Lectures 4, 5, and 6:
Unsupervised learning, cluster analysis

Data Science 2
CS 109b, Stat 121b, AC 209b, E-109b

Mark Glickman Pavlos Protopapas

Reading: James et al., chapter 10.

Basics of unsupervised learning:

- Want to discover subgroups among variables or observations, but not in how they relate to a response variable.
- Usually no precise goal of analysis. Often can be used as a descriptive tool.
- Often easier to obtain data without a response variable.

We will focus on principal components analysis (which you have seen in CS109a) and cluster analysis.

Example applications:

- Collect breast cancer patients; want to find subgroups according to similar gene expressions.
- Obtain browser usage and purchasing history of online shoppers; want to find subgroups that relate to their browsing and purchasing patterns.
- Want to group movies according to ratings assigned by movie viewers.

Difference in attitude between PCA and clustering:

Both are multivariate techniques, but with slightly different goals.

- PCA attempts to find a low-dimensional representation of the observations that explains a large fraction of the variation. Often used for variable reduction (as you have seen in CS109a), but can also be helpful for visualization in seeing groupings of data.
- Clustering instead seeks homogeneous subgroups among the observations.

Review of details of PCA:

Assume n observations are measured on p variables or features. Let X_{ij}, for $i = 1, \ldots, n$ and $j = 1, \ldots, p$, be the value of feature j on observation i.

Want to find uncorrelated vectors $\boldsymbol{\phi}_1, \boldsymbol{\phi}_2, \ldots, \boldsymbol{\phi}_p$ with

$$\boldsymbol{\phi}_j = (\phi_{j1}, \phi_{j2}, \ldots, \phi_{jp})$$

where $\left\|\boldsymbol{\phi}_j\right\|^2 = \sum_{k=1}^{p} \phi_{jk}^2 = 1$, and where

$$z_{i1} = \phi_{11} X_{i1} + \phi_{21} X_{i2} + \cdots + \phi_{p1} X_{ip}$$

for $i = 1, \ldots, n$ has the largest possible sample variance, the

$$z_{i2} = \phi_{12} X_{i1} + \phi_{22} X_{i2} + \cdots + \phi_{p2} X_{ip}$$

has the second largest possible sample variance, and so on.

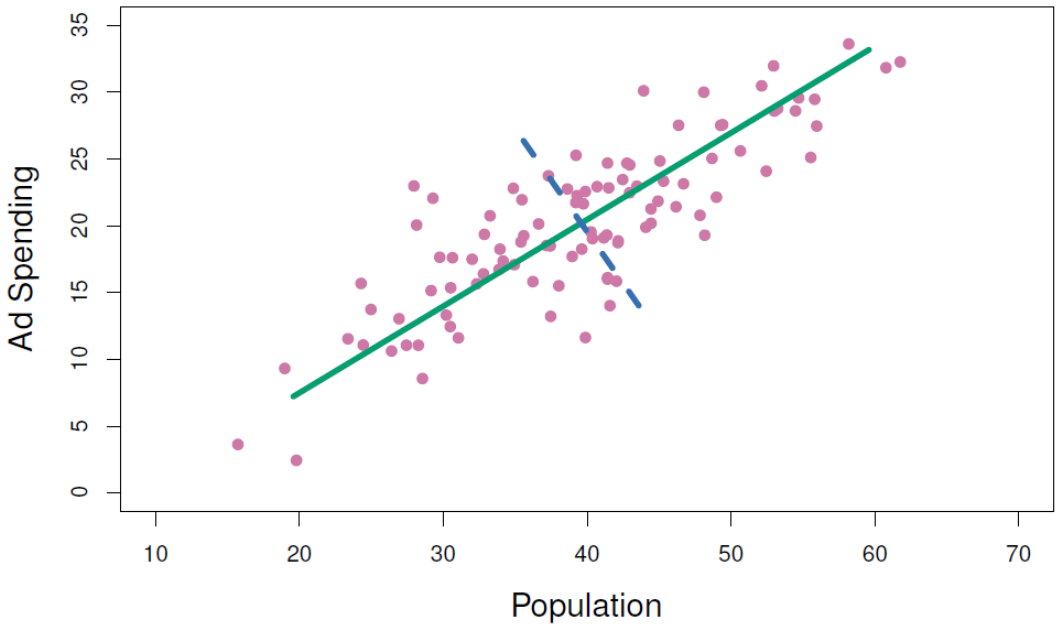
This problem can be solved via a singular-value decomposition of $\boldsymbol{X}$.

Some terminology:

The vector $Z_1 = (z_{11}, \ldots, z_{n1})$ is the first principal component, $Z_2 = (z_{12}, \ldots, z_{n2})$ is the second principal component, and so on.

The j-th "loading vector" $(\phi_{1j}, \phi_{2j}, \ldots, \phi_{pj})$ defines a direction (i.e., a new coordinate axis) in feature space along which the data contain the j-th most variation.

With this procedure, the principal components themselves are uncorrelated.



In the preceding figure, the population size and ad spending for 100 different cities are shown as purple circles.

The green solid line indicates the first principal component direction, and the blue dashed line indicates the second principal component direction.

Usual goal of PCA in unsupervised learning: Produce a scatter plot of first two principal components (PCs).

If most of the data variation is contained in first two PCs, then the scatter plot permits visualization of potential data groupings.

Scaling of variables:

Consider a data set that looks like the following:

```
    X1      X2     X3      X4
435201    0.04   0.21    0.35
839940    0.01   0.33    0.21
582048    0.23   0.19    0.25
390392    0.18   0.30    0.19
...       ...    ...     ...
298989    0.15   0.23    0.27
```

Because basically all of the data variability is in the first feature, PCA will not reorient the data in a meaningful way.

Usually want to ensure each variable has equal contribution to PCA.

Solution: Standardize the variables prior to computing distances.

Standardizing usually just involves transforming each X_{ij} by

$$\frac{X_{ij} - \text{center}(\boldsymbol{x}_j)}{\text{scale}(\boldsymbol{x}_j)}$$

where $\text{center}(\boldsymbol{x}_j)$ can be the sample mean or median of the j-th variable, and $\text{scale}(\boldsymbol{x}_j)$ can be the sample standard deviation or mean absolute deviation of the j-th variable.

Main example: Violent crime rates by State

This data set reports arrests per 100,000 residents for assault, murder, and rape in each of the 50 US states in 1973. Also given is the percent of the population living in urban areas.

Summary:

```
> summary(USArrests)
     Murder          Assault         UrbanPop          Rape
 Min.   : 0.800   Min.   : 45.0   Min.   :32.00   Min.   : 7.30
 1st Qu.: 4.075   1st Qu.:109.0   1st Qu.:54.50   1st Qu.:15.07
 Median : 7.250   Median :159.0   Median :66.00   Median :20.10
 Mean   : 7.788   Mean   :170.8   Mean   :65.54   Mean   :21.23
 3rd Qu.:11.250   3rd Qu.:249.0   3rd Qu.:77.75   3rd Qu.:26.18
 Max.   :17.400   Max.   :337.0   Max.   :91.00   Max.   :46.00
```

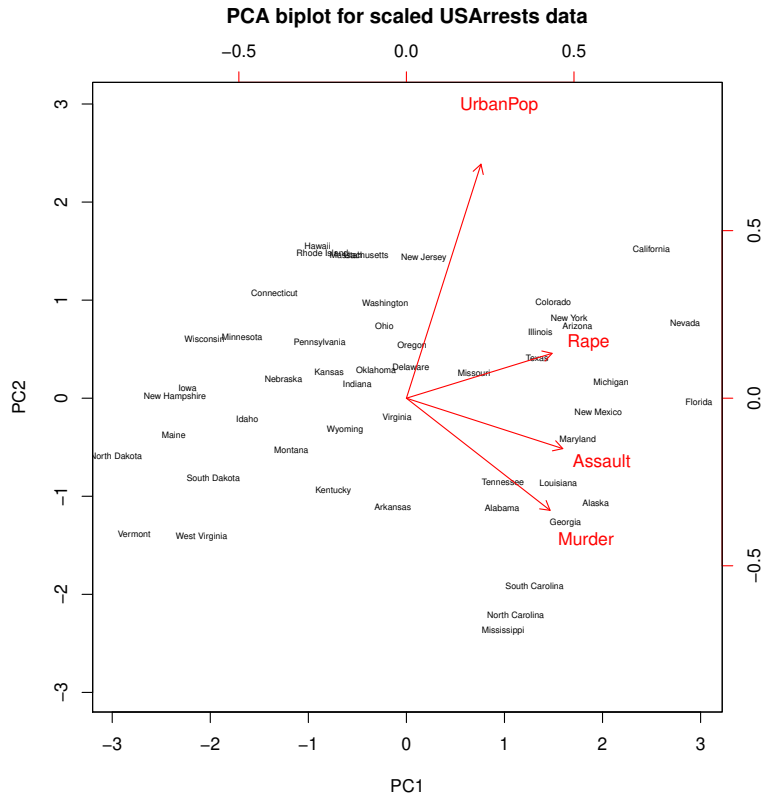


Figure contains a scatter plot of the first two principal components.

- Each state is plotted according to its first two principal components.
- The red arrows indicate the first two PC loading vectors (axes on the top and right). For example, the loading for `Rape` on the first PC is 0.543, and 0.167 on the second PC [the word `Rape` is centered at (0.543, 0.167)].
- This figure is known as a "biplot" as it displays both the PC scores and the PC loadings.

PCA loadings:

```
             PC1        PC2        PC3         PC4
Murder   0.5358995 -0.4181809  0.3412327 -0.64922780
Assault  0.5831836 -0.1879856  0.2681484  0.74340748
UrbanPop 0.2781909  0.8728062  0.3780158 -0.13387773
Rape     0.5434321  0.1673186 -0.8177779 -0.08902432
```

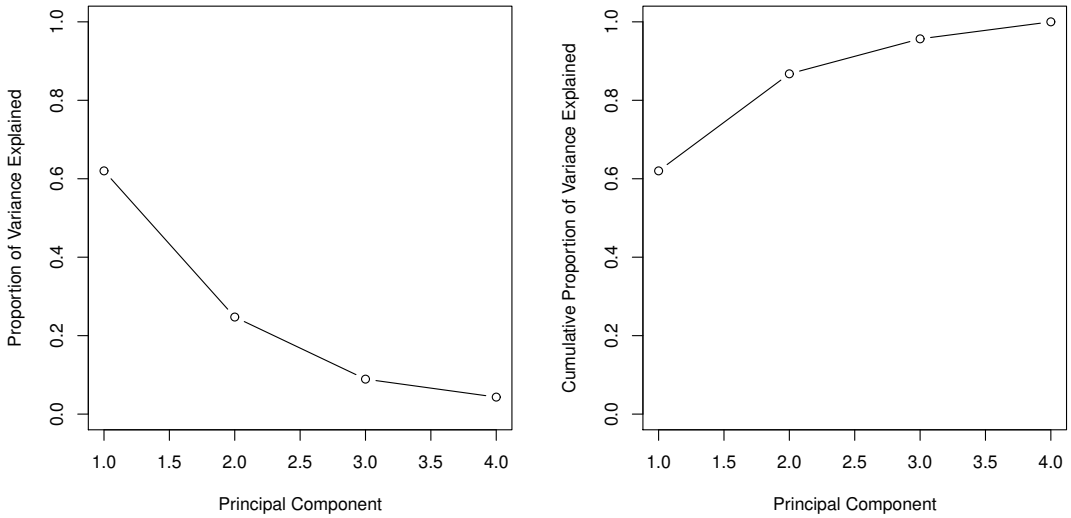
Are two principal components enough?

Idea: Plot the cumulative "proportion variance explained" (PVE) as a function of the number of principal components.

Assuming the columns of $\boldsymbol{X}$ have been centered and scaled, The PVE of the m-th principal component is

$$\text{PVE}_m = \frac{1}{p}\sum_{i=1}^{n} z_{im}^2.$$

USArrests – proportion variance explained



Proportion variance explained:

- About 86.8% of the variance is explained by the first two PCs.
- No principled method (such as cross-validation) to determine the number of PCs to retain. Nothing to validate against.
- Sometimes it has been suggested to look for an "elbow" on the plot of proportion of variances (on the left) as the number of PCs to retain. But not relevant for visualization.

Cluster analysis outline:

- Inter-observation distances
- Partition-based clustering
- Hierarchical clustering

- Soft clustering
- Diagnostics, and optimizing the number of clusters
- Plots, plots, plots and more plots

Clustering:

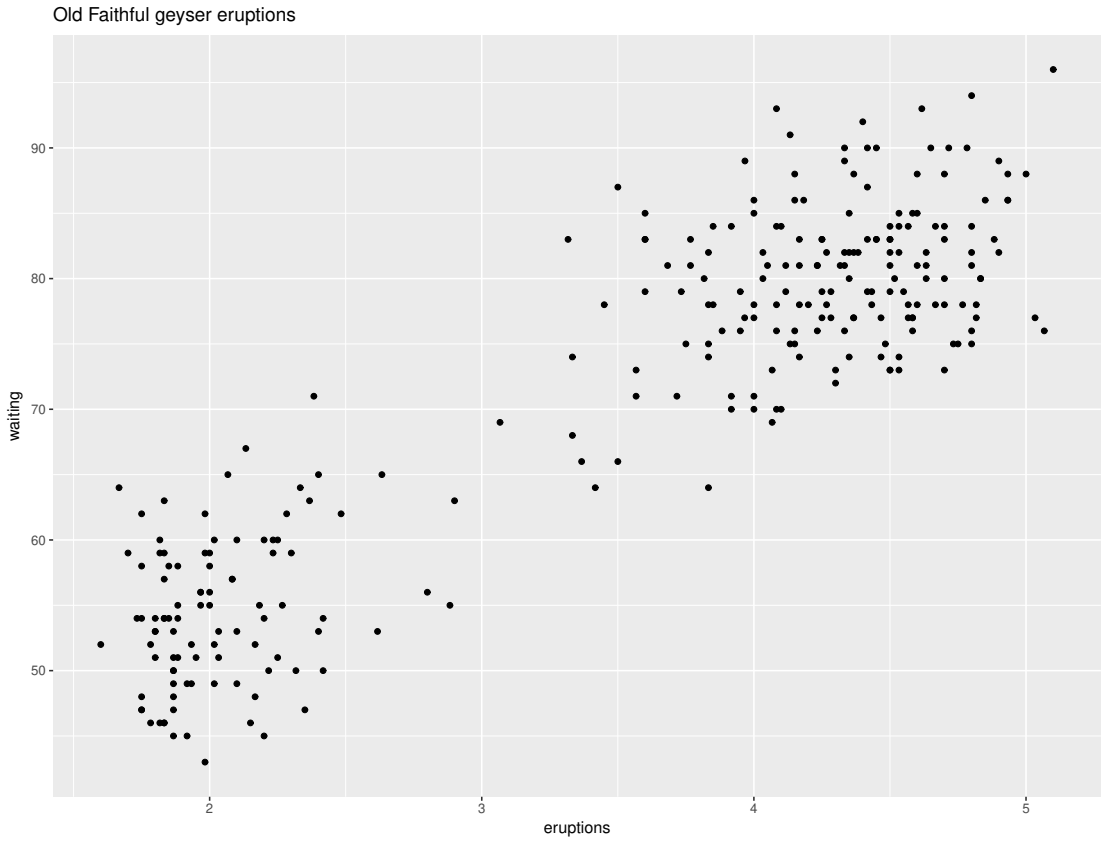
- Cluster analysis consist of techniques for finding subgroups or clusters in a multivariate data set.
- Want to partition the data into distinct groups so that observations within each group are similar to each other.
- Need to define concretely what it means for observations to be similar or different. This is usually domain-specific.

