IEOR E4650 Business Analytics

Session 12: Clustering

Spring 2018

Prof. Adam Elmachtoub

Main Learning Tasks: Supervised Learning

- Supervised learning: We're predicting an output variable for which we get to see examples.

  Data: n observations including response Y and p features $X_1, X_2, \ldots, X_p$.

  Goal: Predict Y using $X_1, X_2, \ldots, X_p$.

  - Regression
  - Classification

Main Learning Tasks: Unsupervised Learning

- Unsupervised learning: There are no dependent variables.

  Data: n observations only including p features $X_1, X_2, \ldots, X_p$.

  Goal: not interested in prediction; want to discover interesting patterns and subgroups; can be used to pre-process and reduce dimension of data.

  - Clustering (today): Used to aggregate observations into meaningful subgroups (aggregate rows)
  - Principle Component Analysis (next time): Used to reduce dimensionality of feature space and for making visualizations, (aggregate columns)

Clustering Methods

- Clustering seeks to find homogeneous subgroups among the observations, so members in each subgroup are close to each other.
- Find homogeneity and heterogeneity among the data.
- K-mean clustering: partition observations into a pre-specified number of clusters
- Hierarchical clustering: no pre-specified number of clusters; end up with a tree-like visual representation

Applications of Clustering

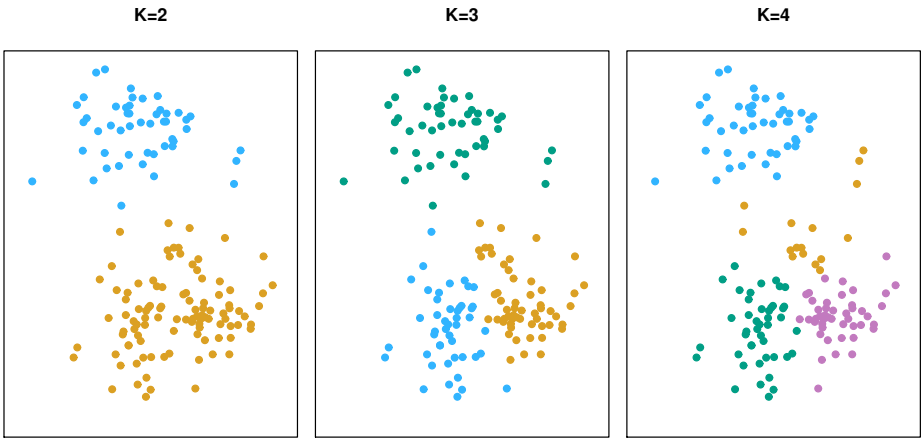
- Market segmentation via clustering to identify similar consumers/users
    - Can use information about a consumer's cluster to try to understand their preferences, and what type of products or advertisements to display
- Product segmentation clusters store/site items into categories
    - Allows us to assign new products to categories, and manage each category separately (assign a manager, give coupon, improve quality)
- Cancer researchers cluster genes into subgroups to obtain better understanding of diseases
    - The similarities within clusters might lead to new hypothesis or discoveries

Overview of K-Means Clustering

- K-means clustering is a simple and elegant approach for partitioning a data set into K distinct, non-overlapping clustering
- To perform K-means clustering, we must first specify the desired number of clusters K
- Then, the K-mean algorithm assigns each observation to exactly one of the K clusters, relying on a distance metric between two observations (e.g., Euclidean distance or correlation distance)

Clustering Illustration

Simulated data in two-dimensional space: ran the K-means clustering algorithm for K=2,3,4



Best K Clusters

Let $C_1, \ldots, C_K$ be the K clusters of observations we wish to create. The sets satisfy two properties:

- $C_1 \bigcup C_2 \bigcup \ldots \bigcup C_K = \{1, 2, \ldots, n\}$. In other words, each observation belongs to at least one of the K clusters.
- $C_k \bigcap C_{k'} = \emptyset$ for all $k \neq k'$. In other words,each observation belongs to at most one of the K clusters.

The idea behind K-means clustering is that a good clustering is one for which the within-cluster variation is as **small** as possible.

