Assignment 6

Description

The goal of this assingment was to use post-clustering techniques to validate the results we got from the FCM assignment, and to compare the post clustering results to the pre-clustering iVAT results.

Methods

I chost to use the partition coefficient, classification entroy, and the CS index for post clustering. I wrote the methods from scratch, and each can be found in the code below:

```
% Assignment 5
% Taten Knight

% Cleanup
close all;
clear all;
clc;

numClusters = [4, 5, 0, 0, 3, 3, 2, 2, 2, 2, 2, 2]; % From
assignment 4
datasets = LoadDataSets(); % Load all datasets
memberships = cell(length(numClusters), 1);
clusters = memberships;
% Partition Coefficient
pc = zeros(length(numClusters), 10);
pcFinal = zeros(length(numClusters), 1);
% Classification Entropy
ce = pc;
ceFinal = pcFinal;
% CS
cs = ce;
csFinal = ceFinal;
for m=1:length(numClusters)
    if numClusters(m) ~= 0
        memberships{m} = cell(10, 1);
        clusters{m} = memberships{m};
        for C=2:10
            memberships{m}{C} =
readtable(['membership_dataset_' num2str(m) '_clusters_'
num2str(C) '.xlsx']);
            clusters{m}{C} = readtable(['clusters_dataset_'
num2str(m) '_clusters_' num2str(C) '.xlsx']);
        end
    end
end
```

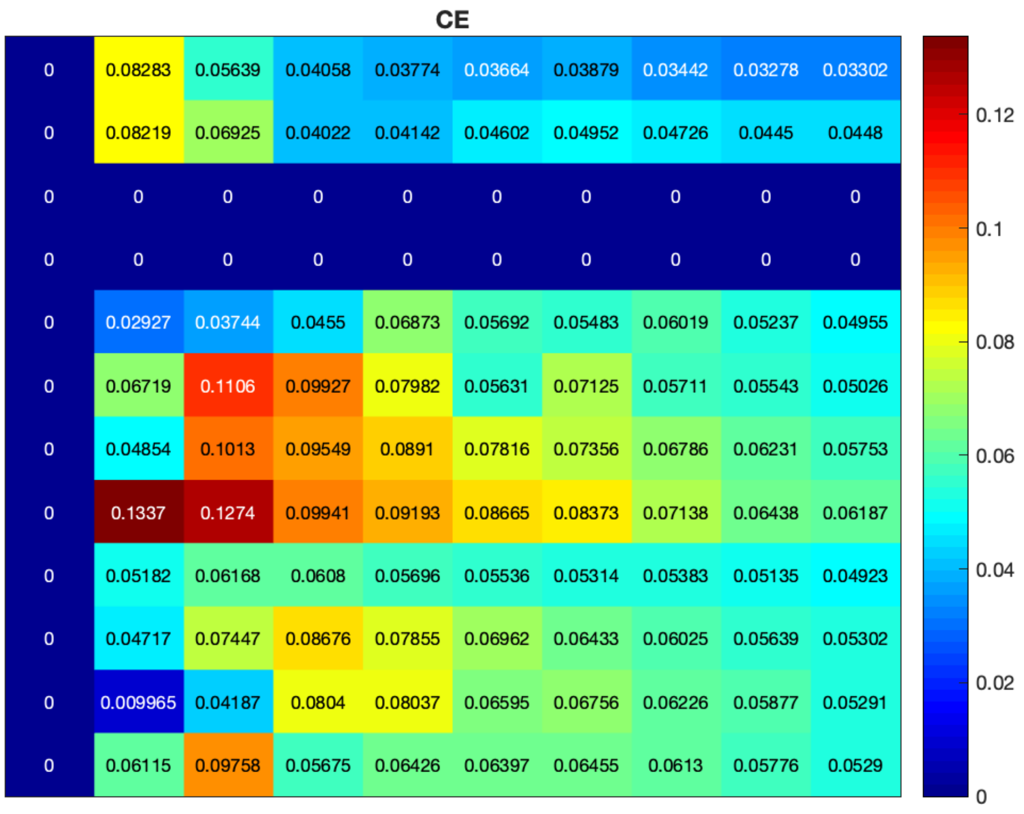
```
for setNum=1:length(numClusters)
    if numClusters(setNum) ~= 0
        % Input Data
        for C = 2:10
            testSet = datasets{setNum};
            testMemb = memberships{setNum}{C}{:, :};
            testClust = clusters{setNum}{C}{:, :};
            dims = size(testSet);
            n = dims(1); % Number of datapoints
            d = dims(2); % Dimensions of data

            % PC
            membershipSquared = testMemb .* testMemb;
            membershipSquaredSummed =
sum(sum(membershipSquared));
            pcVal = membershipSquaredSummed /
numel(membershipSquared);
            pc(setNum, C) = pcVal;

            % CE
            membLogged = testMemb .* log(testMemb);
            membLoggedSummed = sum(sum(membLogged));
            ceVal = membLoggedSummed / numel(membLogged);
            ce(setNum, C) = -1 * ceVal;

            % CS
            numSum = 0;
            denSum = 0;
            clusterDistances = sqrt(pdist2(testClust,
testClust));
            for c = 1:C
                clusterElements = testSet(testMemb(c, :) > .5,
:);
                distances = sqrt(pdist2(clusterElements,
clusterElements));
                scatterSum = 0;
                for j = 1:numel(clusterElements(:, 1))
                    maxScatter = max(distances(j, :));
                    scatterSum = scatterSum + maxScatter;
                end
                scatterSum = scatterSum /
length(clusterElements(:, 1));
                numSum = numSum + scatterSum;
                for cden = 1:C
                    clusterDistance = clusterDistances(c, :);
                    clusterDistance(c) = [];
                    minClusterDistance = min(clusterDistance);
                    denSum = denSum + minClusterDistance;
                end
            end
```

