed better
Minimum sum from part a.) implementation = 8.835681e+03
Run #26
Initialization from part b.) performed better
Minimum sum from part b.) implementation = 8.831461e+03
Run #27
Initialization from part a.) performed better
Minimum sum from part a.) implementation = 8.861643e+03
Run #28
Initialization from part a.) performed better
Minimum sum from part a.) implementation = 8.832241e+03
Run #29
Initialization from part a.) performed better
Minimum sum from part a.) implementation = 8.829609e+03
Run #30
Initialization from part a.) performed better
Minimum sum from part a.) implementation = 8.834557e+03
Run #31
Initialization from part a.) performed better
Minimum sum from part a.) implementation = 8.828239e+03
Run #32
Initialization from part a.) performed better
```

```
Minimum sum from part a.) implementation = 8.828713e+03
Run #33
Initialization from part a.) performed better
Minimum sum from part a.) implementation = 8.829078e+03
Run #34
Initialization from part a.) performed better
Minimum sum from part a.) implementation = 8.836852e+03
Run #35
Initialization from part a.) performed better
Minimum sum from part a.) implementation = 8.833038e+03
Run #36
Initialization from part a.) performed better
Minimum sum from part a.) implementation = 8.835211e+03
Run #37
Initialization from part a.) performed better
Minimum sum from part a.) implementation = 8.831610e+03
Run #38
Initialization from part a.) performed better
Minimum sum from part a.) implementation = 8.835579e+03
Run #39
Initialization from part b.) performed better
Minimum sum from part b.) implementation = 8.831461e+03
Run #40
Initialization from part a.) performed better
Minimum sum from part a.) implementation = 8.835441e+03
Run #41
Initialization from part a.) performed better
Minimum sum from part a.) implementation = 8.831542e+03
Run #42
Initialization from part a.) performed better
Minimum sum from part a.) implementation = 8.837030e+03
Run #43
Initialization from part a.) performed better
Minimum sum from part a.) implementation = 8.832708e+03
Run #44
Initialization from part a.) performed better
Minimum sum from part a.) implementation = 8.828371e+03
Run #45
Initialization from part a.) performed better
Minimum sum from part a.) implementation = 8.832696e+03
Run #46
Initialization from part a.) performed better
Minimum sum from part a.) implementation = 8.833886e+03
Run #47
```

```
Initialization from part a.) performed better
Minimum sum from part a.) implementation = 8.827526e+03
Run #48
Initialization from part a.) performed better
Minimum sum from part a.) implementation = 8.832014e+03
Run #49
Initialization from part a.) performed better
Minimum sum from part a.) implementation = 8.828160e+03
Run #50
Initialization from part b.) performed better
Minimum sum from part b.) implementation = 8.831461e+03
Run #51
Initialization from part a.) performed better
Minimum sum from part a.) implementation = 8.830598e+03
Run #52
Initialization from part a.) performed better
Minimum sum from part a.) implementation = 8.837730e+03
Run #53
Initialization from part b.) performed better
Minimum sum from part b.) implementation = 8.831461e+03
Run #54
Initialization from part a.) performed better
Minimum sum from part a.) implementation = 8.838287e+03
Run #55
Initialization from part a.) performed better
Minimum sum from part a.) implementation = 8.832069e+03
Run #56
Initialization from part a.) performed better
Minimum sum from part a.) implementation = 8.826911e+03
Run #57
Initialization from part a.) performed better
Minimum sum from part a.) implementation = 8.831365e+03
Run #58
Initialization from part a.) performed better
Minimum sum from part a.) implementation = 8.831610e+03
Run #59
Initialization from part a.) performed better
Minimum sum from part a.) implementation = 8.829169e+03
Run #60
Initialization from part a.) performed better
Minimum sum from part a.) implementation = 8.829875e+03
Run #61
Initialization from part a.) performed better
Minimum sum from part a.) implementation = 8.831800e+03
```