Within-Cluster Variation

- The within-cluster variation for cluster C_k, denotedy by $W(C_k)$, is the amount by which the observations within a cluster differ from each other.
- Using squared Euclidean distance, we define

$$W(C_k) = \frac{1}{|C_k|} \sum_{i,i' \in C_k} \sum_{j=1}^{p} (x_{ij} - x_{i'j})^2,$$

  where $|C_k|$ denotes the number of observations in the k-th cluster.
- The best K-means clustering is the solution to

$$\min_{C_1, C_2, \ldots, C_K} \left\{ \sum_{k=1}^{K} \frac{1}{|C_k|} \sum_{i,i' \in C_k} \sum_{j=1}^{p} (x_{ij} - x_{i'j})^2 \right\}$$

- This optimization problem is highly non-linear and too difficult to solve.

Some Intuition

- Let $\bar{x}_{kj}$ be the mean for feature j in cluster C_k. Then

$$W(C_k) = \frac{1}{|C_k|} \sum_{i,i' \in C_k} \sum_{j=1}^{p} (x_{ij} - x_{i'j})^2 = 2 \sum_{i \in C_k} \sum_{j=1}^{p} (x_{ij} - \bar{x}_{kj})^2$$

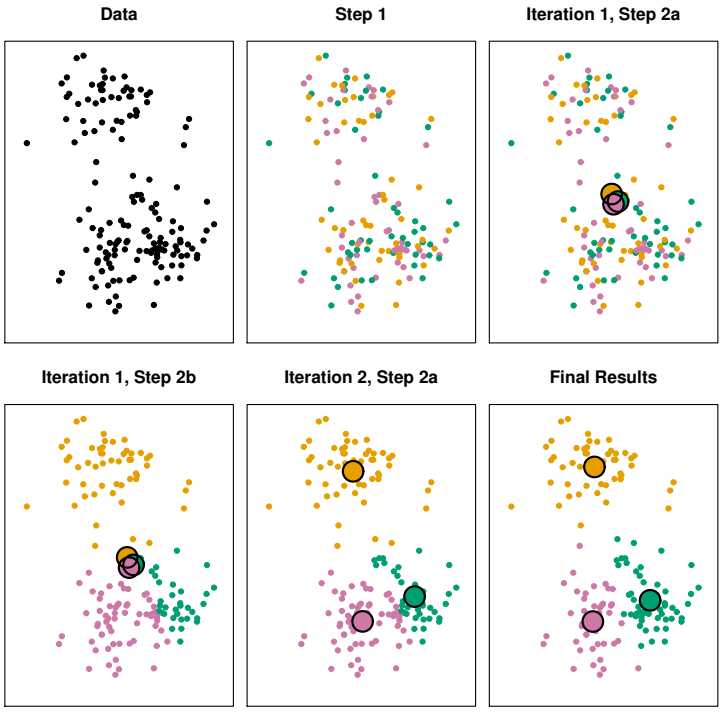
- Thus the within-cluster variation is equivalent to the total squared distance from the centroid of the cluster!

K-means Algorithm

- Instead, we use the following method which obtains a local optimal solution
- The K-means method is as follows:
  (1) Randomly assign a number, from 1 to K, to each of the observations. These serve as initial cluster assignments for the observations.
  (2) Repeat until the cluster assignments stop changing:
    (a) For each of the K clusters, compute the cluster centroid (mean). The kth cluster centroid is the vector of the p feature means for the observations in the k^{th} cluster.
    (b) Assign each observation to the cluster whose centroid is closest (where closest is defined using Euclidean distance).
- You can run the algorithm multiple times from different random initial configurations to obtain multiple local optima, then pick the best clustering.

Illustration of K-means Algorithm

Successive iterations of the K-means clustering algorithm.