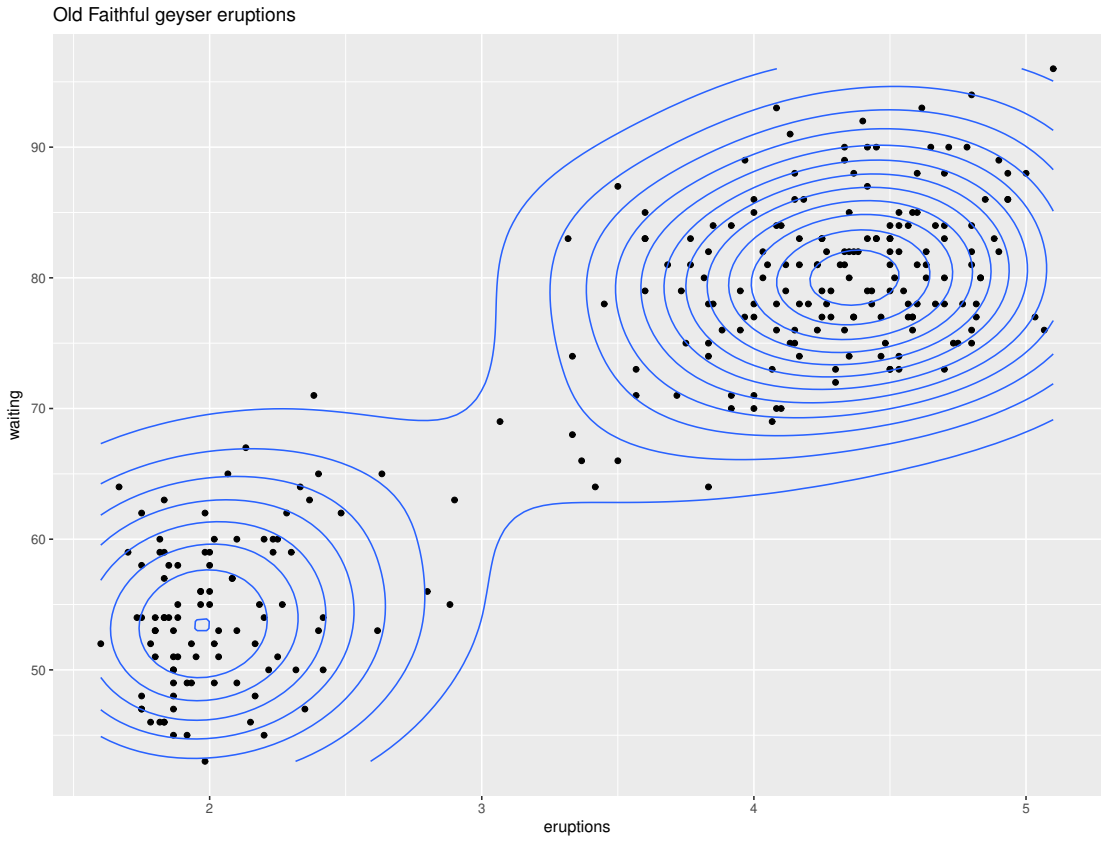
A good portion of these notes are based on material appearing on `www.sthda.com`.

Simple example: Eruptions of Old Faithful

Data were collected on time between eruptions and the duration of eruptions for the Old Faithful geyser at Yellowstone National Park.

```
> data("faithful")
> summary(faithful)
   eruptions        waiting
 Min.   :1.600   Min.   :43.0
 1st Qu.:2.163   1st Qu.:58.0
 Median :4.000   Median :76.0
 Mean   :3.488   Mean   :70.9
 3rd Qu.:4.454   3rd Qu.:82.0
 Max.   :5.100   Max.   :96.0
```


Old Faithful geyser eruptions
waiting
eruptions
90
80
70
60
50
2
3
4
5


Old Faithful geyser eruptions
waiting
eruptions
90
80
70
60
50
2
3
4
5

Three types of approaches to clustering:

- Partitioning clustering - specify the number of clusters in advance, and then invoke an algorithm to partition the data
- Hierarchical clustering - iteratively merge (or divide) the data typically one observation at a time, and then decide on partitions afterward
- Soft clustering - assume each observation can simultaneously be a member of every cluster, with different probabilities computed for cluster membership

Distances between observations: The building block of clustering

- The choice of distance measures is a critical step in clustering. The pairwise distance calculation can influence the shape of the clusters.
- Observations to be clustered can be non-standard (e.g., pictures, audio signals), so the process of computing distances first followed by clustering can be an effective approach.

Assume n observations are measured on p variables or features. Let X_{ij}, for $i = 1, \ldots, n$ and $j = 1, \ldots, p$, be the value of feature j on observation i.

Two common distance measures: For observations $\boldsymbol{x}_i$ and $\boldsymbol{x}_k$ (of length p)

$$d_{Euc}(\boldsymbol{x}_i, \boldsymbol{x}_k) = \sqrt{\sum_{j=1}^{p}(X_{ij} - X_{kj})^2} \quad \text{Euclidean distance } (L_2)$$

$$d_{Man}(\boldsymbol{x}_i, \boldsymbol{x}_k) = \sum_{j=1}^{p} |X_{ij} - X_{kj}| \quad \text{Manhattan distance } (L_1)$$

Other distance measures: Correlation-based measures

$$d_{Cor}(\boldsymbol{x}_i, \boldsymbol{x}_k) = 1 - \frac{\sum_{j=1}^{p}(X_{ij} - \bar{x}_i)(X_{kj} - \bar{x}_k)}{\sqrt{\sum_{j=1}^{p}(X_{ij} - \bar{x}_i)^2 \sum_{j=1}^{p}(X_{kj} - \bar{x}_k)^2}} \quad \text{Pearson correlation distance}$$

$$d_{Spear}(\boldsymbol{x}_i, \boldsymbol{x}_k) = 1 - \frac{\sum_{j=1}^{p}(W_{ij} - \bar{w}_i)(W_{kj} - \bar{w}_k)}{\sqrt{\sum_{j=1}^{p}(W_{ij} - \bar{w}_i)^2 \sum_{j=1}^{p}(W_{kj} - \bar{w}_k)^2}} \quad \text{Spearman correlation distance}$$

where W_{ij} are the ranks of X_{ij} for feature j, and $\bar{w}_i$ is the average of the ranks for observation i.

Correlation distances are often used for micro-array analyses.

Spearman correlation is used when outliers might be a concern in the data.

Gower distance:

Some multivariate data consist of a mix of quantitative, ordinal and nominal data types.

The Gower distance between two multivariate observations with a mix of different types is computed in the following way.

- Quantitative: Use (standardized) Manhattan distance
- Ordinal: Convert to ranks, then use (standardized) Manhattan distance
- Nominal: Record 0 if the category matches, and 1 if the categories do not match.

Sum the contribution of each variable to obtain a distance.

Distance and scaling: Same issue as with PCA

Reconsider the data set that looks like the following:

```
    X1      X2     X3      X4
435201    0.04   0.21    0.35
839940    0.01   0.33    0.21
582048    0.23   0.19    0.25
390392    0.18   0.30    0.19
...        ...    ...     ...
298989    0.15   0.23    0.27
```

The distance between any two observations is basically determined by feature 1. This is an undesired consequence of variables having different scales.

Usually want to ensure each variable has equal contribution to the clustering algorithm, and therefore the distance computation.

Solution: Standardize the variables prior to computing distances.

Same procedure as with PCA.

Distances and clustering in R: Meet the lovely ladies

The R package `cluster` contains a number of helpful functions for computing distances and for clustering.

The first is DAISY, which computes pairwise distances (or "dissimilarities"). By default, DAISY standardizes variables if the data is all of numerical type.

We will see other `cluster` functions shortly.

Distances and violent crimes:

First few randomly chosen observations

```
             Murder Assault UrbanPop Rape
Iowa            2.2      56       57 11.3
Rhode Island    3.4     174       87  8.3
Maryland       11.3     300       67 27.8
Tennessee      13.2     188       59 26.9
Utah            3.2     120       80 22.9
Arizona         8.1     294       80 31.0
```

Now rescale the observations (subtract mean, divide by std dev):

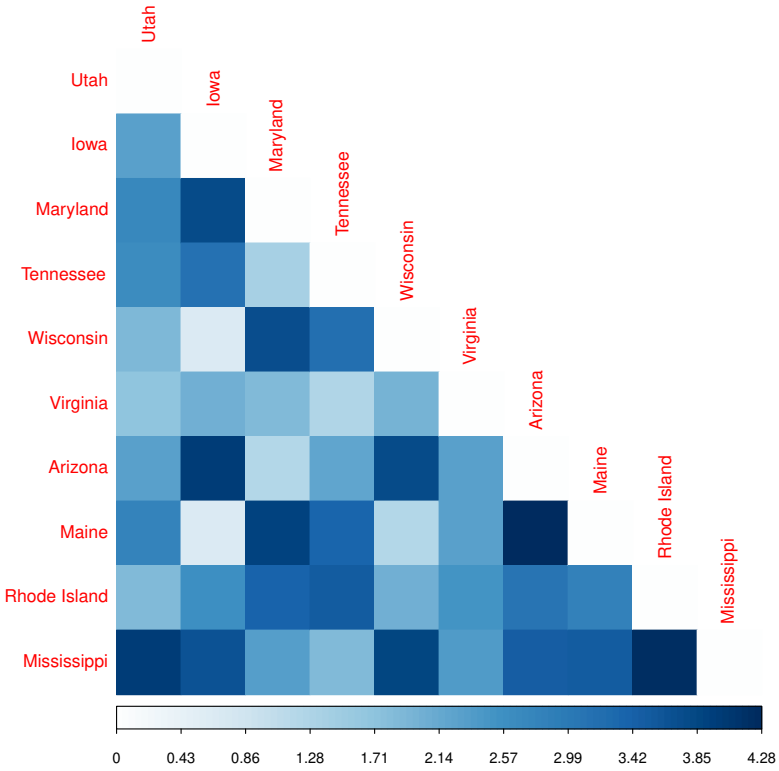
```
> arrests.scaled = scale(arrests) # data in object 'arrests'
```

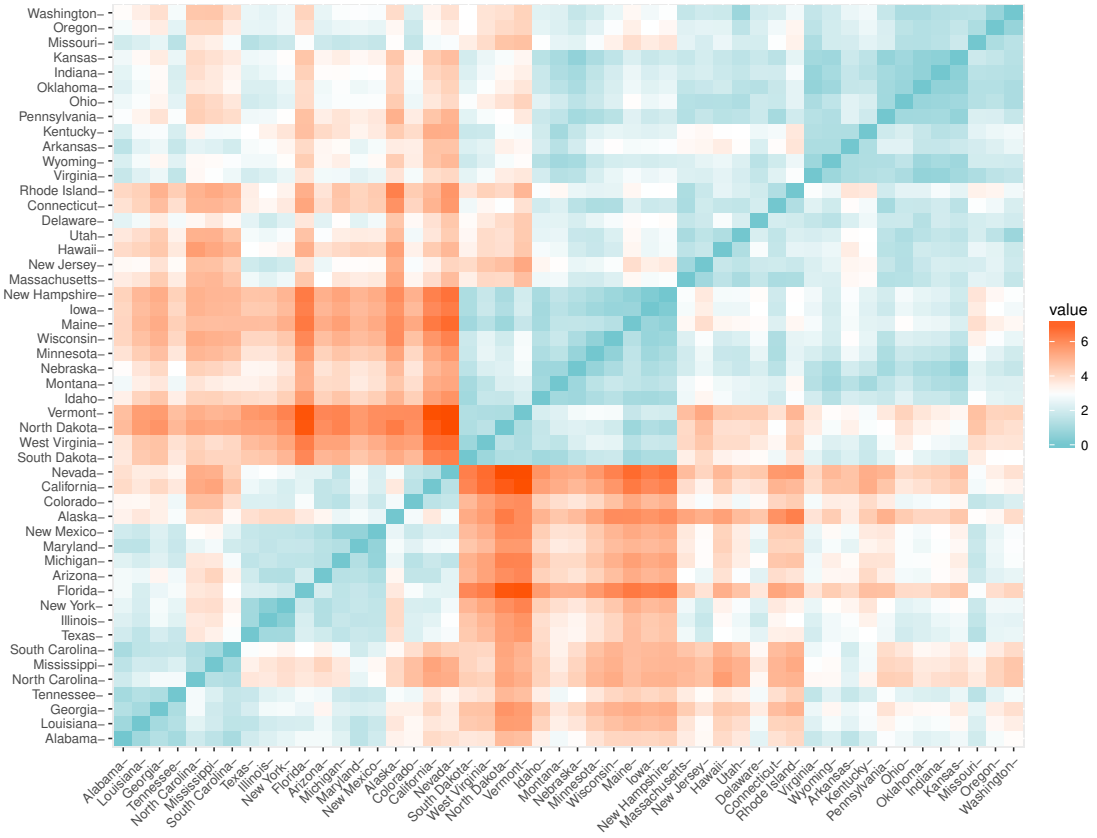
```
             Murder Assault UrbanPop  Rape
Iowa          -0.95   -1.21    -0.62 -0.83
Rhode Island  -0.72    0.06     1.58 -1.18
Maryland       0.82    1.42     0.12  1.08
Tennessee      1.19    0.21    -0.47  0.98
Utah          -0.75   -0.52     1.07  0.51
Arizona        0.20    1.35     1.07  1.45
```

Euclidean distances on scaled observations:

```
> dist.eucl = daisy(arrests.scaled, metric = "euclidean")
> round(as.matrix(dist.eucl)[1:6, 1:6], 1)
```

```
             Iowa Rhode Island Maryland Tennessee Utah Arizona
Iowa          0.0          2.6      3.8       3.1  2.3     4.0
Rhode Island  2.6          0.0      3.4       3.5  1.9     3.1
Maryland      3.8          3.4      0.0       1.4  2.7     1.2
Tennessee     3.1          3.5      1.4       0.0  2.6     2.2
Utah          2.3          1.9      2.7       2.6  0.0     2.3
Arizona       4.0          3.1      1.2       2.2  2.3     0.0
```


Utah
Iowa
Maryland
Tennessee
Wisconsin
Virginia
Arizona
Maine
Rhode Island
Mississippi
0
0.43
0.86
1.28
1.71
2.14
2.57
2.99
3.42
3.85
4.28


value
6
4
2
0

Clustering via partitioning methods:

Basic idea - specify the number of clusters into which the data will be partitioned, and then perform computation to group data so that

1. observations within clusters are similar (low distances/dissimilarities), and
2. observations in different clusters are dissimilar (high distances/dissimilarities)

Separately can address the optimal number of clusters.

We will examine two partitioning algorithms:

- K-means clustering, where each cluster is represented by the centroid (mean of the data points) in each cluster
- K-medioids clustering, where each cluster is represented by the medioid (one particular observation in the "middle") of each cluster.

K-means clustering:

Let $C_1, \ldots, C_K$ denote sets containing the indices of observations within the K clusters such that

1. $C_1 \cup C_2 \cup \cdots \cup C_K = \{1, 2, \ldots, n\}$, and
2. $C_k \cap C_\ell = \varnothing$ for $k \neq \ell$.

Let

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

be the within-cluster variation.

An interesting mathematical fact that we will use shortly is that

$$W(C_k) = 2 \sum_{i \in C_k} \sum_{j=1}^{p} (X_{ij} - \bar{x}_{jk})^2$$

where $\bar{x}_{jk}$ is the sample mean of the X_{ij} for $i \in C_k$.