```
Run #62
Initialization from part a.) performed better
Minimum sum from part a.) implementation = 8.830793e+03
Run #63
Initialization from part a.) performed better
Minimum sum from part a.) implementation = 8.838823e+03
Run #64
Initialization from part a.) performed better
Minimum sum from part a.) implementation = 8.833521e+03
Run #65
Initialization from part a.) performed better
Minimum sum from part a.) implementation = 8.834929e+03
Run #66
Initialization from part a.) performed better
Minimum sum from part a.) implementation = 8.833256e+03
Run #67
Initialization from part a.) performed better
Minimum sum from part a.) implementation = 8.832154e+03
Run #68
Initialization from part a.) performed better
Minimum sum from part a.) implementation = 8.831595e+03
Run #69
Initialization from part a.) performed better
Minimum sum from part a.) implementation = 8.833659e+03
Run #70
Initialization from part a.) performed better
Minimum sum from part a.) implementation = 8.835314e+03
Run #71
Initialization from part a.) performed better
Minimum sum from part a.) implementation = 8.836916e+03
Run #72
Initialization from part a.) performed better
Minimum sum from part a.) implementation = 8.839061e+03
Run #73
Initialization from part a.) performed better
Minimum sum from part a.) implementation = 8.829078e+03
Run #74
Initialization from part a.) performed better
Minimum sum from part a.) implementation = 8.835594e+03
Run #75
Initialization from part a.) performed better
Minimum sum from part a.) implementation = 8.831595e+03
Run #76
Initialization from part a.) performed better
```

```
Minimum sum from part a.) implementation = 8.845988e+03
Run #77
Initialization from part a.) performed better
Minimum sum from part a.) implementation = 8.834624e+03
Run #78
Initialization from part a.) performed better
Minimum sum from part a.) implementation = 8.831610e+03
Run #79
Initialization from part a.) performed better
Minimum sum from part a.) implementation = 8.837007e+03
Run #80
Initialization from part a.) performed better
Minimum sum from part a.) implementation = 8.831960e+03
Run #81
Initialization from part a.) performed better
Minimum sum from part a.) implementation = 8.837940e+03
Run #82
Initialization from part a.) performed better
Minimum sum from part a.) implementation = 8.832154e+03
Run #83
Initialization from part a.) performed better
Minimum sum from part a.) implementation = 8.841489e+03
Run #84
Initialization from part b.) performed better
Minimum sum from part b.) implementation = 8.831461e+03
Run #85
Initialization from part a.) performed better
Minimum sum from part a.) implementation = 8.831677e+03
Run #86
Initialization from part a.) performed better
Minimum sum from part a.) implementation = 8.832616e+03
Run #87
Initialization from part a.) performed better
Minimum sum from part a.) implementation = 8.826860e+03
Run #88
Initialization from part a.) performed better
Minimum sum from part a.) implementation = 8.832553e+03
Run #89
Initialization from part a.) performed better
Minimum sum from part a.) implementation = 8.840767e+03
Run #90
Initialization from part b.) performed better
Minimum sum from part b.) implementation = 8.831461e+03
Run #91
```