Examples with R: Simulated Data

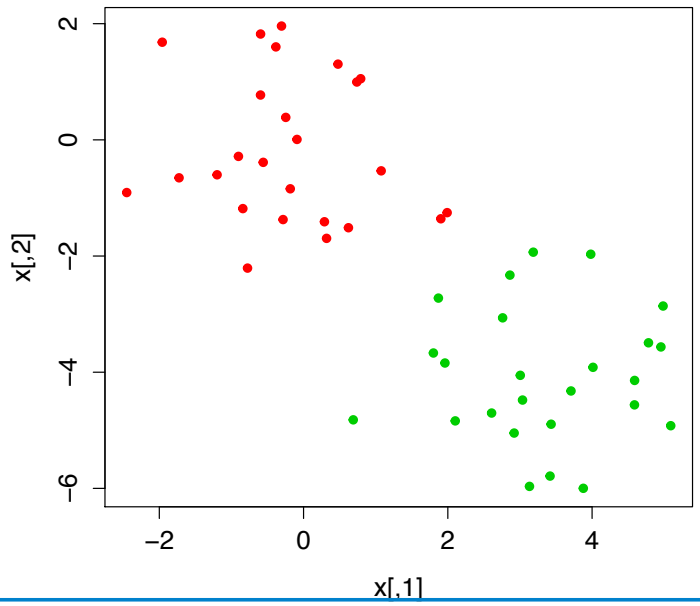
- Function `kmeans()` performs K-means clustering in R.
- Example with simulated data

```
> set.seed(2)
> x=matrix(rnorm(50*2),ncol=2)
> plot(x)
> x[1:25,1]=x[1:25,1]+3
> x[1:25,2]=x[1:25,2]-4
> plot(x)
> km.out=kmeans(x,2,nstart=20)
> names(km.out)
> km.out$cluster

> plot(x,col=km.out$cluster+1,pch=20,lwd=3)
> km.out$tot.withinss
```


K-Mean Clustering

Simulated data in two-dimensional space: clustered into K=2 classes by the K-means clustering algorithm.



Examples with R, Cont.

- Change number of clusters with centers
- nstart is the number of times we run K-means, each time with a different initialization of the clusters

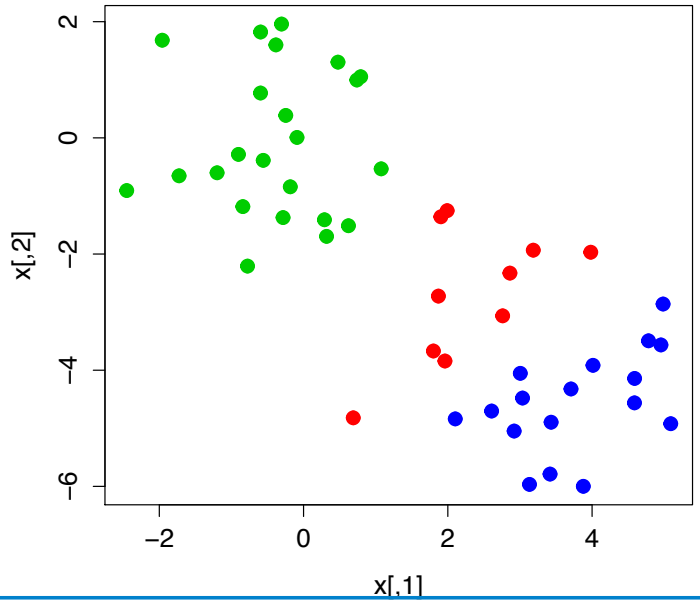
```
> set.seed(4)
> km.out2=kmeans(x,centers=3,nstart=20)
> km.out2

> plot(x,col=km.out2$cluster+1,pch=10,lwd=5)
> km.out2$tot.withinss
```

- Recommend running K-means clustering with a large value of `nstart`, such as 20 or 50

K-Mean Clustering

Simulated data in two-dimensional space: clustered into K=3 classes by the K-means clustering algorithm.



Crime Data for 2014

- Arrests data on fifty states, per 100,000 people

```
> USAdata_0 =read.csv("CrimeOneYearofData2014.csv")
> USAdata=data.frame(USAdata_0[,-1],row.names=USAdata_0[,1])
> head(USAdata)
           Population  Murder  Assault  Burglary
Alabama       4849377     5.7    283.4     819.0
Alaska         736732     5.6    440.2     427.6
Arizona       6731484     4.7    252.1     647.1
Arkansas      2966369     5.6    346.0     835.7
California   38802500     4.4    236.6     522.3
Colorado      5355866     2.8    192.8     438.2
```