Goal of K-means clustering:

Determine $C_1, \ldots, C_K$ such that the expression

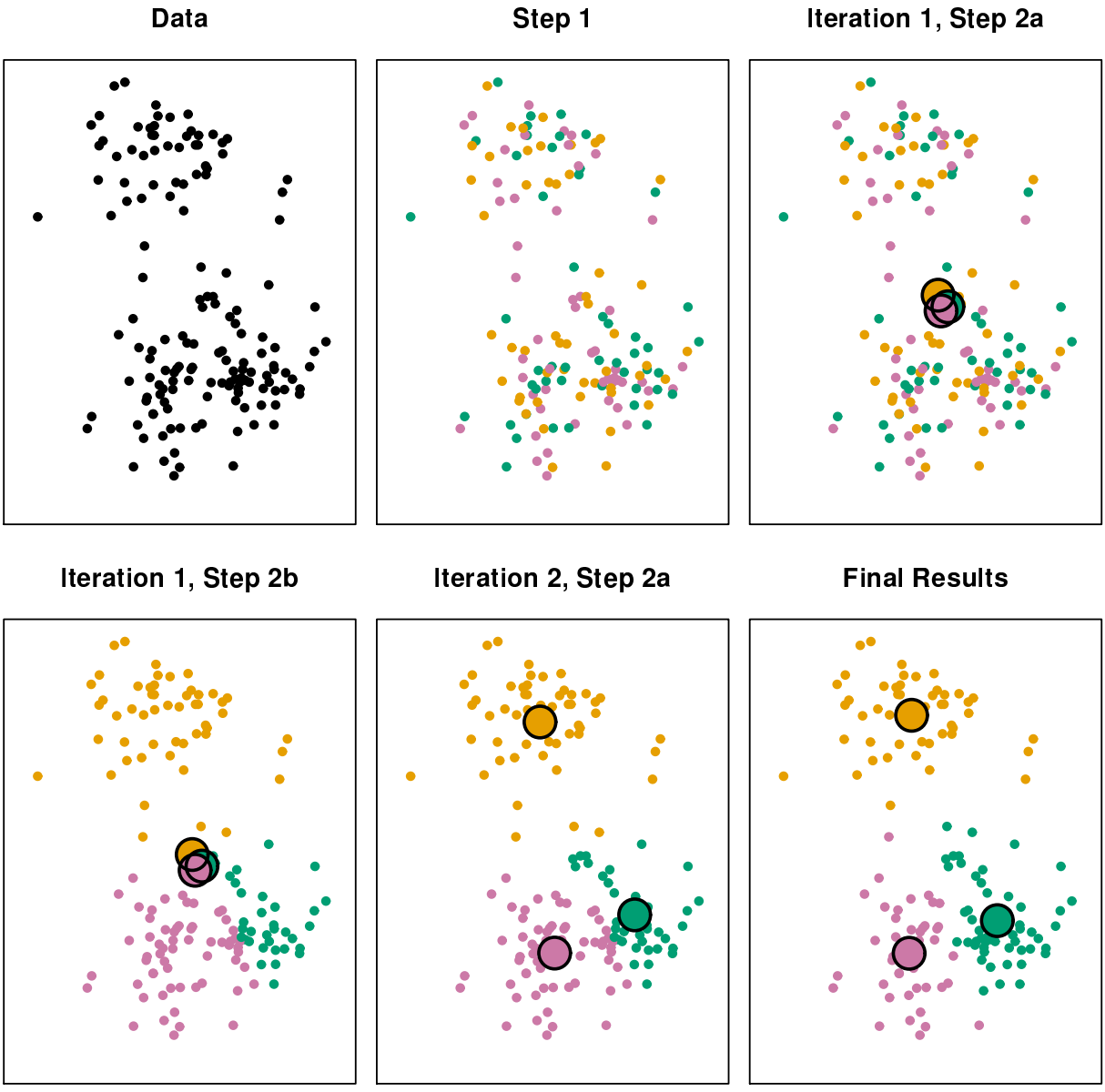
$$\sum_{k=1}^{K} W(C_k)$$

is minimized.

K-means clustering algorithm:

1. Randomly assign each observation to one of K clusters at random.
2. Repeat the following two steps until clusters do not change:
   (a) For each cluster k, compute the cluster centroid $\bar{x}_k$ (the variable-wise average of the observations in cluster k).
   (b) Given the k centroids, reassign all observations to clusters based on their closeness to the centroids.

This procedure is implemented in the `kmeans` function within R.



The bad and the good:

- Requires analyst to select K in advance.

- The algorithm is locally optimal, not globally optimal. As a consequence this means you can get different clusterings depending on the starting cluster assignment.

Potential solutions:

- Can try various values of K and compare results. We will dig deeper into this approach shortly.
- Can try different initial cluster assignments in parallel, and then choose the solution with the best within-cluster sum of squared deviations.

Application to violent crimes data:

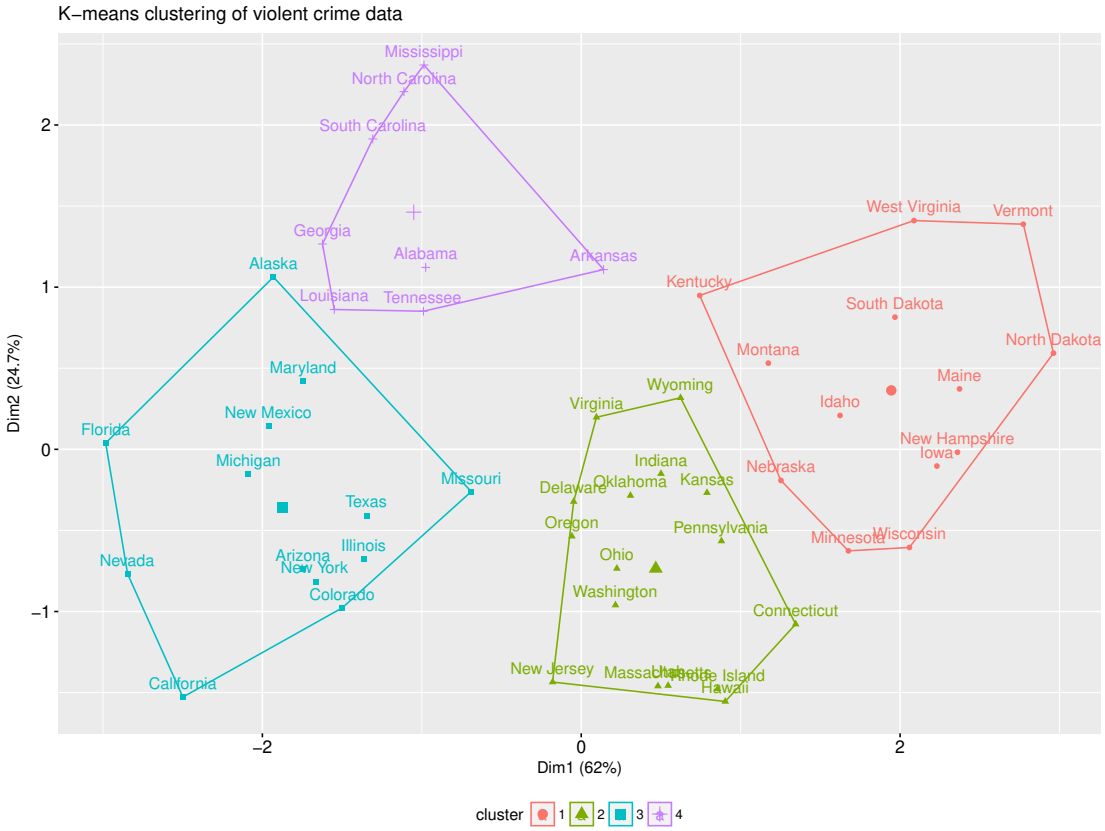
Choose 4 clusters – we will see later why 4 is reasonable.

```
> arrests.km = kmeans(scale(USArrests), 4, nstart = 25)
  # nstart is number of random starting assignments
> print(arrests.km)

K-means clustering with 4 clusters of sizes 13, 16, 13, 8

Cluster means:
      Murder    Assault   UrbanPop        Rape
1 -0.9615407 -1.1066010 -0.9301069 -0.96676331
2 -0.4894375 -0.3826001  0.5758298 -0.26165379
3  0.6950701  1.0394414  0.7226370  1.27693964
4  1.4118898  0.8743346 -0.8145211  0.01927104


Clustering vector:
       Alabama         Alaska        Arizona       Arkansas     California
             4              3              3              4              3
      Colorado    Connecticut       Delaware        Florida        Georgia
             3              2              2              3              4
        Hawaii          Idaho       Illinois        Indiana           Iowa
             2              1              3              2              1
        Kansas       Kentucky      Louisiana          Maine       Maryland
             2              1              4              1              3
 Massachusetts       Michigan      Minnesota    Mississippi       Missouri
             2              3              1              4              3
       Montana       Nebraska         Nevada  New Hampshire     New Jersey
             1              1              3              1              2
    New Mexico       New York North Carolina   North Dakota           Ohio
             3              3              4              1              2
      Oklahoma         Oregon   Pennsylvania   Rhode Island South Carolina
             2              2              2              2              4
  South Dakota      Tennessee          Texas           Utah        Vermont
             1              4              3              2              1
      Virginia     Washington  West Virginia      Wisconsin        Wyoming
             2              2              1              1              2


Within cluster sum of squares by cluster:
[1] 11.952463 16.212213 19.922437  8.316061
 (between_SS / total_SS =  71.2 %)
```

Can compute the (unscaled) mean of each variable within cluster:

```
> aggregate(USArrests, by=list(cluster=arrests.km$cluster), mean)

  cluster   Murder   Assault UrbanPop     Rape
1       1  3.60000  78.53846 52.07692 12.17692
```

```
2        2  5.65625 138.87500 73.87500 18.78125
3        3 10.81538 257.38462 76.00000 33.19231
4        4 13.93750 243.62500 53.75000 21.41250
```



Alternative to K-means: Partitioning around medioids

Also known as PAM.

Goal of PAM: Search for K representative observations that are to be the medioids of clusters.

- These K observations are chosen to minimize the sum of the dissimilarities/distances of observations to their closest representative observation.
- The sum of the dissimilarities is usually calculated using Manhattan (L_1) distances.

As with K-means, can perform the clustering based on the scaled data, or based on the pairwise distances (though PAM assumes L_1 distances)

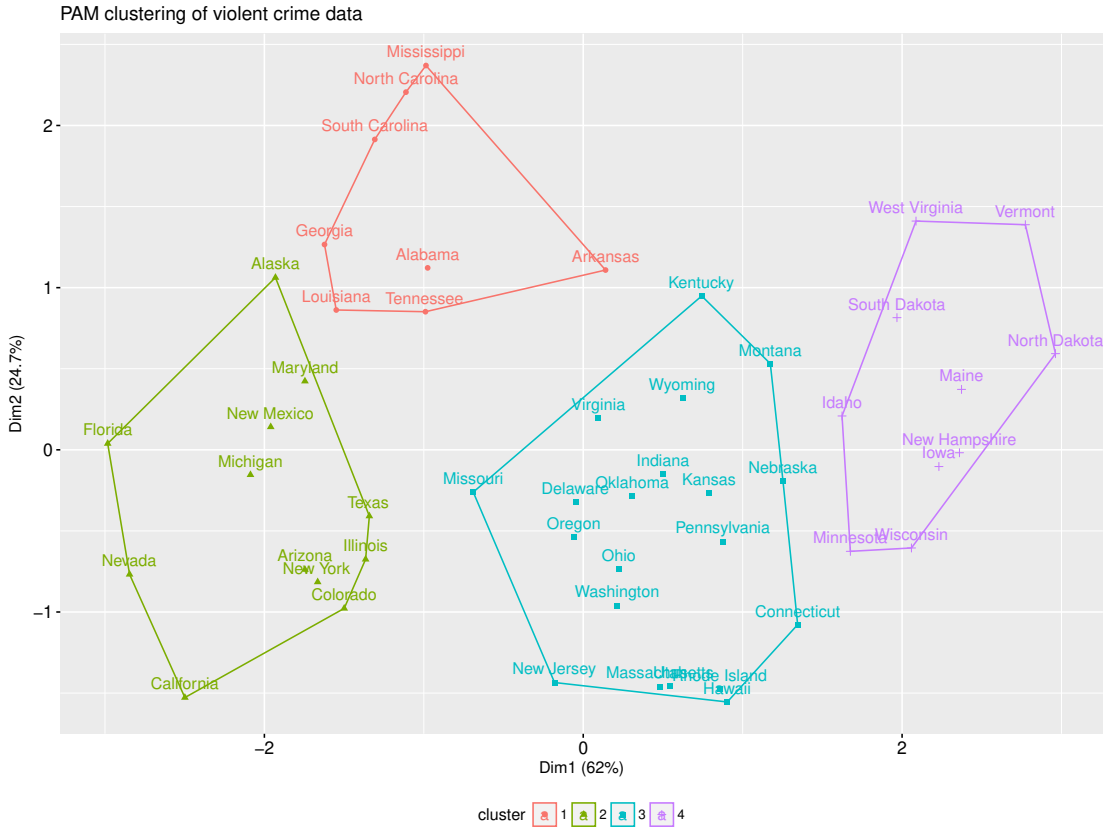
Unlike K-means, PAM is robust to outliers.

The algorithm for PAM is computationally intensive. For large data sets, PAM may require too much memory of computation time. Instead, use CLARA ("clustering large applications").

Reference for clustering algorithm: Chapter 2 of Rousseeuw, P. J., & Kaufman, L. (1990). Finding Groups in Data. Wiley Online Library.

Application to violent crimes data:

```
> arrests.pam = pam(scale(USArrests), k=4)
> arrests.pam
Medoids:
               ID      Murder     Assault   UrbanPop          Rape
Alabama         1  1.2425641   0.7828393 -0.5209066 -0.003416473
Michigan       22  0.9900104   1.0108275  0.5844655  1.480613993
Oklahoma       36 -0.2727580 -0.2371077  0.1699510 -0.131534211
New Hampshire 29 -1.3059321 -1.3650491 -0.6590781 -1.252564419

Clustering vector:
  Alabama          Alaska         Arizona        Arkansas      California
        1               2               2               1               2
 Colorado     Connecticut        Delaware         Florida         Georgia
        2               3               3               2               1
...
```



Silhouette plot: A diagnostic for PAM (and other) clustering

Once a clustering has been determined, let

a_i = average dissimilarity between observation i and the other points in the cluster to which i belongs

b_i = average dissimilarity between observation i and the other points in the next closest cluster to observation i

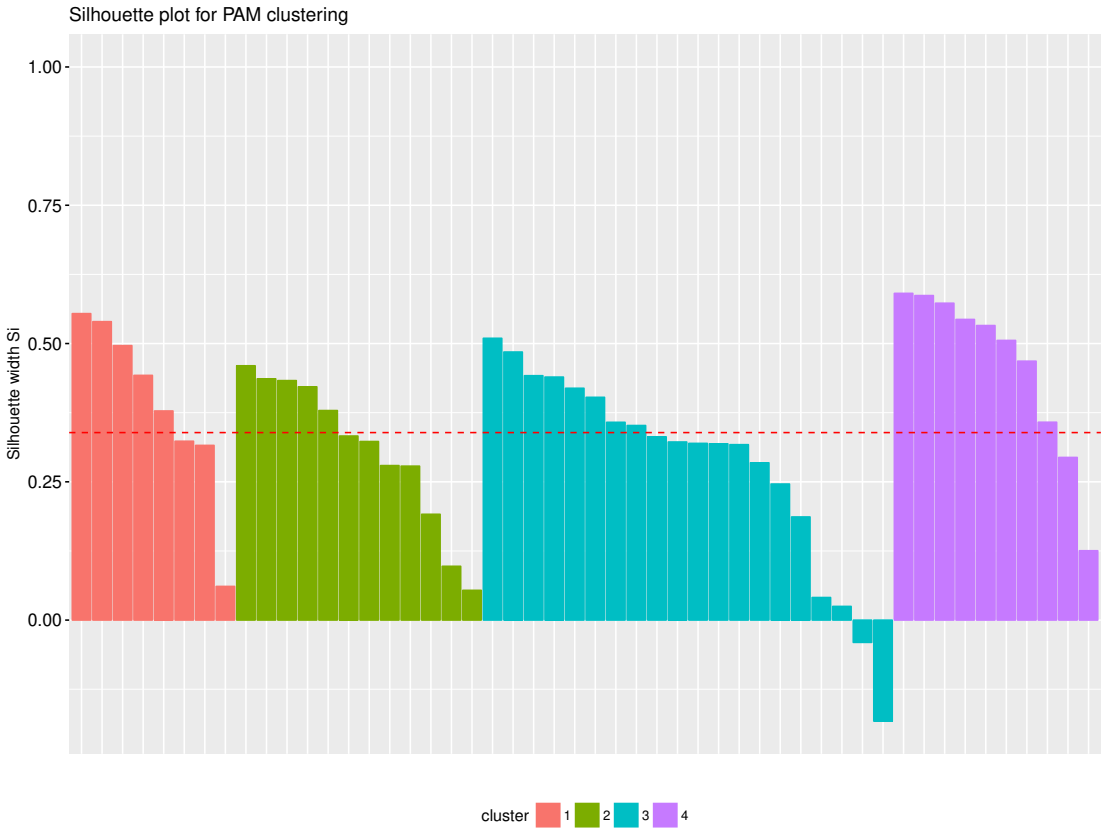
Let

$$s_i = \frac{b_i - a_i}{\max(a_i, b_i)}$$

be the silhouette for observation i.

Interpretation:

- Observations with $s_i \approx 1$ are well-clustered
- Observations with $s_i \approx 0$ lie between two clusters
- Observations with $s_i < 0$ are probably in the wrong cluster.



A few observations may be in the wrong cluster!

