CPE 466

Lab 5

Gilbert, Andrew Terrell, Josh

apgilber@calpoly.edu jmterrel@calpoly.edu

Study Design

We implemented both the K-Means and Agglomerative Hierarchical clustering algorithms. The datasets we studied contained floating point values for all features. We used all the raw feature vectors without normalization. Through this lab, we gained understanding of some performance differences between the algorithms as well as experience tweaking parameters for the algorithms.

K-Means and Agglomerative Hierarchical were implemented using euclidean distance measures between cluster centroids. For K-Means, we stopped iterating when the difference in sum of squared error between two iterations became less than 0.01. For Agglomerative Hierarchical, we built the entire tree, then cut the tree afterward. Since Hierarchical takes a non-trivial amount of time to construct for some larger datasets, we implemented an automatic disk-cache of the tree to allow users to experiment with trying different cutting thresholds quickly.

To help us understand what the best clusterings were for 2D datasets, we decided to implement a brief visualization program. The rendered graphs are contained in the results section. See the README for instructions on executing the programs.

Results

This section contains information about the best settings for running each algorithm on each dataset as well as small discussions on some of the datasets. More general observations are noted in the next section.

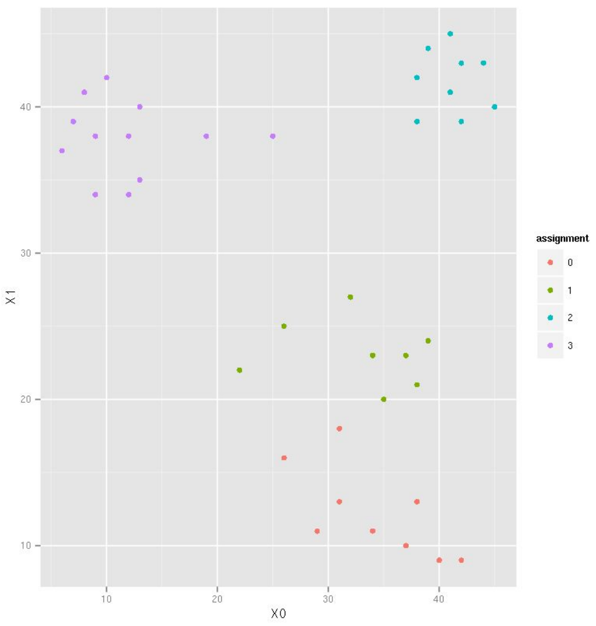
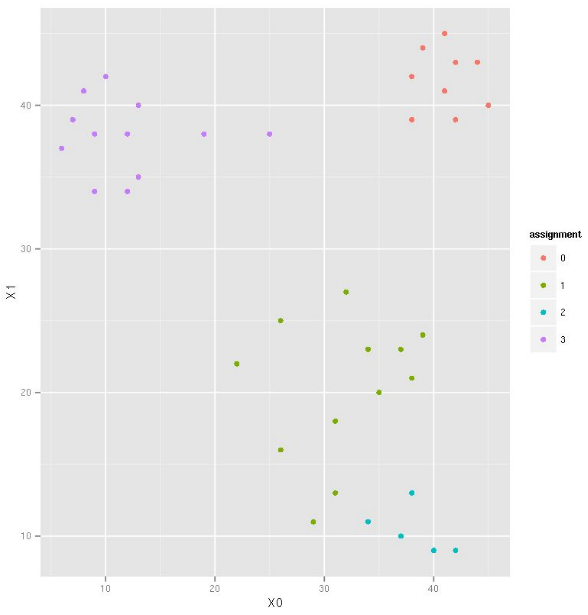
Note: Our K-Means was non-deterministic due to random initialization of centroids. We picked the best clusters we observed.

2 Dimensional Datasets

Many of our datasets were two dimensional. We graphed scatterplots of the results to help us visualize the best clusterings.