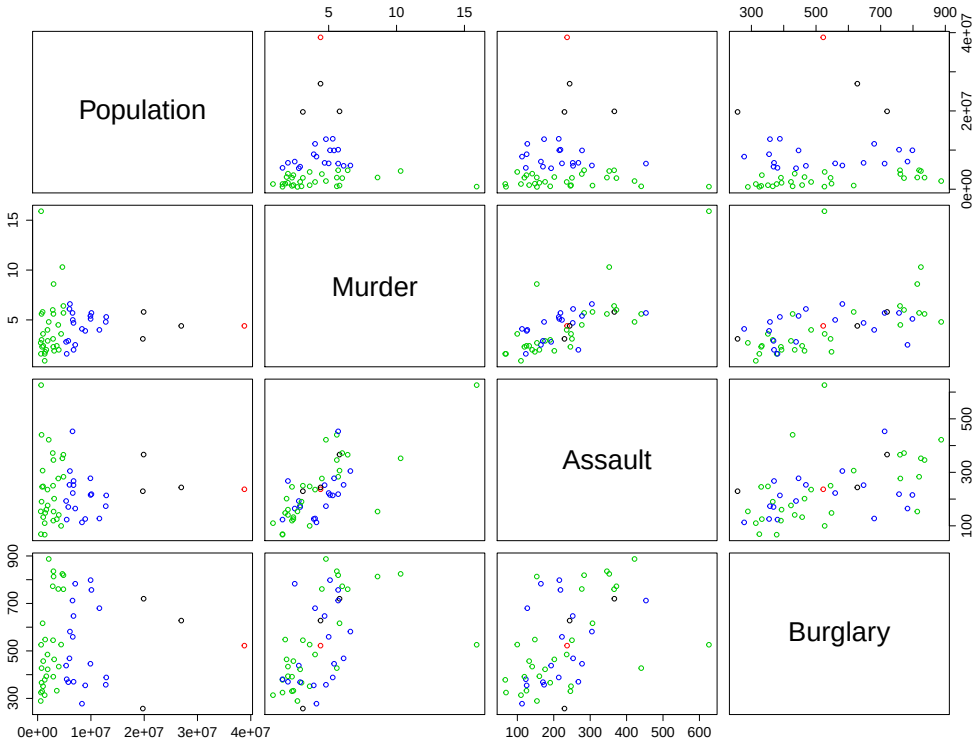
- Could rescale data using my.dat = scale(my.dat).
- scale() will make columns have mean 0 and standard deviation of 1

Data: USArrests

- Output of kmeans in R.

```
> km.fit=kmeans(USAdata, centers=4,nstart=20)
> names(km.fit)
[1] "cluster"      "centers"      "totss"
[4] "withinss"     "tot.withinss" "betweenss"
[7] "size"         "iter"         "ifault"
> head(km.fit$cluster)
Alabama     Alaska    Arizona   Arkansas California
2           3         2          3          4
Colorado
2
> km.fit$withinss
[1] 6.673602e+13 6.165444e+13 2.993154e+13 7.015843e+13
> km.fit$tot.withinss
[1] 2.284804e+14
> plot(USAdata,col=km.fit2$cluster)
```