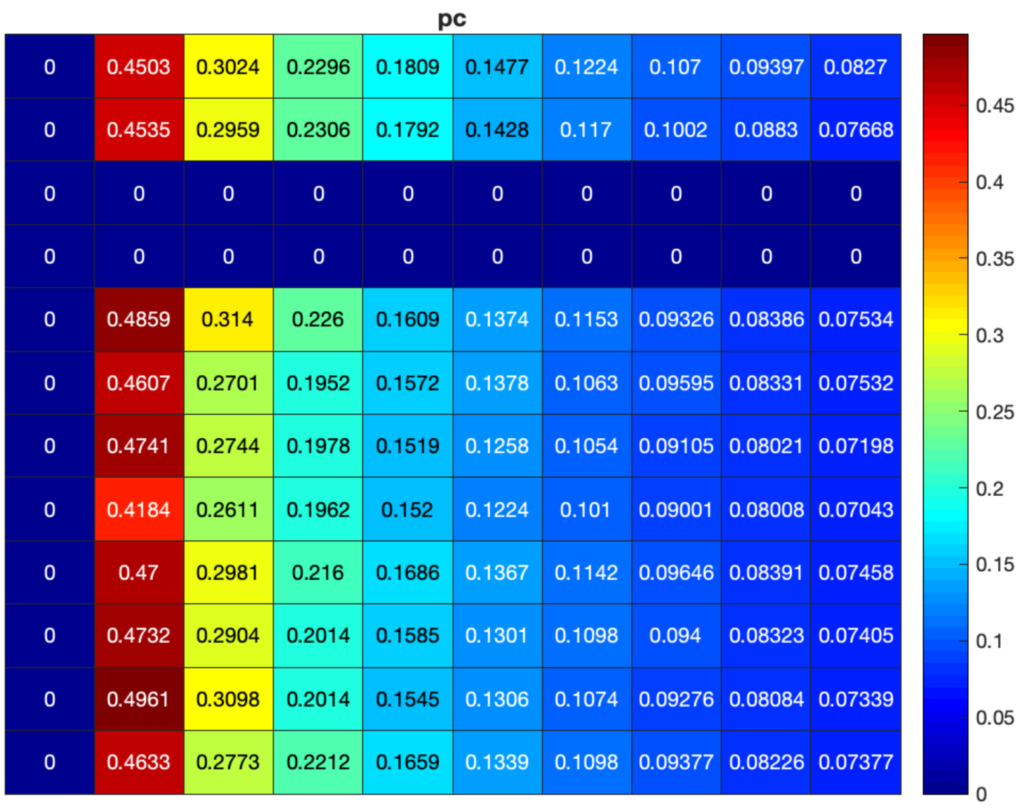
```
                cs(setNum, C) = numSum / denSum;

            end
        end
        tempPc = pc(:, 2:end);
        [holder1, pcI] = min(tempPc(setNum, :));
        pcFinal(setNum) = pcI + 1;

        tempCe = ce(:, 2:end);
        [holder2, ceI] = min(tempCe(setNum, :));
        ceFinal(setNum) = ceI + 1;

        tempCs = cs(:, 2:end);
        [holder3, csI] = min(tempCs(setNum, :));
        csFinal(setNum) = csI + 1;
    end

    fig = figure();
    heat0 = heatmap(cs);
    title('CS');
    heat0.GridVisible = 'off';
    colormap('jet');
    heat0.XDisplayLabels = nan(size(heat0.XDisplayData));
    heat0.YDisplayLabels = nan(size(heat0.YDisplayData));
    saveas(fig, 'cs.png');

    fig = figure();
    heat1 = heatmap(pc);
    title('pc');
    heat0.GridVisible = 'off';
    colormap('jet');
    heat1.XDisplayLabels = nan(size(heat1.XDisplayData));
    heat1.YDisplayLabels = nan(size(heat1.YDisplayData));
    saveas(fig, 'pc.png');

    fig = figure();
    heat2 = heatmap(ce);
    title('CE');
    heat2.GridVisible = 'off';
    colormap('jet');
    heat2.XDisplayLabels = nan(size(heat2.XDisplayData));
    heat2.YDisplayLabels = nan(size(heat2.YDisplayData));
    saveas(fig, 'ce.png');
```