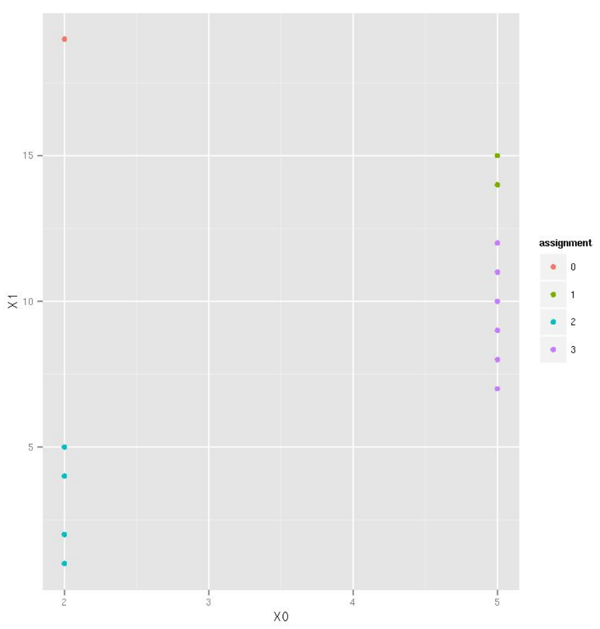
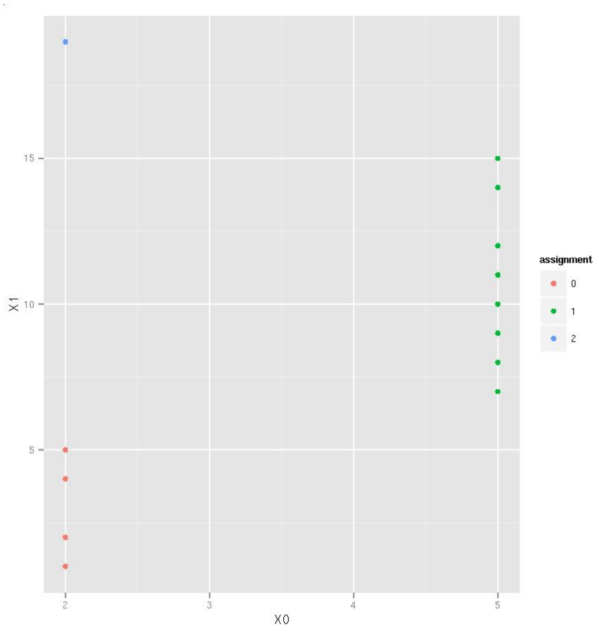
4 Clusters

Neither algorithm got to what we thought would have been the perfect clustering. However we believe hierarchical clustering did better in comparison due to the distance of the two misclassified points in both graphs.

<table>
<tr><th>hierarchical data/4clusters.csv 12</th><th>kmeans data/4clusters.csv 4</th></tr>
<tr><td>
</td><td>
</td></tr>
</table>