```
Initialization from part a.) performed better
Minimum sum from part a.) implementation = 8.837485e+03
Run #92
Initialization from part a.) performed better
Minimum sum from part a.) implementation = 8.838335e+03
Run #93
Initialization from part a.) performed better
Minimum sum from part a.) implementation = 8.838677e+03
Run #94
Initialization from part a.) performed better
Minimum sum from part a.) implementation = 8.835740e+03
Run #95
Initialization from part a.) performed better
Minimum sum from part a.) implementation = 8.832708e+03
Run #96
Initialization from part b.) performed better
Minimum sum from part b.) implementation = 8.831461e+03
Run #97
Initialization from part a.) performed better
Minimum sum from part a.) implementation = 8.835770e+03
Run #98
Initialization from part a.) performed better
Minimum sum from part a.) implementation = 8.831862e+03
Run #99
Initialization from part a.) performed better
Minimum sum from part a.) implementation = 8.829991e+03
Run #100
Initialization from part a.) performed better
Minimum sum from part a.) implementation = 8.831276e+03
The initialization from part a.) performed better: 92 times!
The initialization from part b.) performed better: 8 times!
The initializations tied: 0 times!
```

Based on these results from the 100 runs of the competition it appears that the clustering initialization from part a.) performs better on average (92 times out of 100) than the clustering initialization I have proposed and described above.

Part c.)

The measure of agreement I used was to calculate the purity of each cluster and then average the purities of the clusters together to get the final agreement score. To find the purity of each cluster I first determined which class label occurred most frequently for the data points in each cluster. After that I divided the total count of the chosen label occurring in each cluster and divided this count by the total number of data points in each cluster to obtain the purity for each cluster. Next I averaged the two purities in the clustering together and used this result as my final agreement score for comparing clusterings to class labels.

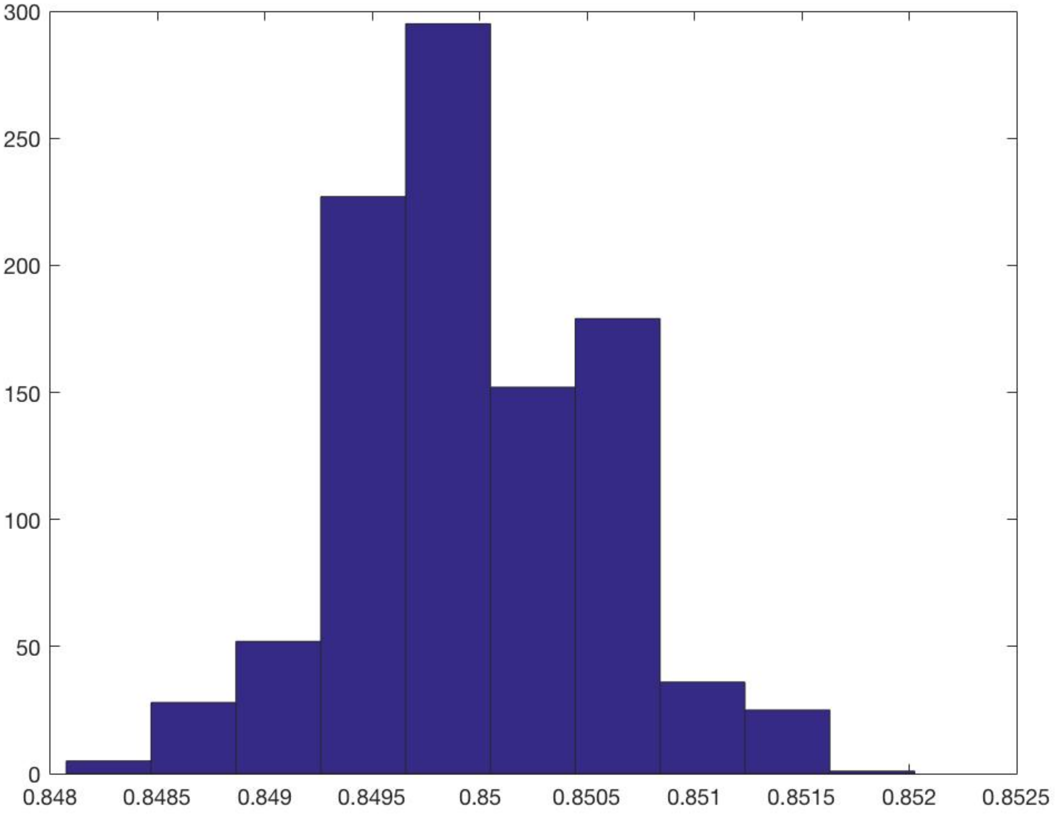
The code for this calculation is included in the calc_agr.m file that was submitted.

When I compared my clustering from part a.) and the class labels from class_labels.txt I calculated an agreement score of 0.8513

Because this value can only range from 0 to 1, and 0.8513 is far above 0.5 (0.5 would essentially represent no correlation between the two/random label selection) I would argue that the clustering and class_labels do agree and are related for majority of data points in the clustering_data dataset.

Part d.)
The code is written and submitted (titled Prob2_d)

True_label agreement score = 0.8513



Graph of distribution of agreement scores calculated from random permutations of class labels.

Number of agreement scores above true_label score: 26
Number of agreement scores below true_label score: 974
(both of these are out of the 1000 random permutations of class labels)

This means that the probability of our Clusterings relating to random labels better than the true class labels is 26/1000.

Because this probability is so low I would argue that the clustering is related to the true class labels and this relation is **not** random.