Results

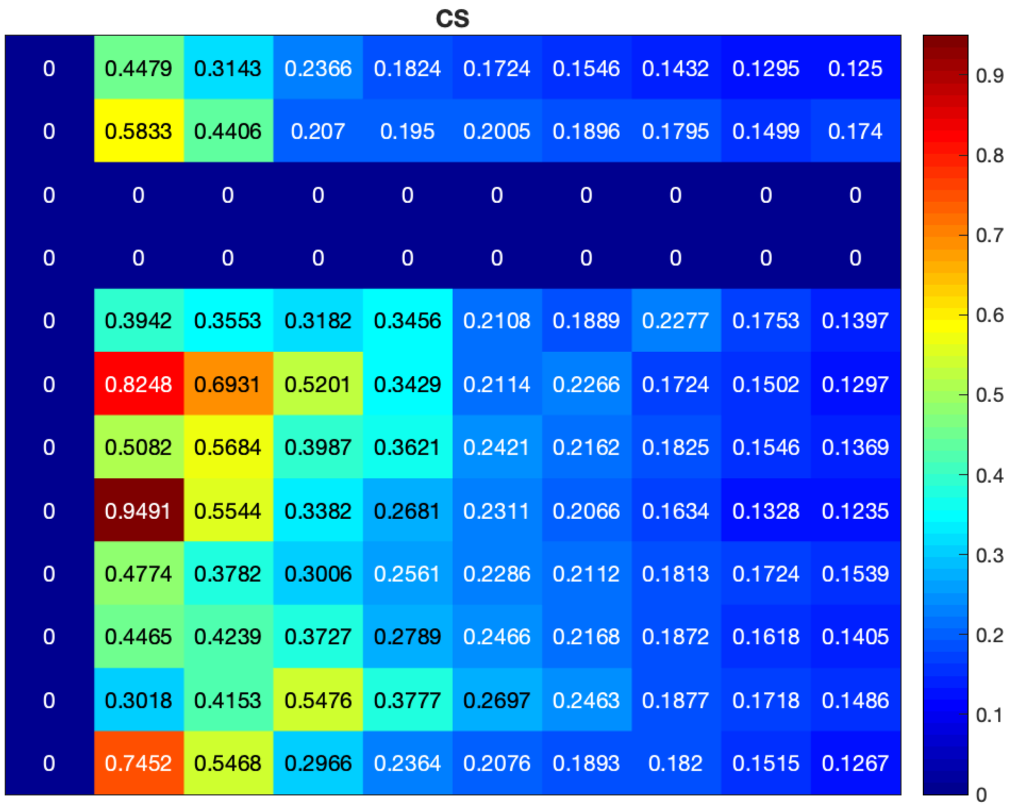
The final results that I yielded were not as expected when taking into consideration the iVAT results, and some intuition from the scatter plots of the point. I carefully checked through my code, and am not sure where the discrepancy is between it and the expected algorithm. I also used this paper as a reference. I am still unsure if there is an issue in my code, or the correct results are unintuitive (I believe it to be the former). The results show that for all three methods I chose (PC, CE, and CS), the methods favor the most clusters. In order to visualize the methods results, I created the following charts:

```
In [13]: from IPython.display import Image, display
         import os
         os.chdir(path='/Users/tatenknight/school/unsupervised_ml/week6')
         display(Image(filename='../week5/ce.png'))
         display(Image(filename='../week5/pc.png'))
         display(Image(filename='../week5/cs.png'))
```