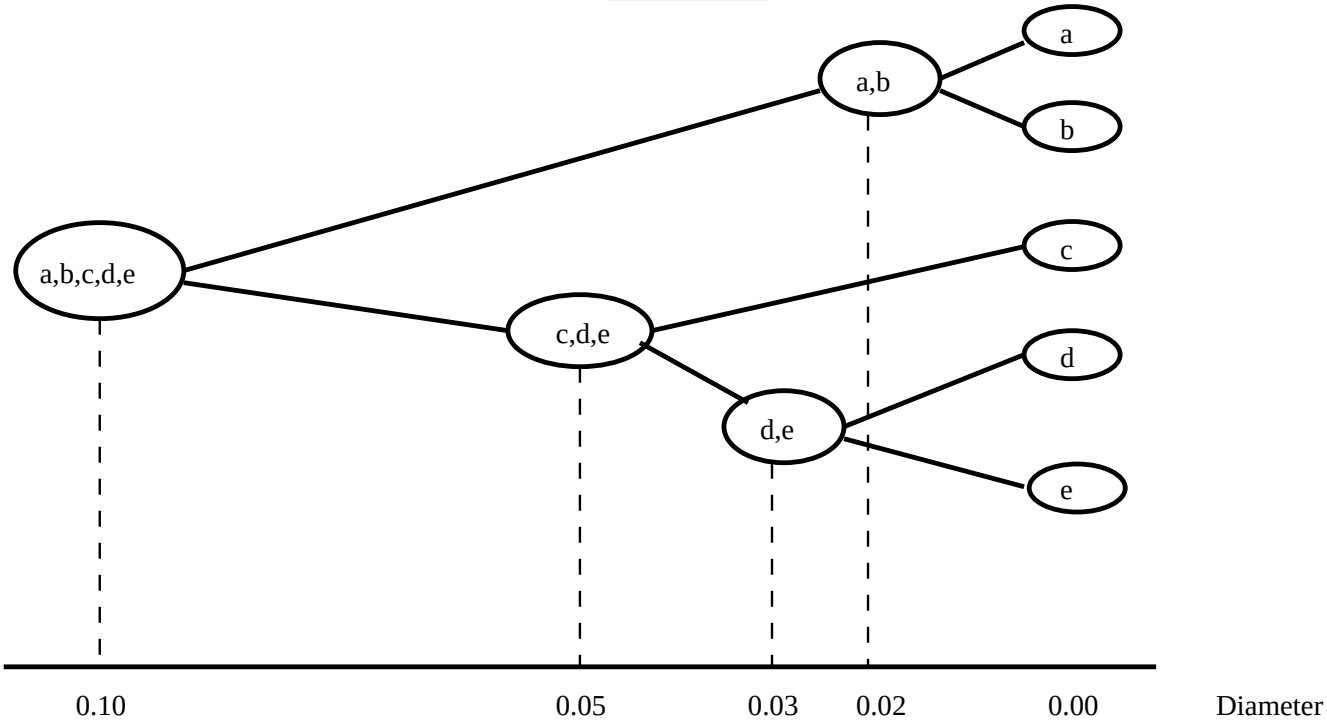
```
> # Compute silhouette
> sil = silhouette(arrests.pam)[, 1:3]
> # Objects with negative silhouette
> neg_sil_index = which(sil[, 'sil_width'] < 0)
> print(sil[neg_sil_index, , drop = FALSE])

          cluster neighbor   sil_width
Nebraska        3        4 -0.04034739
Montana         3        4 -0.18266793
```

Check cluster plot to confirm.

Hierarchical clustering:

- K-means clustering requires pre-specifying the number of clusters K.
  (though we will come back to address this point)
- Hierarchical clustering does not require committing to a particular number of clusters, K.
- Two types of hierarchical clustering:
  - Agglomerative clustering (bottom-up approach).
  - Divisive clustering (top-down approach).

Can summarize hierarchical clustering in a dendrogram



Agglomerative clustering:

The basic algorithm:

- Each observation starts as its own cluster.
- At each step of the algorithm, two clusters that are most "similar (to be described shortly) are combined into a new larger cluster.
- This process of combining clusters is repeated until all observations are members of one single large cluster.

The procedure in the `cluster` R library is called AGNES (agglomerative nesting).

Particularly well-suited at identifying small clusters.

Agglomerative clustering: How do we measure the dissimilarity between two clusters?

A few common approaches:

Complete (or maximum) linkage clustering: For two clusters, determine the maximum dissimilarity between any observation in the first cluster and any observation in the second cluster.

Single linkage clustering: For two cluster, determine the minimum dissimilarity between any observation in the first cluster and any observation in the second cluster.

Average linkage clustering: Compute all pairwise dissimilarities between observations in the first and second cluster, and calculate the average.

Another popular approach is Ward's method.

- Instead of joining two clusters based on their distances, use information about the variance of observations within clusters.
- Specifically, at each step, join two clusters whose merged cluster has the smallest within-cluster sum of squared distances.

Other aggregation approaches are possible.

Divisive clustering:

The basic algorithm:

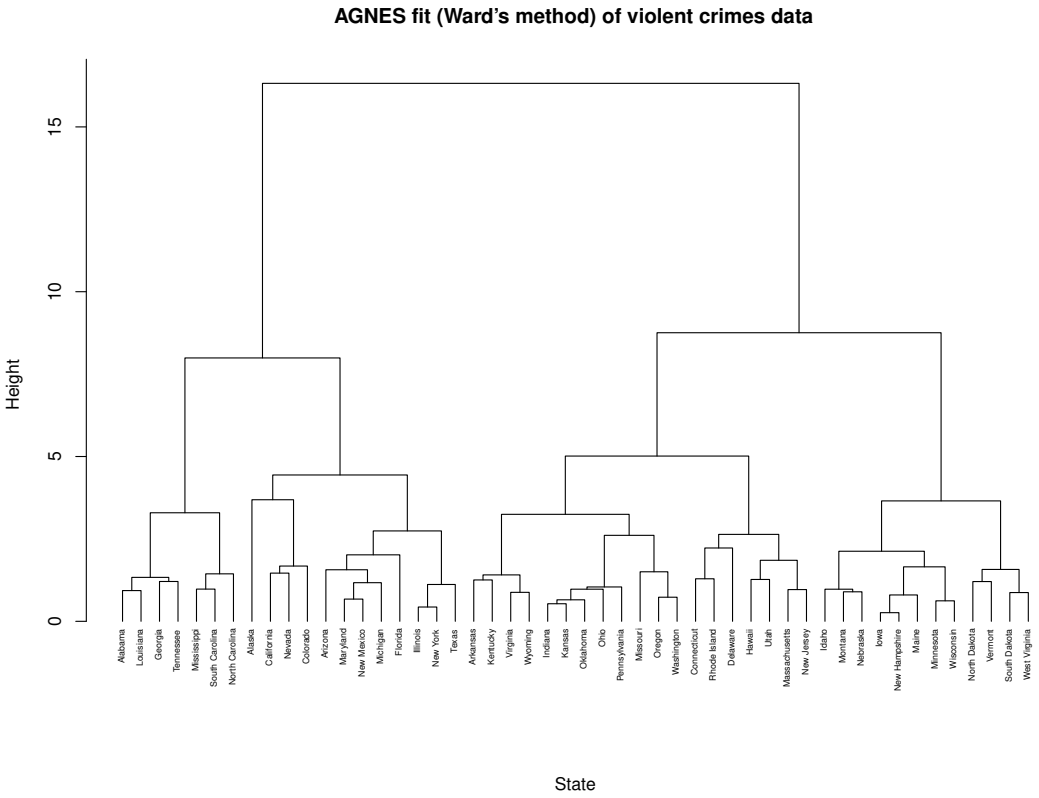
- Each observation starts as a member of one large cluster.
- At each step of the algorithm, the cluster with the greatest heterogeneity is divided into two clusters.
- This process of dividing clusters is repeated until all observations are members of their own cluster.

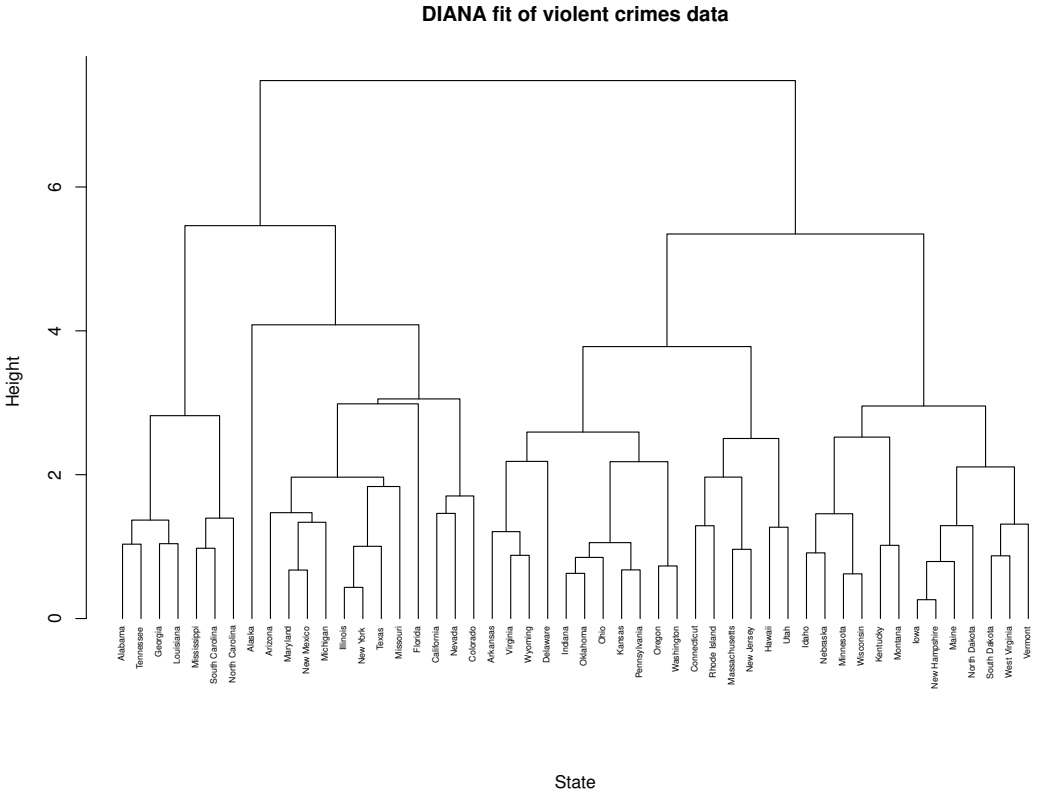
The procedure in the `cluster` R library is called DIANA (divisive analysis).

Particularly good at identifying large clusters, but less commonly used than agglomerative clustering.

Steps for divisive clustering in DIANA:

- Identify the cluster with the largest diameter, i.e., largest dissimilarity between two members.
- Find the observation within this cluster that has the largest average dissimilarity with the other cluster members; this observation starts the formation of a "splinter" group.
- Iteratively reassign observations from the original cluster to the splinter cluster if they are closer to the splinter group.

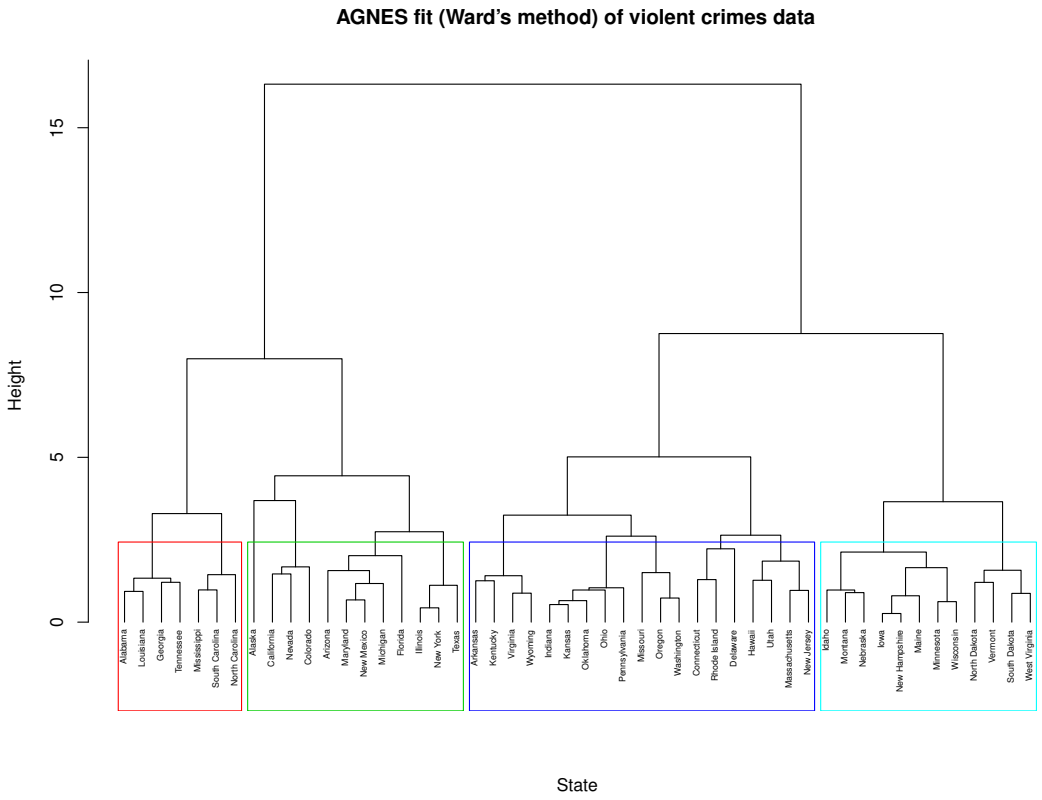

AGNES fit (Ward's method) of violent crimes data
Height
0
5
10
15
Alabama
Louisiana
Georgia
Tennessee
Mississippi
South Carolina
North Carolina
Alaska
California
Nevada
Colorado
Arizona
Maryland
New Mexico
Michigan
Florida
Illinois
New York
Texas
Arkansas
Kentucky
Virginia
Wyoming
Indiana
Kansas
Oklahoma
Ohio
Pennsylvania
Missouri
Oregon
Washington
Connecticut
Rhode Island
Delaware
Hawaii
Utah
Massachusetts
New Jersey
Idaho
Montana
Nebraska
Iowa
New Hampshire
Maine
Minnesota
Wisconsin
North Dakota
Vermont
South Dakota
West Virginia
State

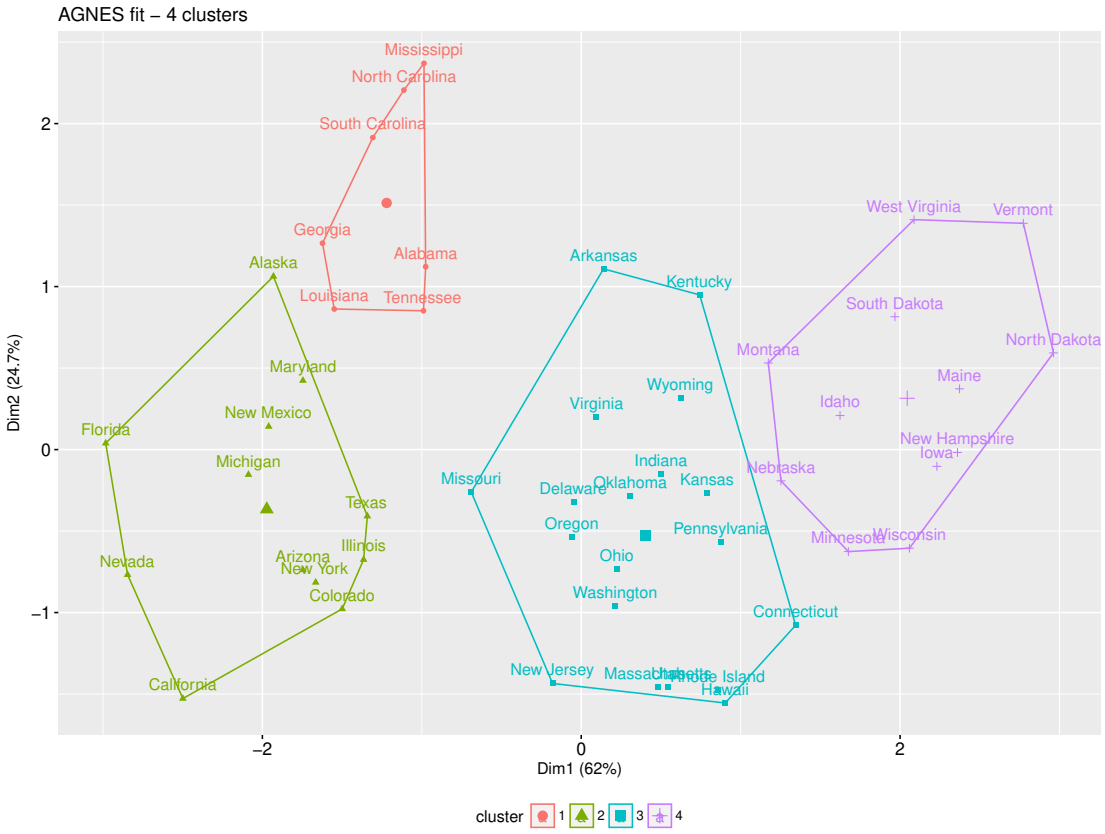

DIANA fit of violent crimes data
Height
0
2
4
6
Alabama
Tennessee
Georgia
Louisiana
Mississippi
South Carolina
North Carolina
Alaska
Arizona
Maryland
New Mexico
Michigan
Illinois
New York
Texas
Missouri
Florida
California
Nevada
Colorado
Arkansas
Virginia
Wyoming
Delaware
Indiana
Oklahoma
Ohio
Kansas
Pennsylvania
Oregon
Washington
Connecticut
Rhode Island
Massachusetts
New Jersey
Hawaii
Utah
Idaho
Nebraska
Minnesota
Wisconsin
Kentucky
Montana
Iowa
New Hampshire
Maine
North Dakota
South Dakota
West Virginia
Vermont
State