Visualizing the Clusters

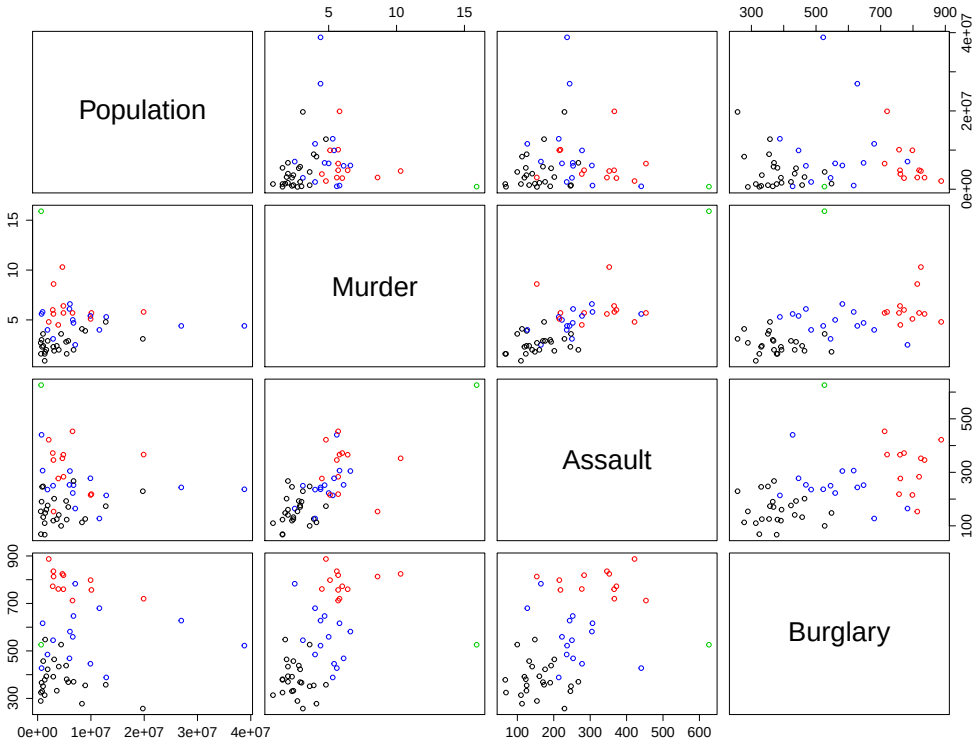


Better idea: Scale Data, Remove population

- Output of kmeans in R.

```
> USAdata2=data.frame(scale(USAdata[,-1]))
> head(USAdata2)
Murder     Assault    Burglary
Alabama     0.55667580  0.47832031  1.60552062
Alaska      0.51696885  1.89973681 -0.54863427
Arizona     0.15960635  0.19458091  0.65943165
Arkansas    0.51696885  1.04579909  1.69743270
California  0.04048551  0.05407099 -0.02743225
Colorado   -0.59482561 -0.34298285 -0.49029486
> km.fit2=kmeans(USAdata2, centers=4,nstart=20)
> plot(USAdata,col=km.fit2$cluster)
```

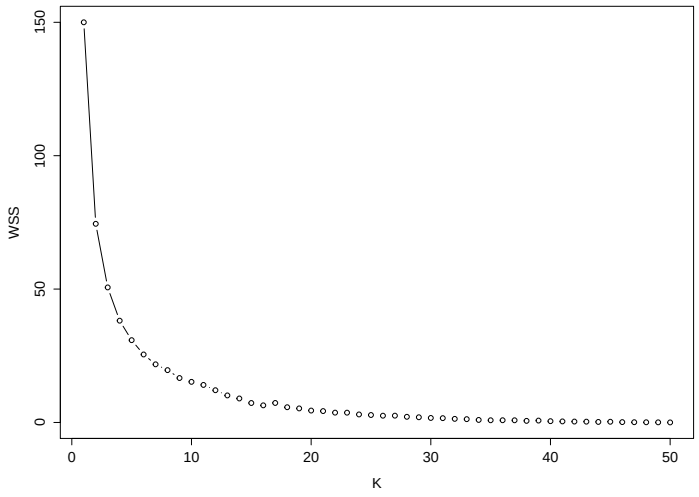
Visualizing the Clusters



How Many Clusters?

- Idea: Get small tot.withinss without using too many clusters

```
> wss=rep(0,nrow(USAdata2)-1)
> for (i in 1:length(wss)) wss[i]<-kmeans(USAdata2,centers=i,nstart=10)$tot.withinss
> plot(1:length(wss),wss,type="b",xlab="K",ylab= "WSS")
```



- It seems that $4 \leq K \leq 7$ should work well.

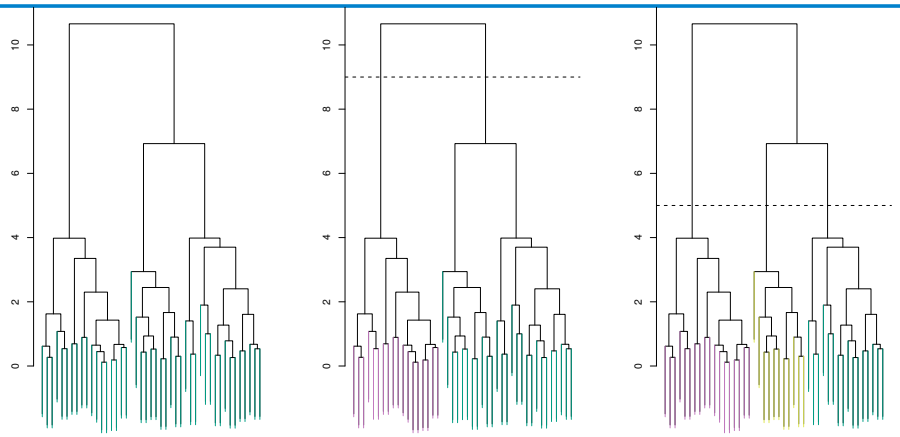
Hierarchical Clustering

- Disadvantage of K-mean clustering: it requires to pre-specify the number of clusters K
- Hierarchical Clustering is an alternative: Does not require K
- Advantage of Hierarchical Clustering: it results in an attractive tree-based representation of the observations, called a dendrogram

