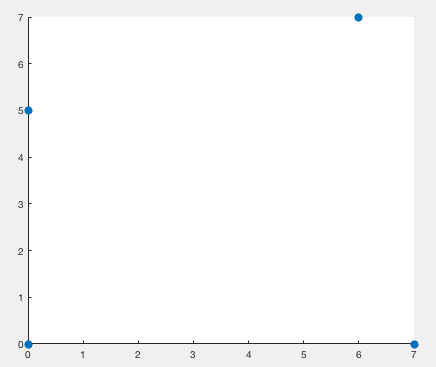
Problem assignment 9

Ang Li, anl125

**Problem 1. K-means clustering**

Assume we have a 2-dimensional space and four points (0, 0), (0, 5), (6, 7) and (7, 0) and we want to cluster the examples into two groups using the k-means algorithm and the Euclidean distance.



**Part a.** Let us assume the algorithm is initialized with means (0, 0) and (7, 0). What are the values of the two means the algorithm converges to.

How are datapoints divided into groups?

*data = [0, 0; 0,5; 6,7; 7,0];*

*For the initialized group mean at* ***g1\_mean=(0,0)*** *and* ***g2\_mean=(7,0)***

*The distance of data point (0, 5) to the group means:*

*%(0,5)*

*v1a = data(2,:) - g1\_mean;*

*dis1a = sqrt(v1a\*v1a'); %5*

*v2a = data(2,:) - g2\_mean;*

*dis2a = sqrt(v2a\*v2a'); %8.60*

*So for data point (0,5), it will be clustered into group 1*

*The distance of data point (6,7) to the group means:*

*%(6,7)*

*v1b = data(3,:) - g1\_mean;*

*dis1b = sqrt(v1b\*v1b'); %9.22*

*v2b = data(3,:) - g2\_mean;*

*dis2b = sqrt(v2b\*v2b'); %7.07*

*So for data point (6,7), it will be clustered into group 2*

*Now, group 1 has data: (0,5); (0,0). New mean for group 1: (0, 2.5)*

*Group 2 has data: (6,7); (7,0). New mean for group 2: (6.5, 3.5)*

**Part b.** Now let us assume the algorithm is initialized from the means (3, 3) and (7, 0). What are the values of the two means the algorithm converges to. How are datapoints divided into groups?

*For the initialized means as (3,3) and (7,0)*

*g1\_mean = [3,3];*

*g2\_mean = [7,0];*

*%(0,0)*

*v1a = data(1,:) - g1\_mean;*

*dis1a = sqrt(v1a\*v1a'); %4.24*

*v1b = data(1,:) - g2\_mean;*

*dis1b = sqrt(v1b\*v1b'); %7*

*So for data point (0,0), it will be clustered into group 1*

*%(0,5)*

*v2a = data(2,:) - g1\_mean;*

*dis2a = sqrt(v2a\*v2a'); %3.61*

*v2b = data(2,:) - g2\_mean;*

*dis2b = sqrt(v2b\*v2b'); %8.60*

*So for data point (0,5), it will be clustered into group 1*

*%(6,7)*

*v3a = data(3,:) - g1\_mean;*

*dis3a = sqrt(v3a\*v3a'); %5*

*v3b = data(3,:) - g2\_mean;*

*dis3b = sqrt(v3b\*v3b'); %7.07*

*So for data point (6,7), it will be clustered into group 1*

*For data point (7,0), it is in group 2*

*So we have new groups:*

*Group 1: (0,0), (0,5), (6,7). New mean: (2, 4)*

*Group 2; (7,0). New mean: (7,0)*

**Part c.** Let us assume the two runs of the k-means lead to two different clusterings. Write a math expression that would let you compare these different clusterings and pick the best one.

*The k-means minimizes the sum of squared center-point distances for all clusters. Therefore, to evaluate the two clusters, I will calculate the squared center-point distances for both clusters and pick the cluster with smaller measure.*

**Problem 2. Clustering experiments**

Please load the dataset clustering data.txt.

**Part a.** Run the k-means algorithm (implemented in Matlab in the function kmeans) for finding 2 clusters. Use Euclidean distance to define the differences in between the points. The kmeans procedure (if initial means seeds are not set) uses a random set of seeds in each run. Run the kmeans procedure (in the default mode) 30 times. Report the cluster sizes for these different runs? Use formula from Problem 1 Part c. to decide which clustering is the best.

*Group sizes for the 30 times run:*

*group\_size =*

*2 178*

*24 156*

*110 70*

*86 94*

*113 67*

*177 3*

*179 1*

*179 1*

*141 39*

*89 91*

*2 178*

*101 79*

*38 142*

*137 43*

*179 1*

*179 1*

*105 75*

*176 4*

*179 1*

*131 49*

*160 20*

*87 93*

*51 129*

*80 100*

*179 1*

*108 72*

*171 9*

*1 179*

*118 62*

*179 1*

*The point-to-centroid distances for the 30 times run:*

*measures =*

*1.0e+03 \**

*8.9487*

*8.9349*

*8.8583*

*8.9098*

*8.8610*

*8.8618*

*9.1265*

*9.1151*

*8.8902*

*8.9841*

*9.0639*

*8.8465*

*8.9899*

*8.9079*

*9.0981*

*9.1490*

*8.9113*

*8.8720*

*9.1490*

*8.8855*

*8.9314*

*8.8481*

*8.9635*

*8.8443*

*8.9954*

*8.9405*

*9.0409*

*9.2247*

*8.8571*

*The 24th run had the smallest point-to-central distances, therefore is the best cluster result.*

**Part b.** kmeans procedure implemented in Matlab allows you to control initial means’ seeds. Propose, describe and implement a method for generating initial seeds and run the k-means procedure for 30 times with this new seed procedure. To determine the best clustering from 30 runs use again the formula from Problem 1 Part c. Is your clustering initialization better in Part b than in Part a? Compare this by running the competition 100 times, such that each time, we initialize the k-means 30 times and compare the best clusters from each approach. Please analyze the results.