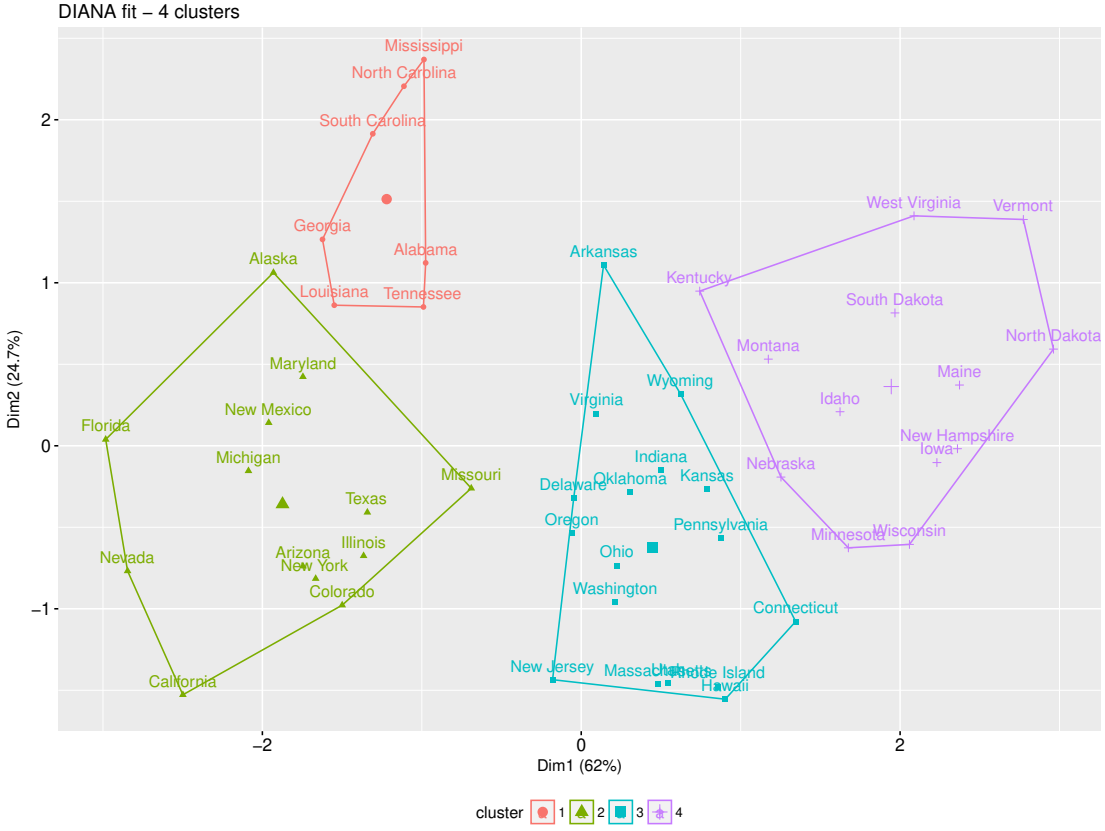
Some additional graphical summaries:

If we identify the optimal number of clusters to be 4 (which we will investigate shortly), we can

- modify a dendrogram to highlight the clusters
- produce a principal components plot of points with the clusters drawn




AGNES fit – 4 clusters
Mississippi
North Carolina
South Carolina
Georgia
Alabama
Louisiana
Tennessee
Alaska
Maryland
New Mexico
Florida
Michigan
Texas
Nevada
Arizona
Illinois
New York
Colorado
California
Arkansas
Kentucky
Virginia
Wyoming
Missouri
Indiana
Delaware
Oklahoma
Kansas
Oregon
Pennsylvania
Ohio
Washington
Connecticut
New Jersey
Massachusetts
Rhode Island
Hawaii
West Virginia
Vermont
South Dakota
North Dakota
Montana
Maine
Idaho
New Hampshire
Iowa
Nebraska
Minnesota
Wisconsin
Dim2 (24.7%)
Dim1 (62%)
cluster 1 2 3 4


DIANA fit – 4 clusters
Mississippi
North Carolina
South Carolina
Georgia
Alabama
Louisiana
Tennessee
Alaska
Maryland
New Mexico
Florida
Michigan
Missouri
Texas
Nevada
Arizona
Illinois
New York
Colorado
California
Arkansas
Kentucky
Virginia
Wyoming
Indiana
Delaware
Oklahoma
Kansas
Oregon
Pennsylvania
Ohio
Washington
Connecticut
New Jersey
Massachusetts
Rhode Island
Hawaii
West Virginia
Vermont
South Dakota
North Dakota
Montana
Maine
Idaho
New Hampshire
Iowa
Nebraska
Minnesota
Wisconsin
Dim2 (24.7%)
Dim1 (62%)
cluster 1 2 3 4

Choosing the optimal number of clusters:

No single compelling way to choose clusters, but we will explore two types of methods.

- Direct methods that involve optimizing a particular criterion (elbow and silhouette methods)
- Testing methods that evaluate evidence against a null hypothesis (gap statistic)

Elbow method: A little squishy, but useful

As previously, let $W(C_k)$ be the within-cluster sum of squared distances between all pairs for cluster k for a particular clustering, and let

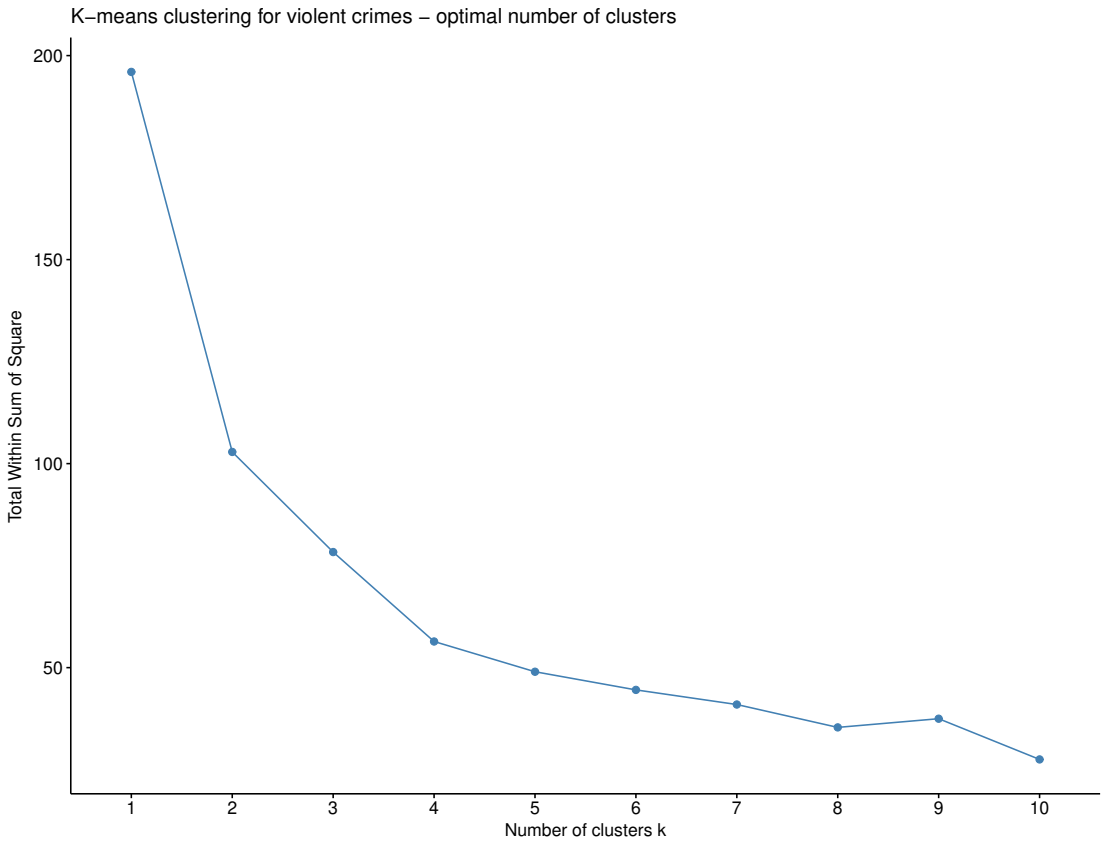
$$T_K = \sum_{k=1}^{K} W(C_k)$$

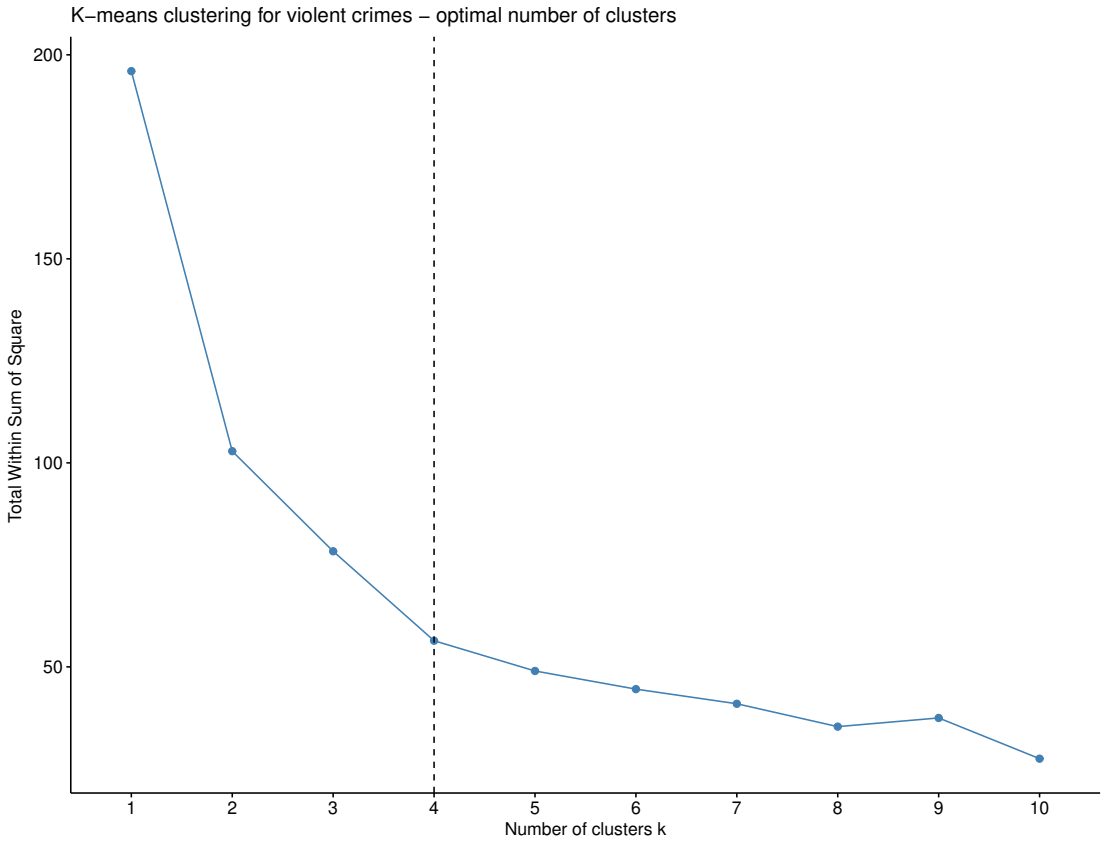
be the total within-cluster variation for the clustering with K clusters.

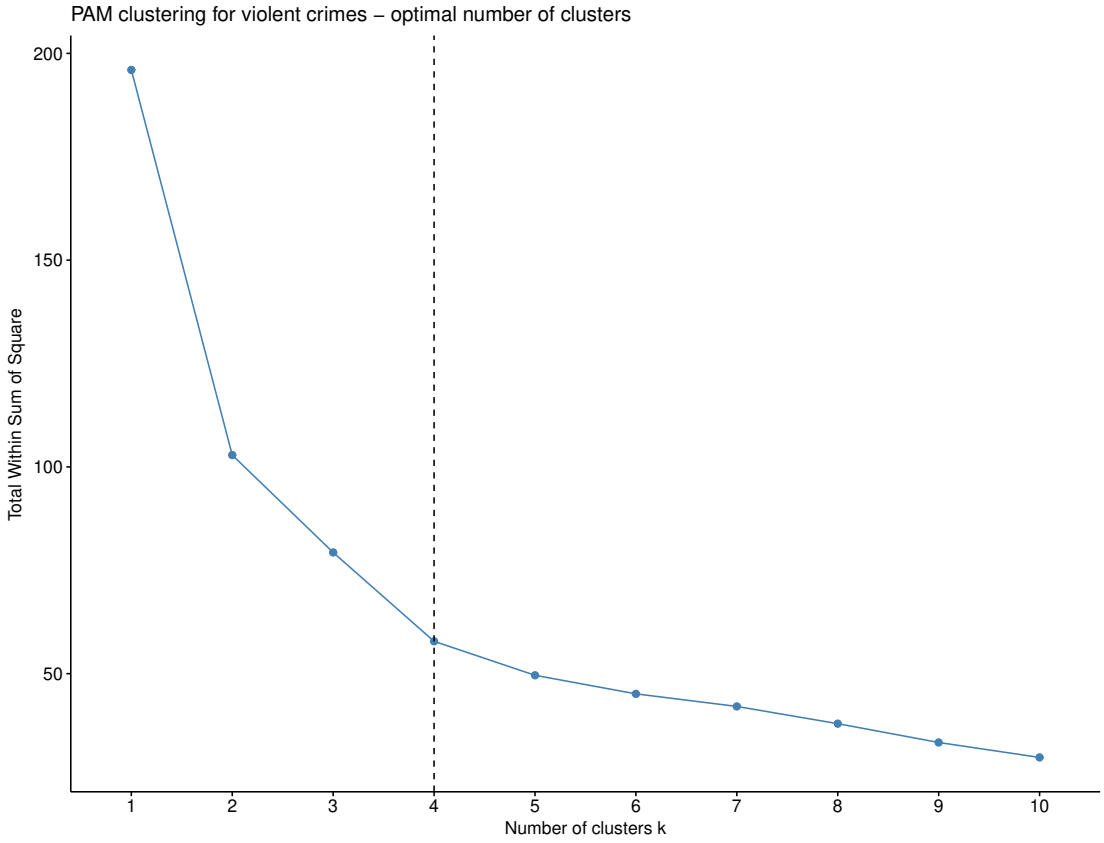
The method:

1. For the particular clustering method, let K vary over a range of values (say 1 to 10).
2. Compute T_K for each K.
3. Plot T_K against K, and look for a clear bend ("knee") in the graph.

The value of K where the bend occurs is considered the appropriate number of clusters.