Hierarchical Clustering

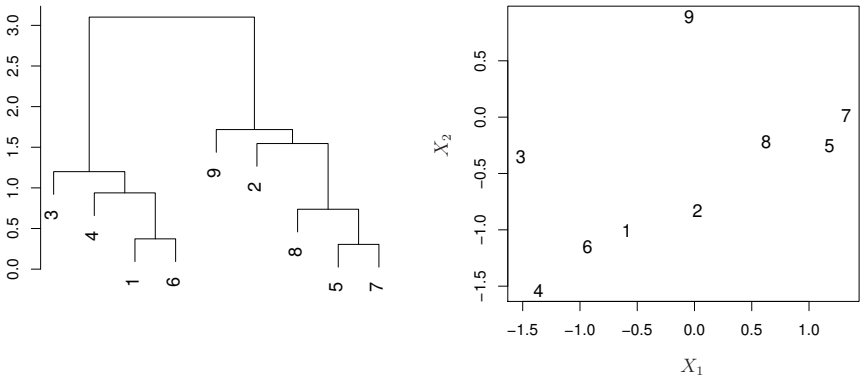
- An agglomerative approach
  - Find closest two things
  - Put them together
  - Find next closest
- Requires
  - A defined distance
  - A merging approach
- Produces
  - A tree showing how close things are to each other

Interpreting Dendrograms



- The leaves at the bottom of the dendrogram represent the individual observations
- Leaves are combined to form branches
- Small branches are combined into larger branches, until one reaches the trunk or the root
- Draw any horizontal (dashed) line - this corresponds to a clustering
- Thus we can visualize all possible clusterings in one dendrogram

Interpreting Dendrograms



- Distance between two observations corresponds to where their branches are fused
- Observations 1 and 6 are .4 apart, 5 and 7 are also about .4 apart
- Observation 3 and 4 are 1.2 apart, NOT .5

Hierarchical Clustering Algorithm

- Algorithm
  (1) Begin with n observations and a measure (such as Euclidean distance) of all the pairwise dissimilarities. Treat each observation as its own cluster.
  (2) For $i = n, n-1, \ldots, 2$
    (a) Examine all pairwise inter-cluster dissimilarities among the i clusters and identify the pair of clusters that are least dissimilar (that is, most similar). Fuse these two clusters.
    (b) Compute the new pairwise inter-cluster dissimilarities among the $i-1$ remaining clusters.
- Three common types of dissimilarity (linkages): complete, average, and single.

Linkages

Linkage is known as the dissimilarity between two clusters. There are three types of linkages. For each, begin by computing Euclidean distance between all pairs of points in both clusters.

- Complete: Maximum distance is dissimilarity between the two clusters.
- Single: Smallest distance is dissimilarity between the two clusters.
- Average: Average of distances is dissimilarity between the two clusters.
- Centroid: Distance between centroids is dissimilarity between the two clusters.

Hierarchical Clustering: R

- Function `hclust()` implements Hierarchical Clustering in R
- Example: use Euclidean distance matrix to measure dissimilarity: `dist()`

```
>  set.seed(2)
>  x=matrix(rnorm(50*2),ncol=2)
>  x[1:25,1]=x[1:25,1]+3
>  x[1:25,2]=x[1:25,2]-4
> hc.complete=hclust(dist(x),method="complete")
> hc.average=hclust(dist(x),method="average")
> hc.single=hclust(dist(x),method="single")

> par(mfrow=c(1,4))
> plot(hc.complete)
> plot(hc.average)
> plot(hc.single)
```

- Use function `cutree()` to determine the cluster labels

```
> cutree(hc.complete,3)
```


The value of scaling

- In addition to carefully selecting the dissimilarity measure used, one must consider whether or not the variables should be scaled to have standard deviation one.
- Example: Say feature k corresponds to how many times a customer bought product k. A regular clustering would group low volume shoppers together and high volume shoppers together. However, what if $x_i = 10x_j$, thus customers i and j have same pattern but different volume!
- After scaling, each variable will be in effect given equal importance.
- Mean of every feature is 0 and standard deviation is 1
- R code: function `scale()`. Try with and without!