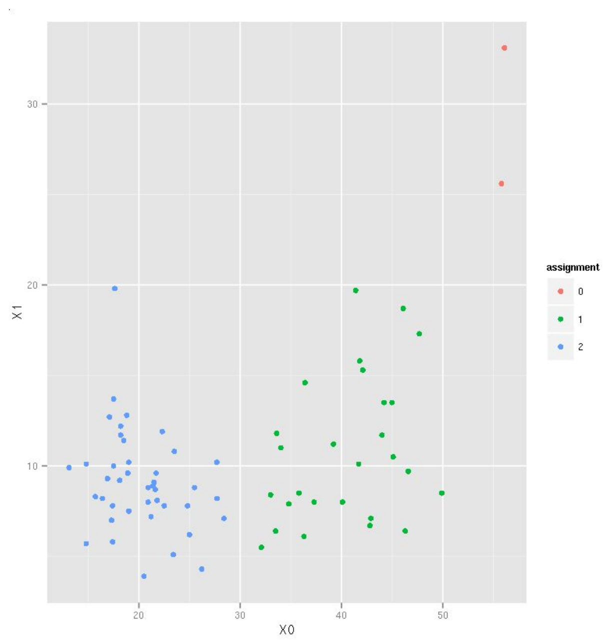
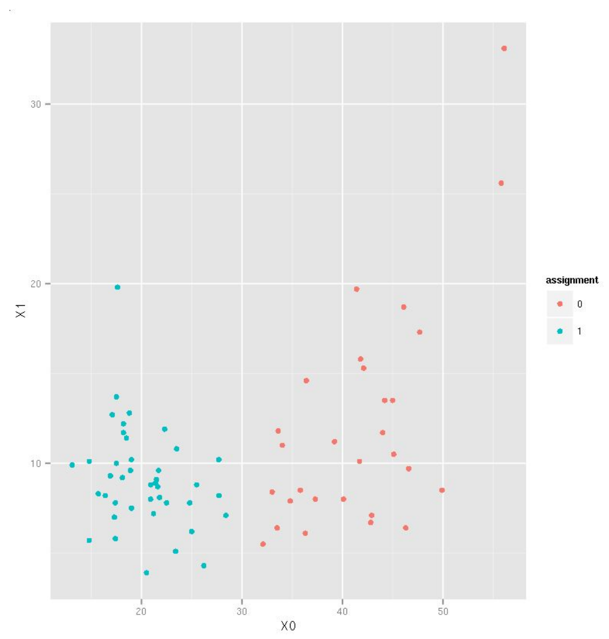
Accidents 1

K-Means is the clear winner on this dataset according to this visualization. Note that the axes are scaled differently.

hierarchical data/AccidentsSet01.csv 5	kmeans data/AccidentsSet01.csv 3
	

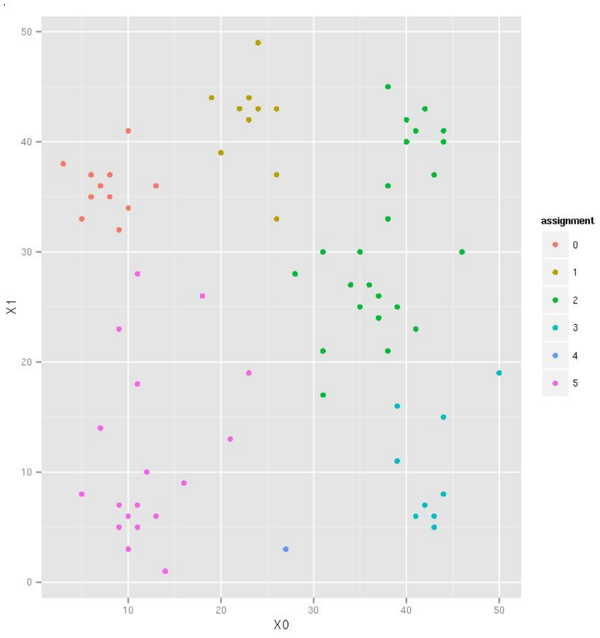
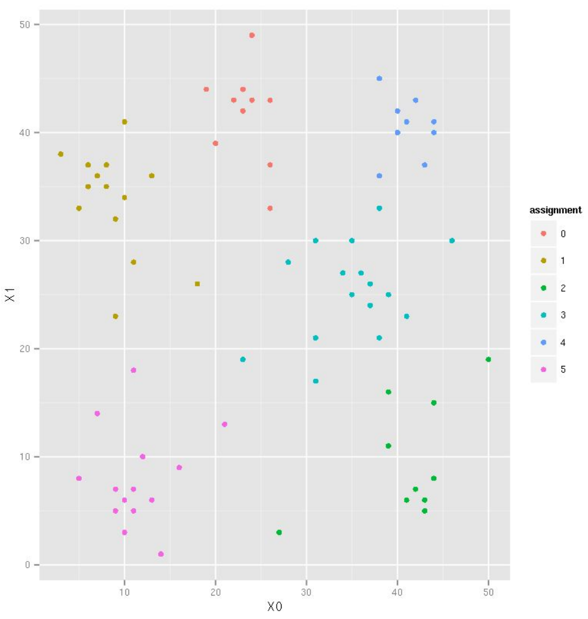
Birth Death Rate