K−means clustering for violent crimes − optimal number of clusters
Total Within Sum of Square
200
150
100
50
1
2
3
4
5
6
7
8
9
10
Number of clusters k


K−means clustering for violent crimes − optimal number of clusters
Total Within Sum of Square
200
150
100
50
1
2
3
4
5
6
7
8
9
10
Number of clusters k



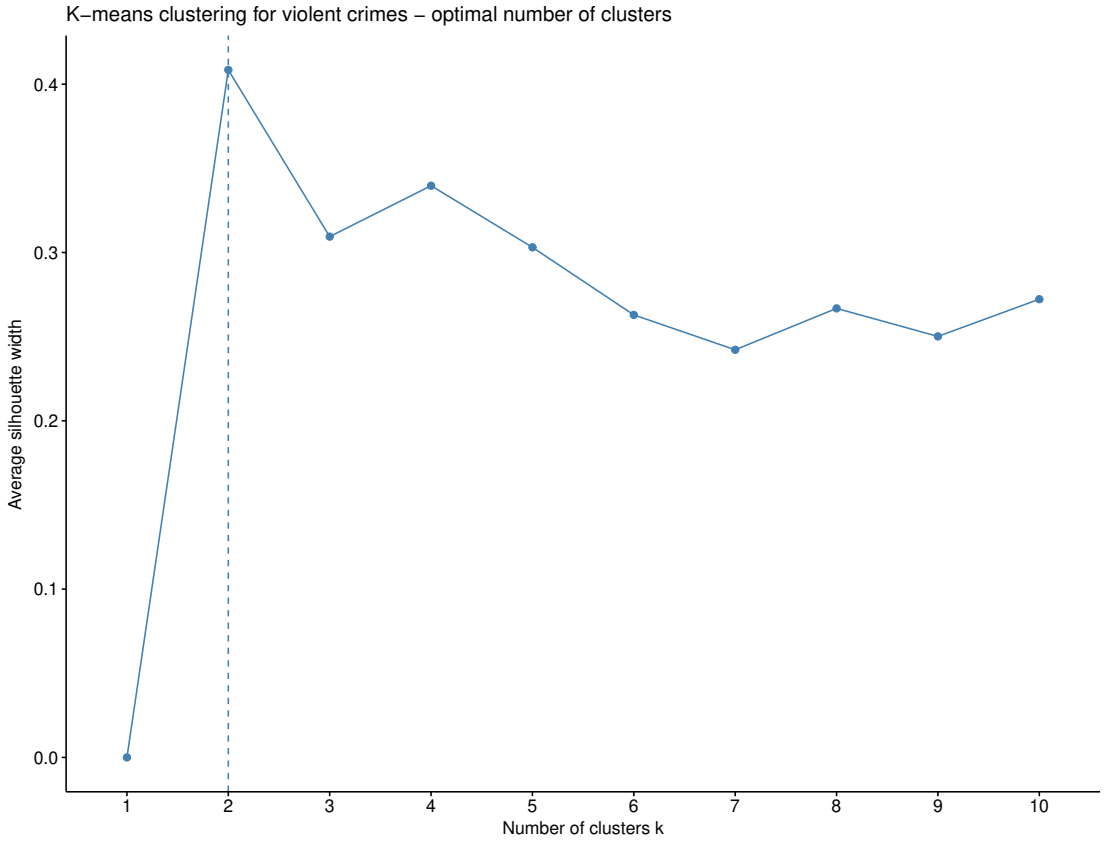
Average silhouette method: Similar construction to the elbow method

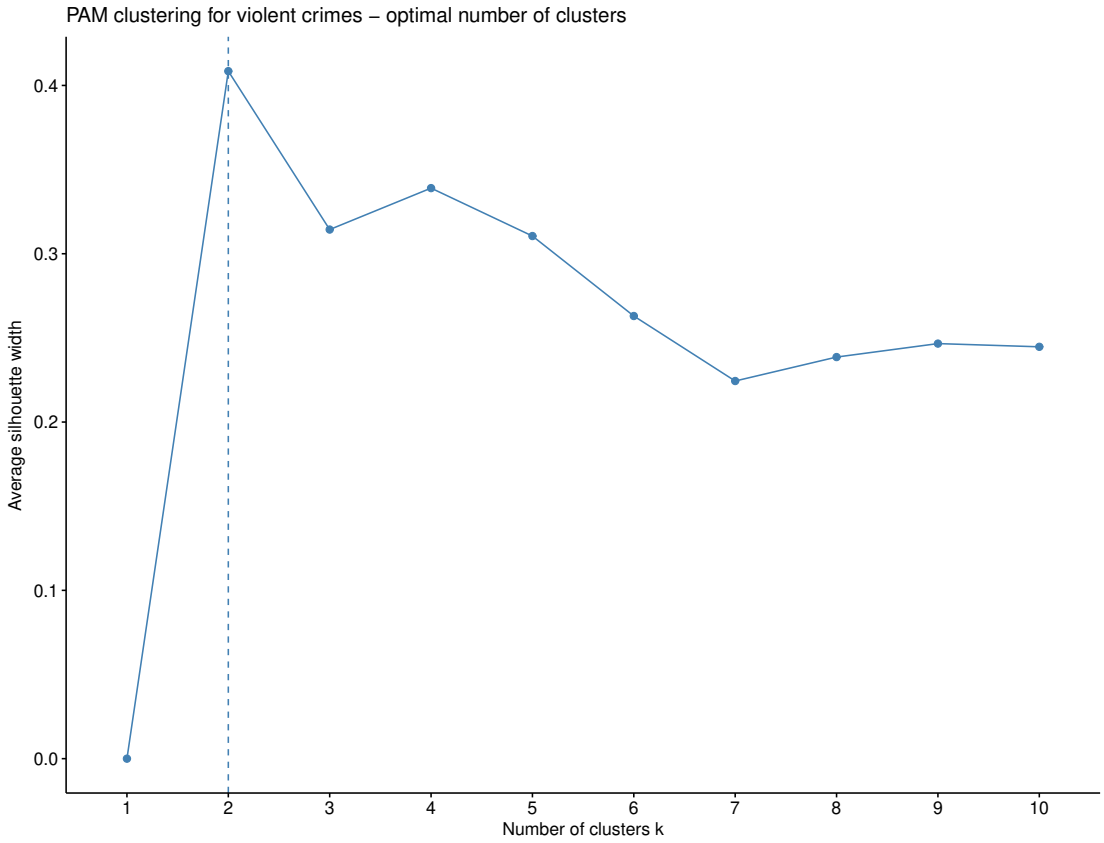
1. Compute clustering algorithm for different numbers of clusters K, varying K from (say) 1 to 10.
2. For each K, calculate the average silhouette

$$S_K = \frac{1}{n}\sum_{i=1}^{n} s_i$$

   across all observations.
3. Plot S_K against K.

The value of K where S_K is maximized is considered the appropriate number of clusters.


K−means clustering for violent crimes − optimal number of clusters
Average silhouette width
0.0
0.1
0.2
0.3
0.4
1
2
3
4
5
6
7
8
9
10
Number of clusters k


PAM clustering for violent crimes − optimal number of clusters
Average silhouette width
0.0
0.1
0.2
0.3
0.4
1
2
3
4
5
6
7
8
9
10
Number of clusters k

Comments:

- Elbow method suggests four clusters for both K-means and PAM; average silhouette method suggests two clusters
- Computation for each method are different enough that inconsistent results are not uncommon
- Both approaches measure global clustering characteristics only, and are informal (and somewhat ad hoc) approaches

Gap statistic:

Due to Tibshirani et al., (JRSS-B, 2001).

Idea: For a particular choice of K clusters, compare the total within cluster variation to the expected within-cluster variation under the assumption that the data have no obvious clustering (i.e., randomly distributed).

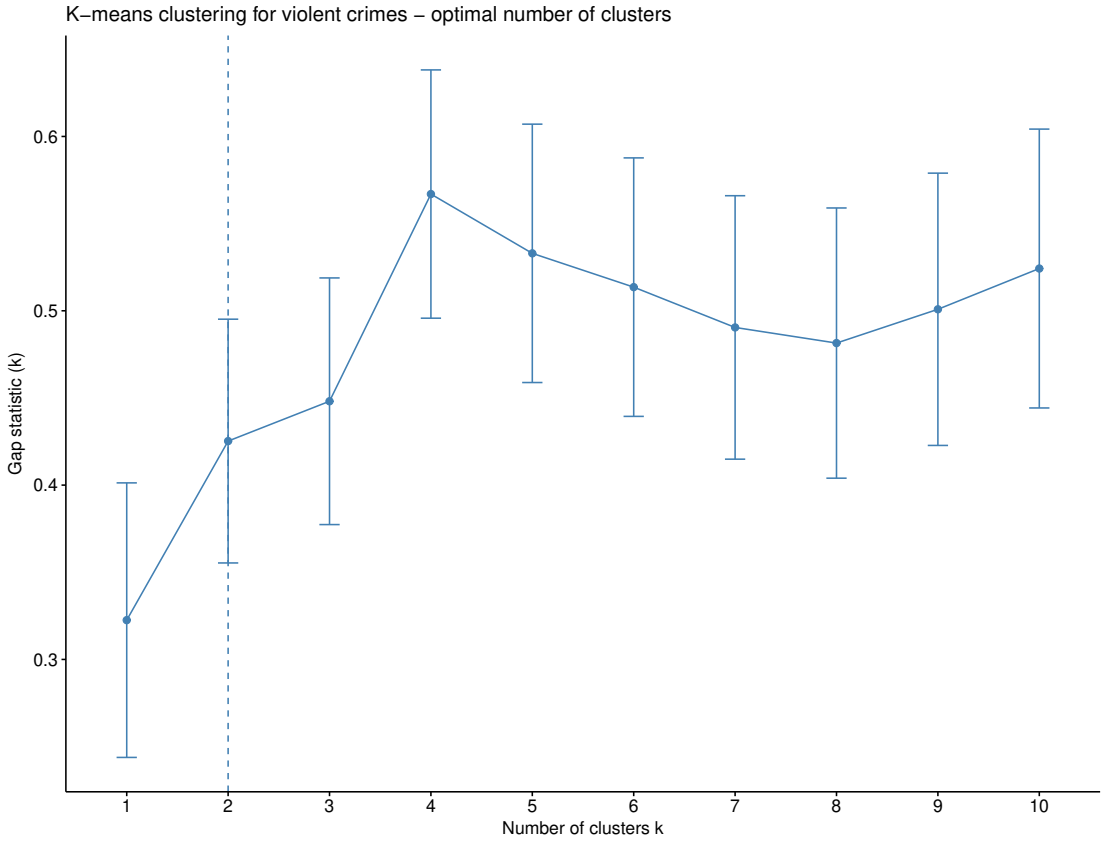
The gap statistic in essence detects whether the data clustered into K groups is significantly better than if they were generated at random.

Algorithm for computing Gap statistic:

1. Cluster the data at varying number of total clusters K, say from 1 to 10. Let T_K be the total within-cluster sum of squared distances.
2. Generate B reference data sets of size n, with the simulated values of variable j uniformly generated over the range of the observed variable $\boldsymbol{x}_j$. Typically $B = 500$.
3. For each generated data set $b = 1, \ldots, B$, perform the clustering for each K (from 1 to 10, say). Compute the total within-cluster sum of squared distances $T_k^{(b)}$.
4. Compute the Gap statistic
$$\text{Gap}(K) = \left(\frac{1}{B} \sum_{b=1}^{B} \log(T_K^{(b)}) \right) - \log(T_K).$$
5. Let $\bar{w} = \frac{1}{B} \sum_{b=1}^{B} \log(T_K^{(b)})$. Compute the standard deviation
$$\text{sd}(K) = \sqrt{\frac{1}{B} \sum_{b=1}^{B} \left(\log(T_K^{(b)}) - \bar{w} \right)^2}.$$
Define $s_K = \text{sd}(K)\sqrt{1 + 1/B}$.
6. Finally, choose the number of clusters as the smallest K such that
$$\text{Gap}(K) \geq \text{Gap}(K+1) - s_{K+1}.$$

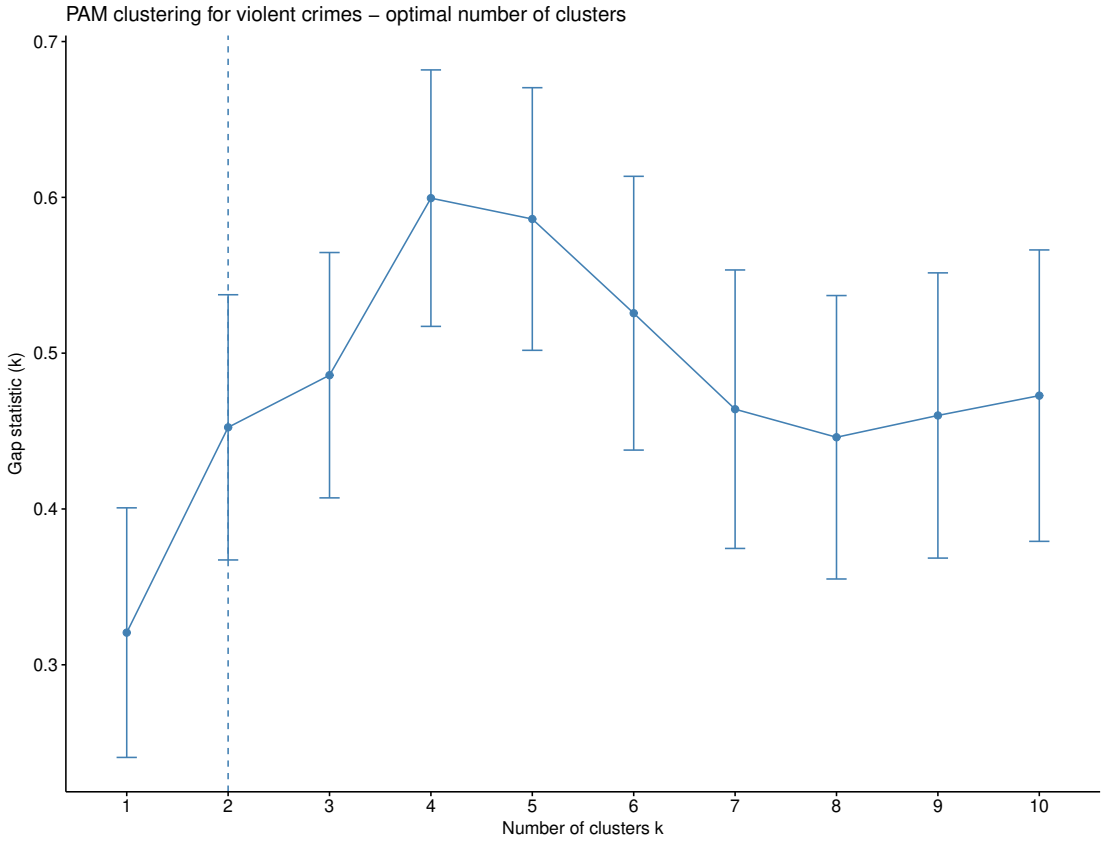
R computation for Gap statistic (K-means): Use `d.power=2` in `clusGap`.

```
> gapstat = clusGap(scale(USArrests),FUN=kmeans,nstart=25,
   d.power=2,K.max=10,B=500)
> print(gapstat, method="Tibs2001SEmax")
Clustering Gap statistic ["clusGap"] from call:
clusGap(x = scale(USArrests), FUNcluster = kmeans, K.max = 10,
B = 500, d.power = 2, nstart = 25)
B=500 simulated reference sets, k = 1..10; spaceH0="scaledPCA"
 --> Number of clusters (method 'Tibs2001SEmax', SE.factor=1): 2
          logW   E.logW       gap     SE.sim
 [1,] 4.584967 4.907504 0.3225366 0.07874126
 [2,] 3.940245 4.365505 0.4252604 0.06991504
 [3,] 3.667698 4.115800 0.4481020 0.07074145
 [4,] 3.339378 3.906337 0.5669584 0.07121078
 [5,] 3.197534 3.730479 0.5329454 0.07410821
 [6,] 3.064162 3.577725 0.5135632 0.07414276
 [7,] 2.951196 3.441636 0.4904401 0.07556430
 [8,] 2.836404 3.317880 0.4814760 0.07750226
 [9,] 2.703637 3.204488 0.5008511 0.07809635
[10,] 2.574965 3.099201 0.5242364 0.07998386
```



R computation for Gap statistic (PAM):

```
> gapstat = clusGap(scale(USArrests),FUN=pam,d.power=2,K.max=10,B=500)
> print(gapstat, method="Tibs2001SEmax")
Clustering Gap statistic ["clusGap"] from call:
clusGap(x = scale(USArrests), FUNcluster = pam, K.max = 10, B = 500,
  d.power = 2)
B=500 simulated reference sets, k = 1..10; spaceH0="scaledPCA"
 --> Number of clusters (method 'Tibs2001SEmax', SE.factor=1): 2
          logW   E.logW       gap     SE.sim
 [1,] 4.584967 4.905601 0.3206337 0.08009264
 [2,] 3.940245 4.392629 0.4523840 0.08510875
 [3,] 3.680220 4.166071 0.4858506 0.07873542
 [4,] 3.364196 3.963686 0.5994908 0.08230313
 [5,] 3.210919 3.797018 0.5860995 0.08428397
 [6,] 3.116075 3.641724 0.5256496 0.08785362
 [7,] 3.046182 3.510214 0.4640323 0.08936437
 [8,] 2.942032 3.388051 0.4460185 0.09095425
 [9,] 2.814149 3.274152 0.4600037 0.09152990
[10,] 2.699437 3.172174 0.4727366 0.09351808
```



Comments:

- Both K-means and PAM clustering are optimized for $K = 2$ based on the Gap statistic.
- This is a more principled approach to choosing cluster sizes, though clearly different conclusions can be reached based on different clustering approaches.
- Lots of measures beyond the ones presented for determining the best number of clusters. How can we choose which measure to use?