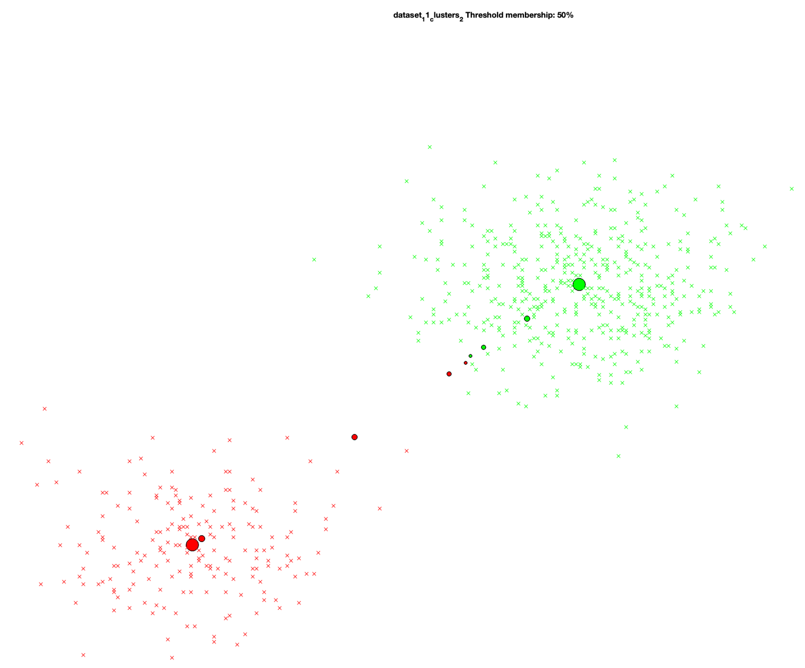
0	0.4503	0.3024	0.2296	0.1809	0.1477	0.1224	0.107	0.09397	0.0827
0	0.4535	0.2959	0.2306	0.1792	0.1428	0.117	0.1002	0.0883	0.07668
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0.4859	0.314	0.226	0.1609	0.1374	0.1153	0.09326	0.08386	0.07534
0	0.4607	0.2701	0.1952	0.1572	0.1378	0.1063	0.09595	0.08331	0.07532
0	0.4741	0.2744	0.1978	0.1519	0.1258	0.1054	0.09105	0.08021	0.07198
0	0.4184	0.2611	0.1962	0.152	0.1224	0.101	0.09001	0.08008	0.07043
0	0.47	0.2981	0.216	0.1686	0.1367	0.1142	0.09646	0.08391	0.07458
0	0.4732	0.2904	0.2014	0.1585	0.1301	0.1098	0.094	0.08323	0.07405
0	0.4961	0.3098	0.2014	0.1545	0.1306	0.1074	0.09276	0.08084	0.07339
0	0.4633	0.2773	0.2212	0.1659	0.1339	0.1098	0.09377	0.08226	0.07377