*Another method of selecting initial seed centroid is to perform a preliminary clustering phase on a random 10% subsample of X. This preliminary phase is itself initialized using 'sample'. I will use this method.*

***Cluter\_30times.m***

*The best clustering from this new method has the smallest point-to-centroid distance: 8.8443e+03, which is smaller than the result got from part A (default initialization seed)*

*After run the competition 100 times, I am able to calculate and pick the best cluster based on the point-to-centroid distance for each competition of two approaches. I then use paired t-test to compare the distance measures from two approaches.*

*[h,p] = ttest(min\_dists(:,1),min\_dists(:,2))*

*%no significant different*

*% p = 0.4253*

*However, there is no significant difference between two approaches, therefore, at this situation, we cannot decide which initiation approach is better than another.*

**Part c.** One reason for performing the clustering is to analyze data and see what data points fall into the same class. When performing this analysis you see the groups and you often try to make sense of the groups, that is, you try to see whether there is something common about the elements in the group.

One way to evaluate the clustering is to use an additional class (group) label related to some aspect of the data and compare the clustering results to these labels, that is, you want to see whether clusters agree with these labels. We provide one such set of labels in class ***labels.txt*** file. Our goal is to see if the clustering found matches well the labels in the class labels file. Both files are aligned accross rows.

To start the analysis, we need to define a measure (or score) reflecting the agreement between the clusters and labels. Propose a score that reflects how well the clusters match labels. Hint: the best cluster should have the representatives of just one class. After that please use the score to evaluate the clustering you found in Part 1. Do you think the clustering and the class labels agree?

*To evaluate how well the clustering result match the labeled class, we could use the idea of confusion matrix and precision rate. As the best cluster should have the representatives of just one class, we first need to find the majority class of the clustering results and the majority class of the labeled result. We can then assume that the clustering results with the majority class represent the labeled results’ majority class. The evaluation can be then calculated based on the confusion matrix of the agreement of the clustering results and the labelled class.*

*In our case, the labeled class from the data show that 153 of the data are in the class 0. The clustering result indicates that 160 data points are in one group. Therefore, we assume that 160 data from the clustering result should represent the 153 class 0 data. We then use the idea from confusion matrix and compare the agreement:*

*crosstab(clusterID, class\_labels)*

*>> crosstab(clusterID, class\_labels)*

*ans =*

*134 26*

*19 1*

*So there are 134 data points from 160 points are correctly clustered into one group. The precision rate is 134/160. They pretty much agree.*

**Problem 3. Feature/Input ranking**

Consider the dataset in file Data.txt. The dataset consists of 259 examples (in rows) where each example is defined by 70 dimensional input vector (represented in columns) and an associated binary label (in last column).

**Part a.** Write and submit a function **Fisher­­\_score(x, y)** that takes as arguments a vector of one-dimensional inputs x and a vector of outputs y and calculates the Fisher score as defined in the lecture.

Use this function to evaluate the different dimensions of the input space (there are 70 dimensions) to estimate their individual predictive power.

Please report the ordered list of dimensions with the top 20 Fisher scores, and their Fisher score values. The dimensions should be labeled from 1 to 70 depending on their position in the dataset.

***Fisher­­\_score.m***

*The ordered list of dimensions with the top 20 Fisher scores are:*

*>> ind(1:20)*

*ans =*

*Columns 1 through 12*

*48 25 21 70 65 40 29 19 57 20 24 30*

*Columns 13 through 20*

*12 47 61 10 34 27 39 41*

*Their Fisher score values:*

*>> val(1:20)*

*ans =*

*Columns 1 through 7*

*0.3192 0.2140 0.1910 0.1892 0.1693 0.1673 0.1650*

*Columns 8 through 14*

*0.1402 0.1255 0.1212 0.0995 0.0950 0.0858 0.0846*

*Columns 15 through 20*

*0.0607 0.0579 0.0527 0.0462 0.0461 0.0422*

**Part b.** Write and submit a function AUROC\_score(x, y) that takes as arguments a one dimensional vector of inputs x and a vector of outputs y and calculates the area under the ROC curve. You may use Matlab functions to calculate the area under the curve for this purpose. Similarly to part a, evaluate the different dimensions of the input space and their individual predictive power based on AUROC score. Again, report the ordered list of 20 dimensions with the top 20 AUROC scores, and their values.

Compare the results from part a and part b and discuss your findings. Are the ordered lists the same? In general, do you expect them to be the same.

**AUC\_score.m**

*ans =*

*Columns 1 through 12*

*48 40 21 67 70 65 12 24 39 6 57 20*

*Columns 13 through 20*

*5 14 31 13 41 16 38 37*

*>> val(1:20)*

*ans =*

*Columns 1 through 7*

*0.7133 0.6887 0.6833 0.6730 0.6707 0.6620 0.6459*

*Columns 8 through 14*

*0.6432 0.6412 0.6383 0.6270 0.6208 0.6168 0.6090*

*Columns 15 through 20*

*0.6043 0.5911 0.5833 0.5738 0.5712 0.5693*

*They are not the same.*

*In general, I am not expecting them to reach exactly same results as they rank the feature based on different criteria and formulas*