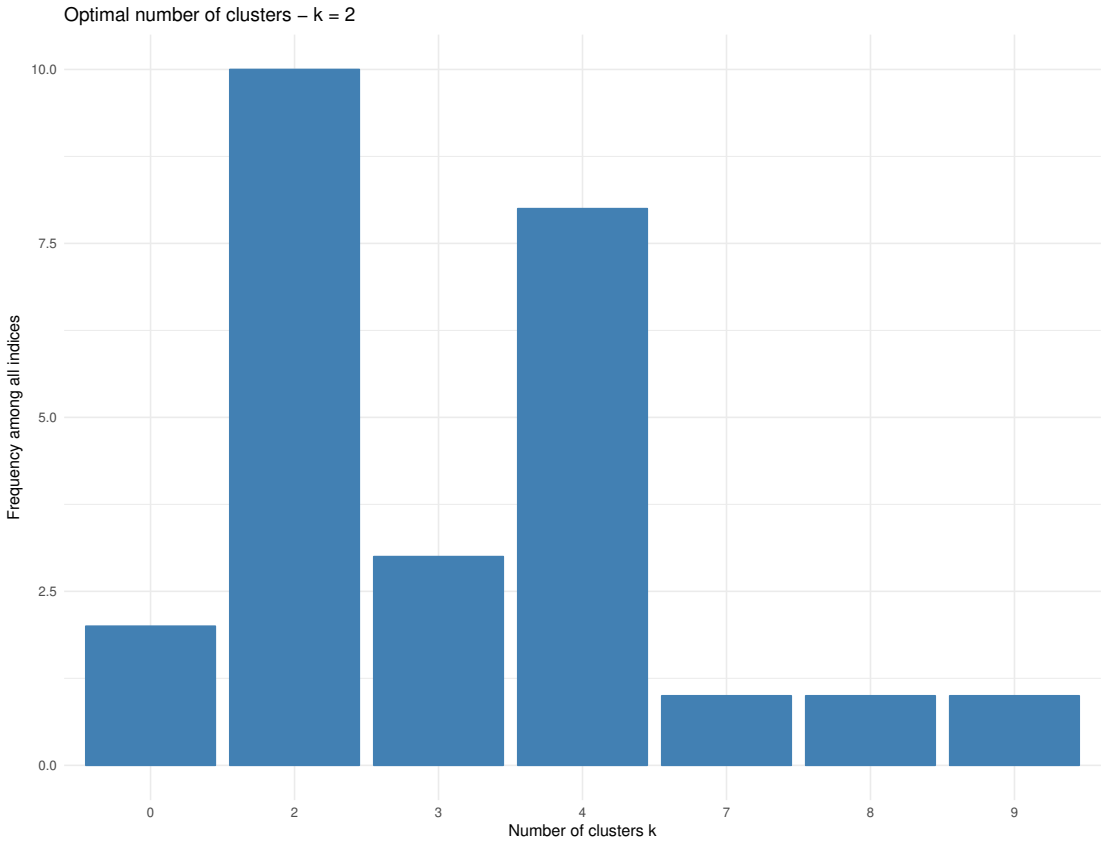
Why bother choosing: Can summarize the distribution of cluster sizes across measures!

The `NbClust` function in the `NbClust` R library offers 30 different commonly used measures.

Can plot histogram of the optimal number of clusters across the different measures.

For the violent crimes data,

- Seems that $K = 2$ is the majority choice
- $K = 4$ is a strong second place finisher



Introduction to soft clustering:

Basic idea: Rather than assigning each observation to a distinct cluster, determine probabilities or weights for each observation being a member of a cluster.

We will take a quick look at

- Fuzzy clustering
- Model-based clustering

Fuzzy clustering: FANNY (in the `cluster` R library)

FANNY = Fuzzy Analysis Clustering

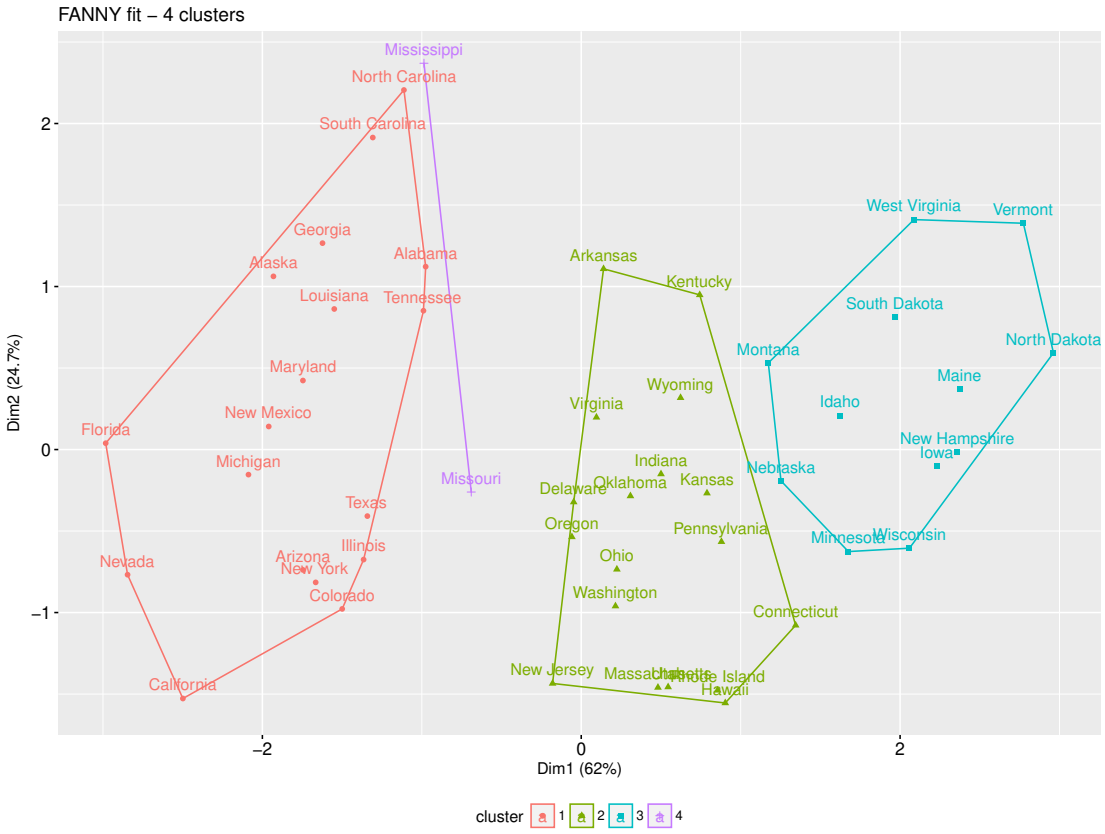
1. Specify the number of clusters K
2. Determine the membership weights u_{ik} for observation i belonging to cluster $k = 1, \ldots, K$ (where $\sum_{k=1}^{K} u_{ik} = 1$) that minimize

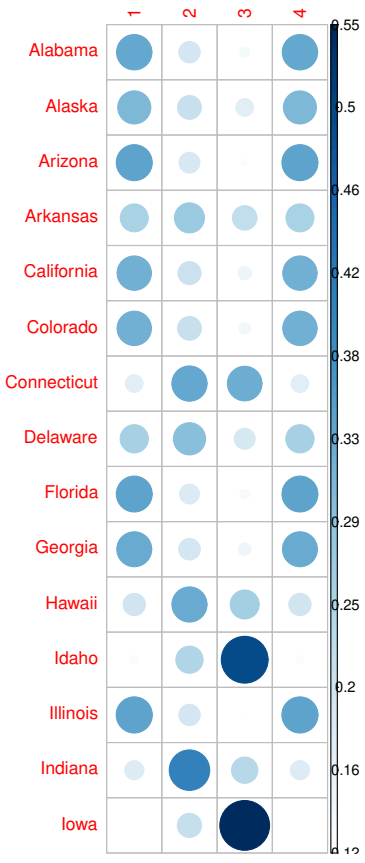
$$T_F(K) = \sum_{k=1}^{K} \left(\frac{\sum_{i,i'} u_{ik}^r u_{i'k}^r d_{ii'}}{2 \sum_{i=1}^{n} u_{ik}^r} \right)$$

where $d_{ii'}$ is the dissimilarity between observations i and i', and the given exponent r (a recommended choice is $r = 2$) is the "fuzzifier."

FANNY analysis:

```
> arrests.fanny = fanny(scale(USArrests),k=4)
> head(round(arrests.fanny$membership,3),15)
             [,1]  [,2]  [,3]  [,4]
Alabama     0.335 0.196 0.134 0.335
Alaska      0.307 0.215 0.172 0.307
Arizona     0.342 0.192 0.124 0.342
Arkansas    0.253 0.274 0.221 0.253
California  0.322 0.208 0.147 0.322
Colorado    0.322 0.215 0.140 0.322
Connecticut 0.171 0.333 0.326 0.171
Delaware    0.256 0.297 0.192 0.256
Florida     0.341 0.185 0.133 0.341
Georgia     0.329 0.197 0.144 0.329
Hawaii      0.203 0.332 0.263 0.203
Idaho       0.128 0.245 0.498 0.128
Illinois    0.344 0.195 0.116 0.344
Indiana     0.180 0.402 0.237 0.180
Iowa        0.117 0.218 0.548 0.117
```


FANNY fit – 4 clusters
Mississippi
North Carolina
South Carolina
Georgia
Alaska
Alabama
Louisiana
Tennessee
Maryland
New Mexico
Florida
Michigan
Missouri
Texas
Illinois
Arizona
New York
Nevada
Colorado
California
Arkansas
Kentucky
Virginia
Wyoming
Indiana
Delaware
Oklahoma
Kansas
Oregon
Pennsylvania
Ohio
Washington
Connecticut
New Jersey
Massachusetts
Rhode Island
Hawaii
West Virginia
Vermont
South Dakota
North Dakota
Montana
Maine
Idaho
New Hampshire
Iowa
Nebraska
Minnesota
Wisconsin
Dim2 (24.7%)
Dim1 (62%)
cluster 1 2 3 4


1
2
3
4
Alabama
Alaska
Arizona
Arkansas
California
Colorado
Connecticut
Delaware
Florida
Georgia
Hawaii
Idaho
Illinois
Indiana
Iowa
0.55
0.5
0.46
0.42
0.38
0.33
0.29
0.25
0.2
0.16

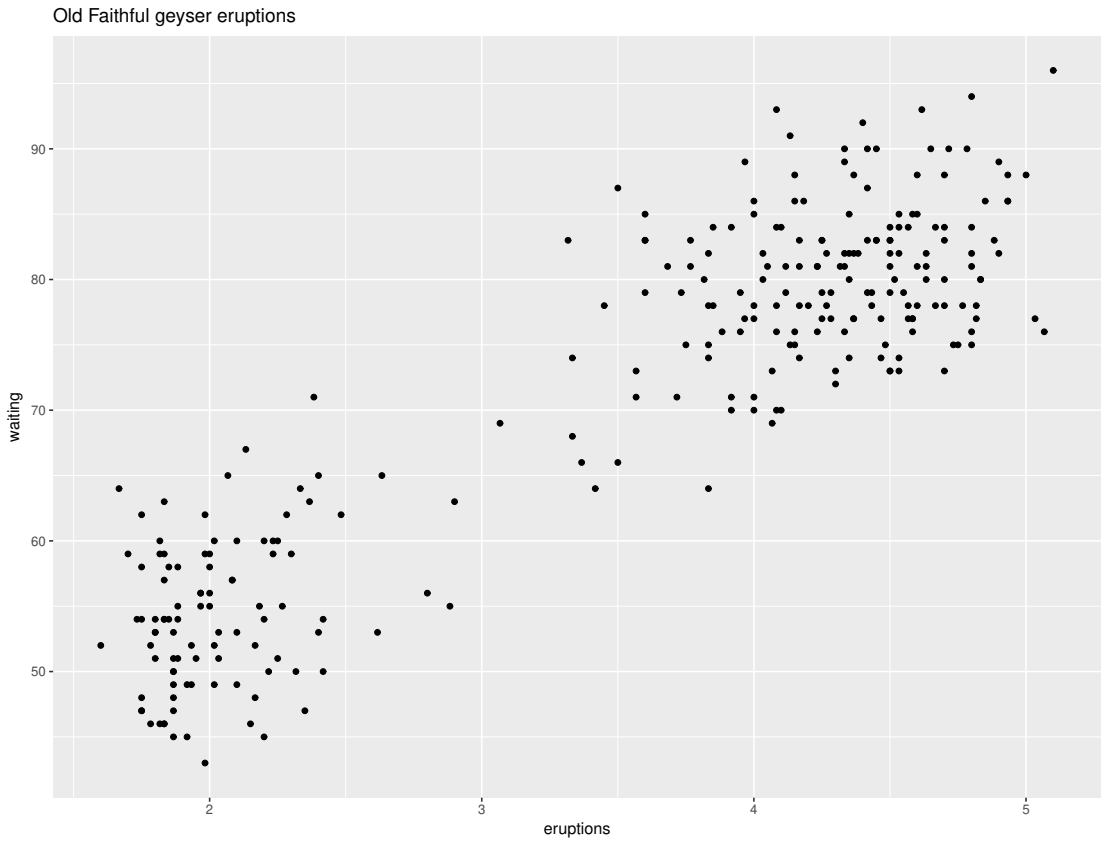
Model-based clustering: Mixture models

- This approach to clustering relies on probability modeling
- Assume the number of clusters (mixture components) is known to be K.
- A common approach is to assume the data come from a mixture of K multivariate normal distributions with mean vector $\boldsymbol{\mu}_k$ and covariance matrix $\boldsymbol{\Sigma}_k$ for mixture component k, with mixture weights w_k.
- The probability density for an observation is given by

$$f(\boldsymbol{x}) = w_1\varphi(\boldsymbol{x}|\boldsymbol{\mu}_1, \boldsymbol{\Sigma}_1) + \cdots + w_K\varphi(\boldsymbol{x}|\boldsymbol{\mu}_K, \boldsymbol{\Sigma}_K).$$

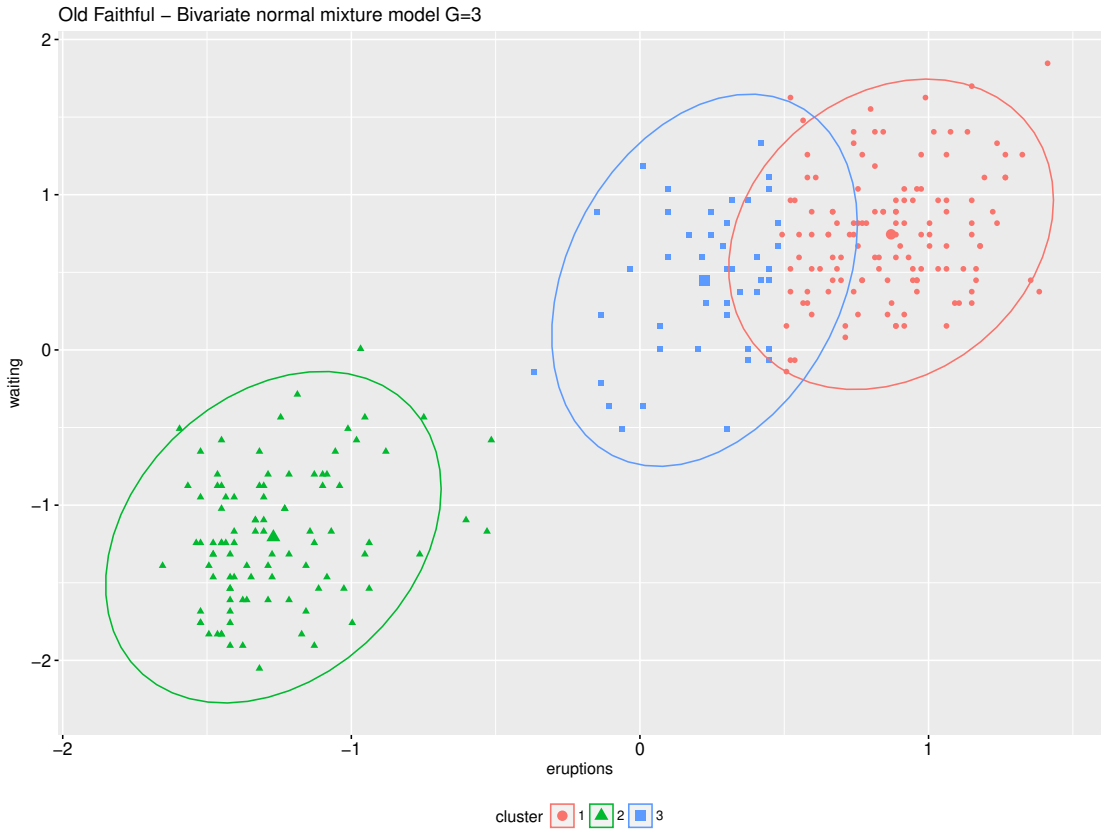
Computational issues:

- The model parameters (w_k, $\boldsymbol{\mu}_k$, $\boldsymbol{\Sigma}_k$) can be estimated using the EM algorithm, which is an iterative optimization algorithm developed by faculty in the Harvard Statistics department.
- Can initialize mixture component assignments using a partitioning or hierarchical clustering algorithm.
- Need to specify assumptions about the covariance matrices: volume, shape and orientation.
- Can select optimum K through the BIC statistic (or other information-based measures).