Hierarchical Clustering: another example

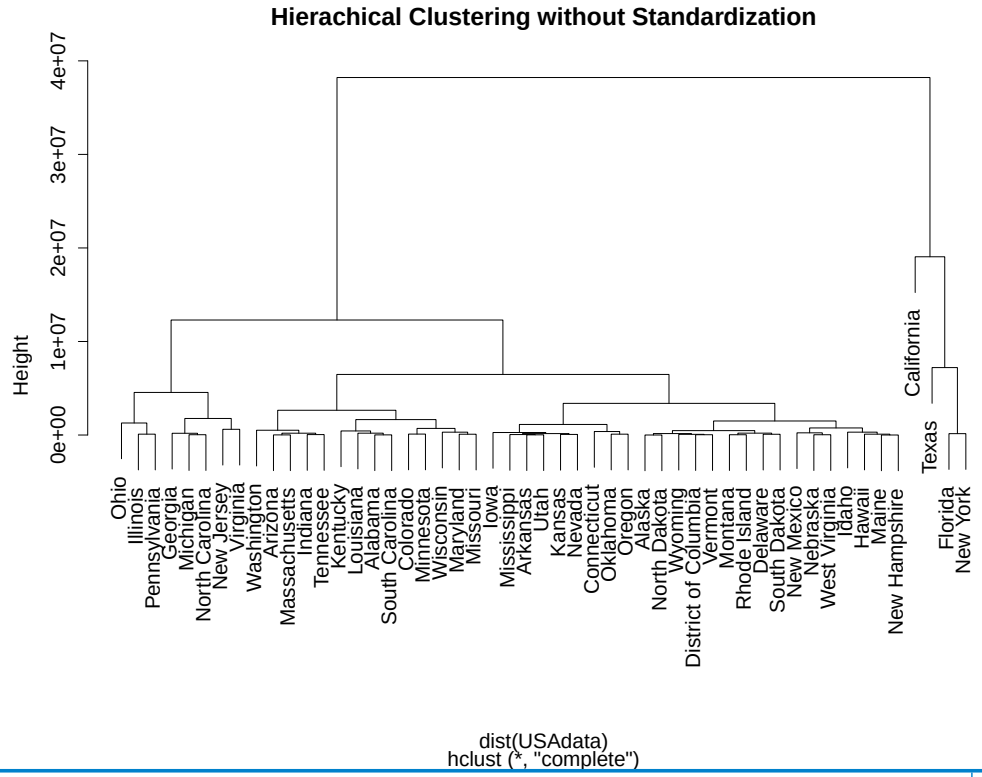
- We will now perform hierarchical clustering on the crime data.
- We will use complete linkage and Euclidean distance to cluster the states.
- Cut the dendrogram at a height that results in four distinct clusters. Which states belong to which clusters?

Hierarchical Clustering: R

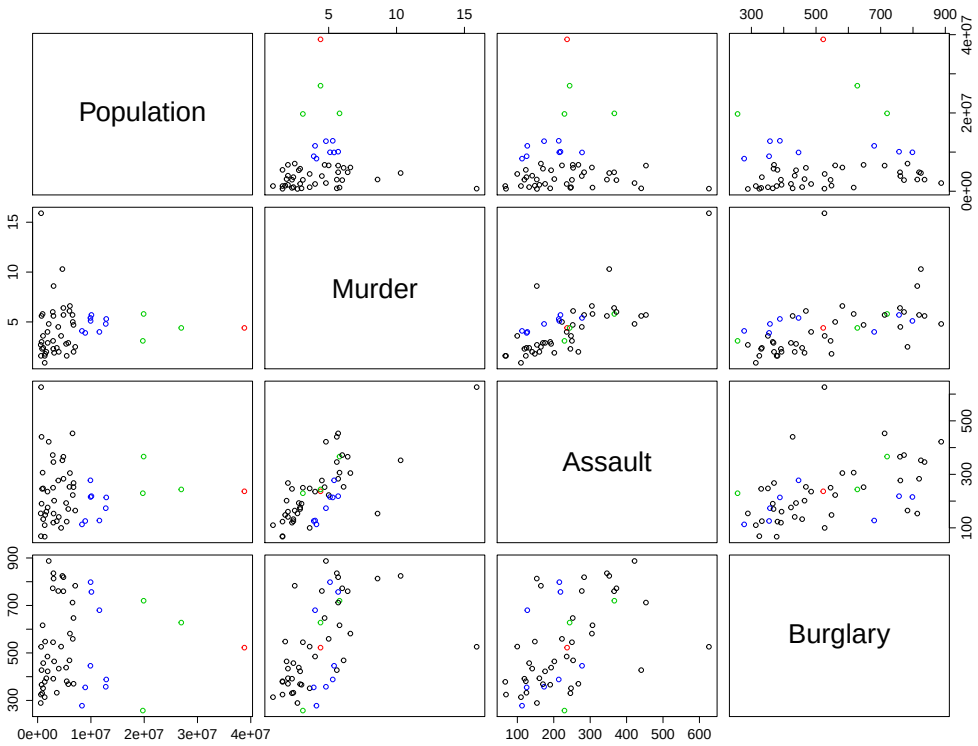
- Sample code:

```
USAdata_0 =read.csv("CrimeOneYearofData2014.csv")
USAdata=data.frame(USAdata_0[,-1],row.names=USAdata_0[,1])
hc.complete=hclust(dist(USAdata),method="complete")
plot(hc.complete,main="Hierachical Clustering without Standardization")
plot(USAdata,col=cutree(hc.complete,4))
```