We were not sure which of the two clusterings were better for this dataset. However since hierarchical puts the outliers in their own class, we lean toward hierarchical doing better than k-means.

`hierarchical data/birth_death_rate.csv 20`	`kmeans data/birth_death_rate.csv 2`
	

Many Clusters

When we originally visualized this dataset without clusters, we saw 6 clusters. We were able to get the clusters we saw using k-means, but hierarchical was unable to split the top-right cluster successfully due to the outlier point on the center bottom receiving its own cluster.

`hierarchical data/many_clusters.csv 8`	`kmeans data/many_clusters.csv 6`
	

N>2 Dimensional Datasets

We were able to visualize the datasets in two dimensions by neglecting other dimensions, but we didn't find these representations of the data terribly useful.

Mammal Milk

hierarchical data/mammal_milk.csv 15		kmeans data/mammal_milk.csv 3	
size	avg dist to center	size	avg dist to center
2	3.69	2	3.68
4	5.01	16	4.51
19	6.56	7	6.35

Planets

3 clusters seemed about right for this dataset. Breaking it up into 4 removed a single not so influential outlier from the size 8 dataset. 2 clusters resulted in the dist to center skyrocketing. Note this metric of average dist to center isn't too helpful since the features are not normalized.

hierarchical data/planets.csv 60		kmeans data/planets.csv 3	
size	avg dist to center	size	avg dist to center
2	0.74	2	0.74
8	15.43	8	14.3
9	33.19	9	19.8

Economy

hierarchical data/economy.csv 10		kmeans data/economy.csv 3	
size	avg dist to center	size	avg dist to center
4	6.07	12	3.31
20	6.65	8	4.44
		4	5.50

Accidents 2

hierarchical data/AccidentsSet02.csv 15		kmeans data/AccidentsSet02.csv 2	
size	avg dist to center	size	avg dist to center
1	0.0	15	4.45
14	4.95	34	5.55
34	6.08		

Accidents 3

hierarchical data/AccidentsSet03.csv 2.5		kmeans data/AccidentsSet03.csv 1	
size	avg dist to center	size	avg dist to center
1	0.0	62	1.81
1	0.0		
15	1.41		
45	1.45		

Discussion

There are many different ways to measure the distance between clusters. For accidents#1, using distance between the closest points of the clusters would have enabled us to get the correct clustering using hierarchical. Throwing out outliers from the computations of centroids in K-Means may have provided better results as well.

We realized normalizing the features before clustering would have been especially helpful using euclidean distance. A lot of our features were unscaled and thus our ability to cluster suffered.

Finally, we realized how difficult it is to determine the correct clustering for more than two dimensions. Having feature names would help us understand the ranges of features for each cluster to determine whether clusters made sense.

Analysis

The difference between the accuracy of the clustering algorithms is difficult to assess. For our two dimensional datasets, k-means was clearly the winner on some, but tweaking hierarchical may have provided superior results.

We realized that the specific algorithm didn't matter too much. Both did alright on any dataset. What mattered more was the choice of parameters for the algorithm. Finding the parameters involves understanding what good clusterings look like and what bad clusterings look like. We found clusterings of just one point--these could have been eliminated with outlier detection.

The most important facet of the algorithm was the ability to iterate quickly and receive quick feedback. Once the clustering was made, one could visualize it using different tools and determine which direction the parameters needed to be changed, whether outliers needed to be removed, etc.