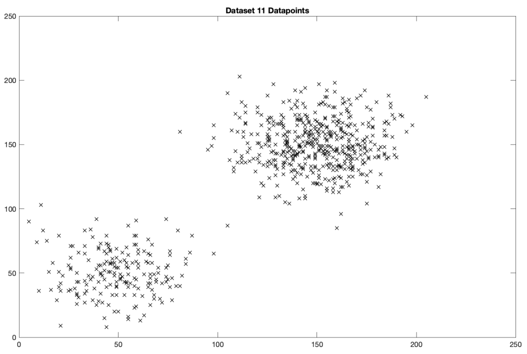


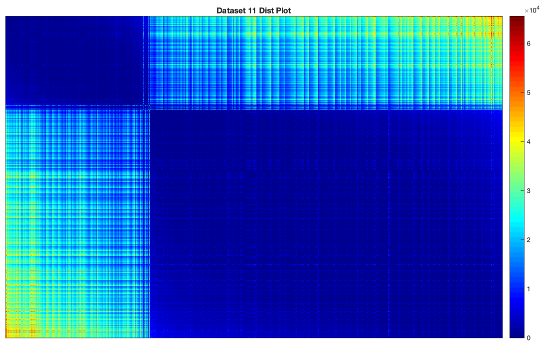
0	0.4479	0.3143	0.2366	0.1824	0.1724	0.1546	0.1432	0.1295	0.125
0	0.5833	0.4406	0.207	0.195	0.2005	0.1896	0.1795	0.1499	0.174
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0.3942	0.3553	0.3182	0.3456	0.2108	0.1889	0.2277	0.1753	0.1397
0	0.8248	0.6931	0.5201	0.3429	0.2114	0.2266	0.1724	0.1502	0.1297
0	0.5082	0.5684	0.3987	0.3621	0.2421	0.2162	0.1825	0.1546	0.1369
0	0.9491	0.5544	0.3382	0.2681	0.2311	0.2066	0.1634	0.1328	0.1235
0	0.4774	0.3782	0.3006	0.2561	0.2286	0.2112	0.1813	0.1724	0.1539
0	0.4465	0.4239	0.3727	0.2789	0.2466	0.2168	0.1872	0.1618	0.1405
0	0.3018	0.4153	0.5476	0.3777	0.2697	0.2463	0.1877	0.1718	0.1486
0	0.7452	0.5468	0.2966	0.2364	0.2076	0.1893	0.182	0.1515	0.1267

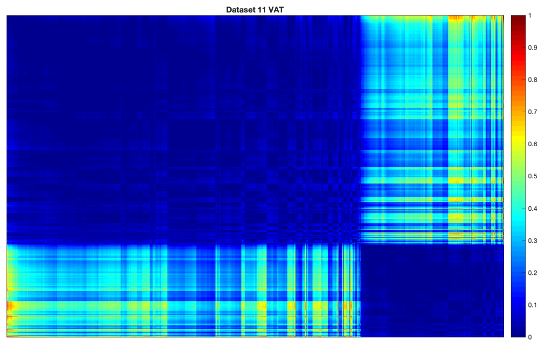
For each of the heatmaps above, the lower values (or darkest blue values) are those that the algorithms indicate are the best for the datasets. I chose to ignore the single cluster case, and datasets 3 and 4 did not yield viable results for iVAT, and so have not been analyzed since then. The CS results indicate that a higher number of clusters is preferable to a lower number, and indicate that 9 or 10 clusters is the optimal choice for each case. *Please note that there is likely an error somewhere in my CS code*. The same tendency is evident in the PC chart. The CE method does not have perfect results, but does indicate that for dataset 11, two clusters is the optimum number. Comparing this to the iVAT results and the scatter plot, we see that this is a very reasonable answer:

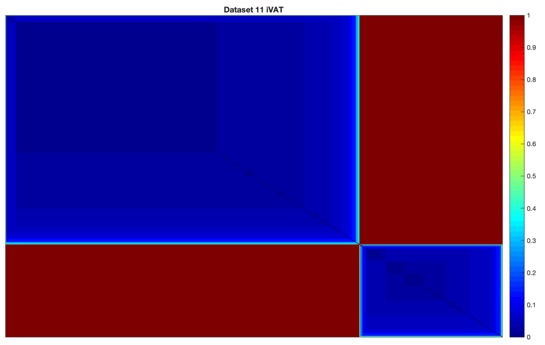
In [14]:
```
display(Image(filename='../week5/dataset_11_clusters_2_threshold_50.png'))
display(Image(filename='../week4/dataset_11_results.png'))
```











Conclusion

Looking at the results above, we can see that the methods are capable of drawing accurate results given the data. I do however believe that there was an error in my code. I prefer to write the code from scratch to have a better understanding of the algorithms we use, and I'll run back through it at a later date and look for error (any feedback here would be appreciated). Although not perfect, this exercise does indicate the validity of using post-clustering techniques to verify results. Ironically though, I had to re-run the assignment 5 code for all 9 cluster numbers just to get the values needed for the post-clustering algorithms. If you are looking at a simple dataset or an easily separable dataset after LDA, iVAT alone is likely a solid choice for picking the correct number of clusters. It already resembles the numerator of the CS index to some degree, and LDA has already taken into account separability via the between cluster and within cluster scatters.

With this in mind, I believe that the best number of clusters for datasets 1-12 are: [4, 5, 0, 0, 3, 3, 2, 2, 2, 2, 2, 2] based on the results from Assignment 4.