Hierarchical Clustering: Dendrogram



Visualizing Clusters

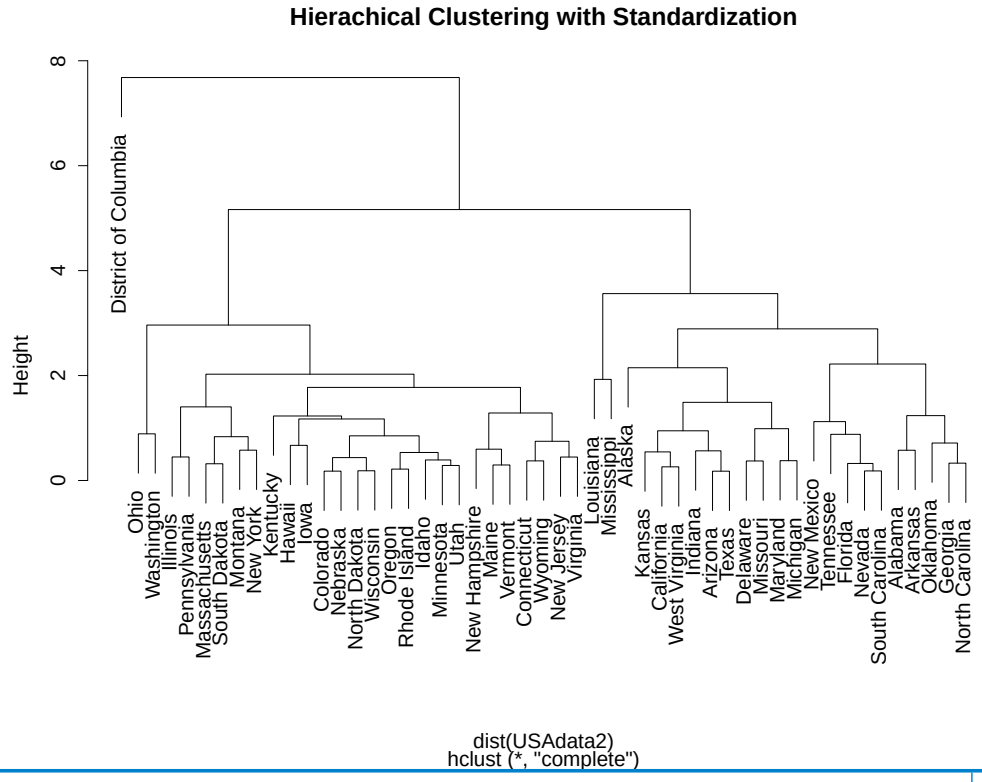


Hierarchical Clustering: R

- Sample code:

```
USAdata2=data.frame(scale(USAdata[,-1]))
hc.complete.2=hclust(dist(USAdata2),method="complete")
plot(hc.complete.2,main="Hierachical Clustering with Standardization")
plot(USAdata,col=cutree(hc.complete.2,4))
```


Hierarchical Clustering: Dendrogram



Visualizing Clusters