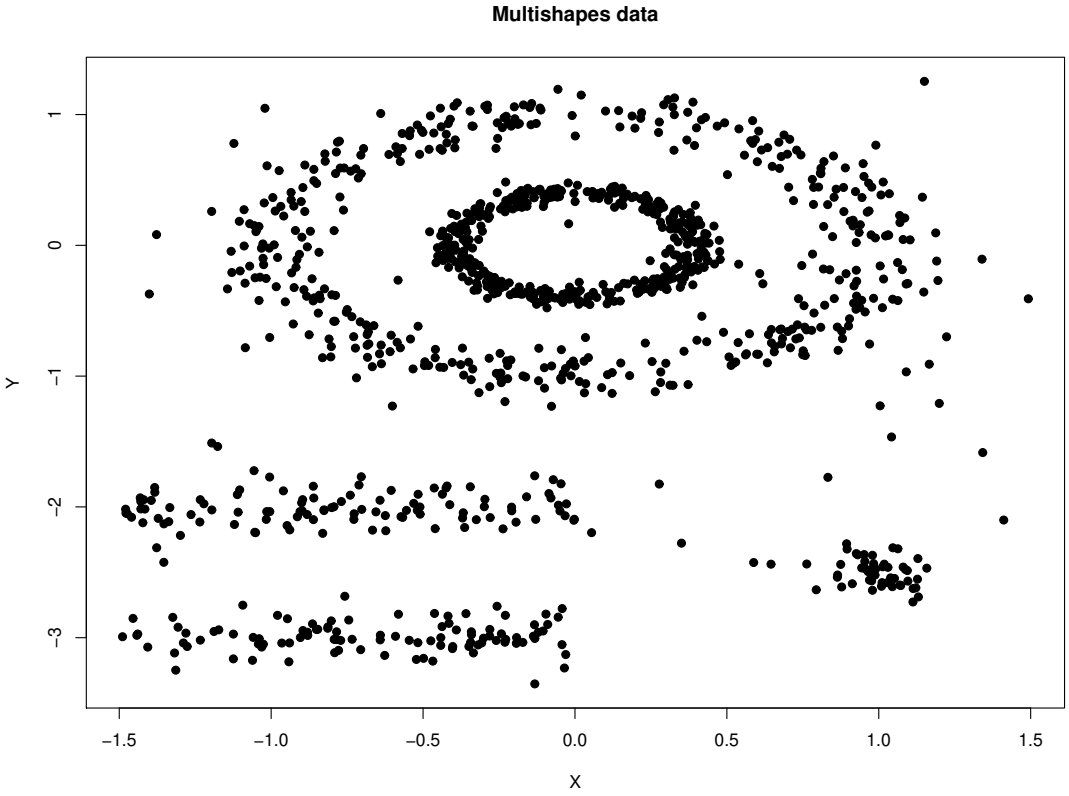
Example: Old Faithful geyser eruptions (duration, time between eruptions)

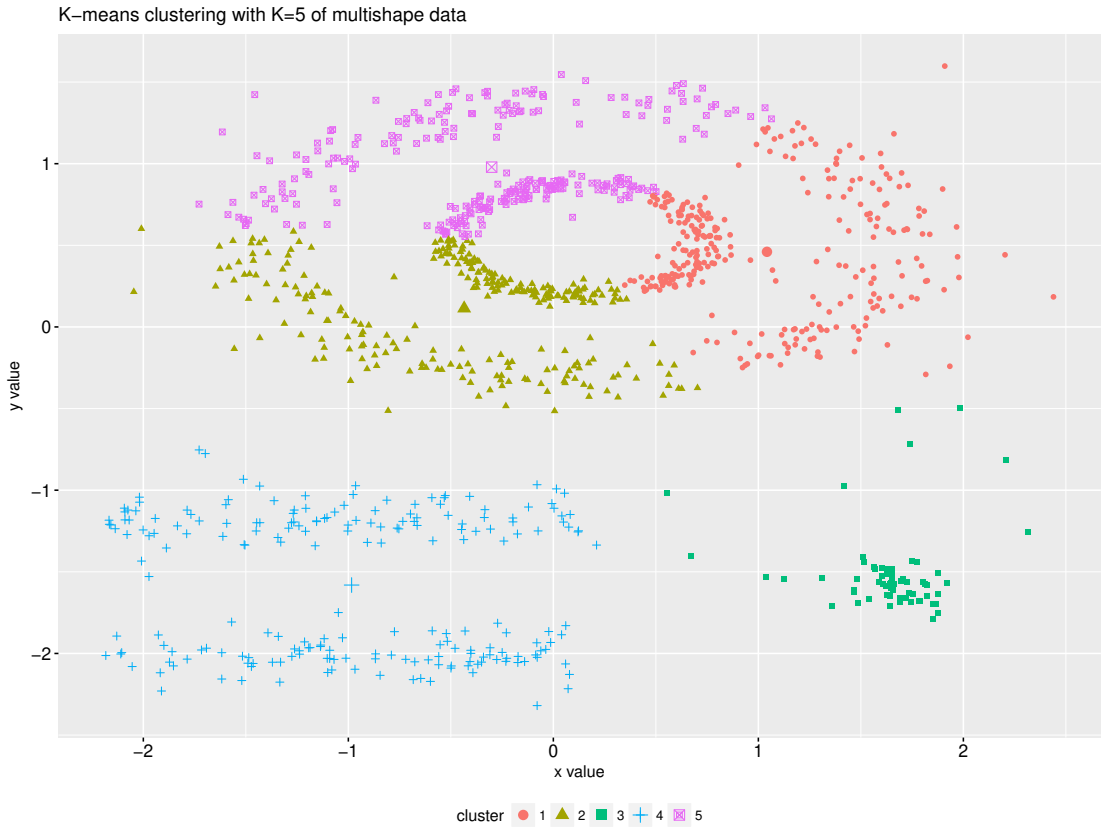
```
> library(mclust)
> faithful.mc = Mclust(faithful)
> summary(faithful.mc)
----------------------------------------------------
Gaussian finite mixture model fitted by EM algorithm
----------------------------------------------------

Mclust EEE (ellipsoidal, equal volume, shape and orientation)
model with 3 components:

 log.likelihood   n df       BIC       ICL
      -1126.361 272 11 -2314.386 -2360.865

Clustering table:
  1   2   3
130  97  45

> # optimal number of clusters
> faithful.mc$G
[1] 3


> # estimated probability for an observation to be
   in each cluster
> print(head(round(faithful.mc$z,3)))
   [,1]  [,2]  [,3]
1 0.022 0.000 0.978
2 0.000 1.000 0.000
3 0.003 0.000 0.997
4 0.000 1.000 0.000
5 0.984 0.000 0.016
6 0.000 0.998 0.002
```



One final clustering algorithm:

Imagine trying to cluster the following set of observations.


Multishapes data
Y
X
1
0
−1
−2
−3
−1.5
−1.0
−0.5
0.0
0.5
1.0
1.5


K−means clustering with K=5 of multishape data
y value
x value
1
0
−1
−2
−2
−1
0
1
2
cluster
1
2
3
4
5

DBSCAN: Density-based clustering algorithm (Ester et al., 1996, KDD-96)

Unlike previous clustering algorithms, DBSCAN

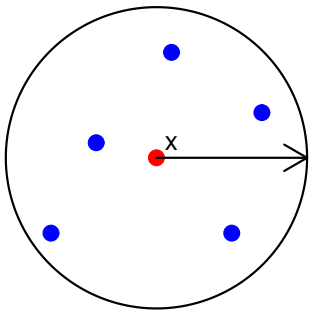
- can find any shape of clusters
- identifies observations that do not belong to clusters as outliers
- does not require specifying the number of clusters (like hierarchical clustering)
- can be used for predicting cluster membership for new data

DBSCAN algorithm: Identify dense regions of observations

Need to specify two parameters of the algorithm

1. ϵ: the radius of a neighborhood around an observation
2. `MinPts`: the minimum number of points within an ϵ radius of an observation to be considered a "core" point

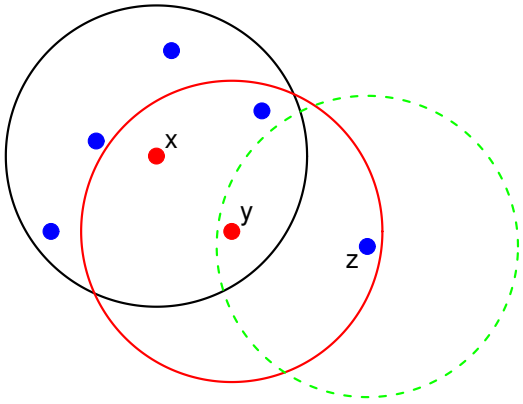
Example of core observation with `MinPts`$= 6$.



Three types of points:

- Core points - observations with `MinPts` total observations within an ϵ radius
- Border points - observations that are not core points, but are within ϵ of a core point
- Noise points - everything else

In the following, `x` is a core point, `y` is a border point, and `z` is a noise point.



Two terms:

- Density-reachable: Point A is density-reachable from point B if there is a set of core points leading from B to A.
- Density-connected: Two points A and B are density-connected if there is a core point C such that both A and B are density-reachable from C.

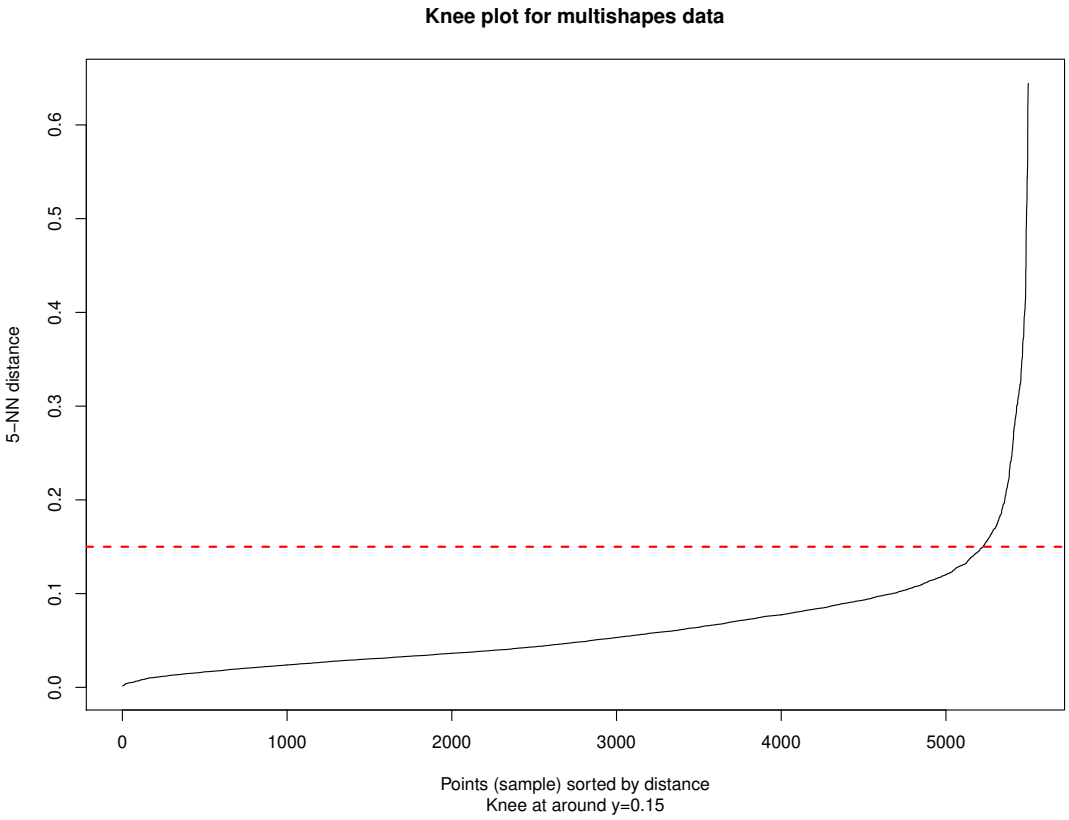
A density-based cluster is defined as a group of density-connected points.

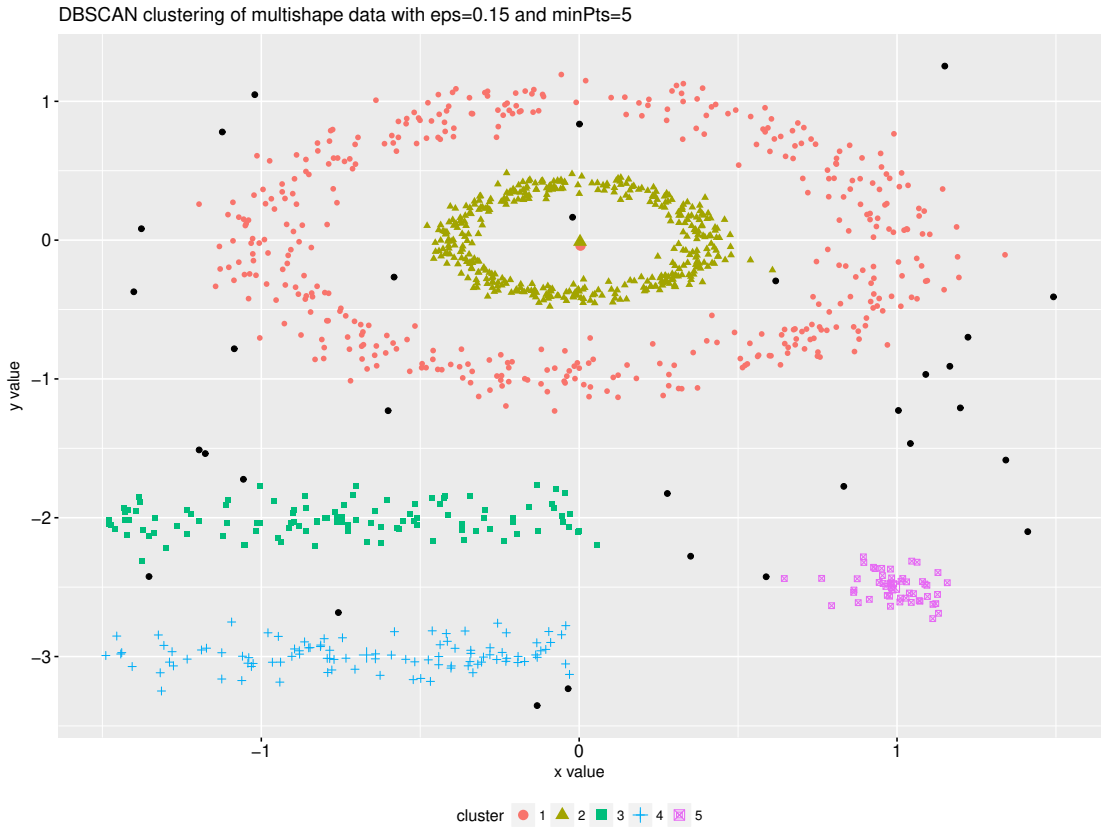
Choosing ϵ:

- Compute the k-th nearest neighbor distance for each point.
- Plot the distances in sorted order.

- Look for a bend (the "knee") in the plot, and use the distance at the knee as the choice of ϵ.

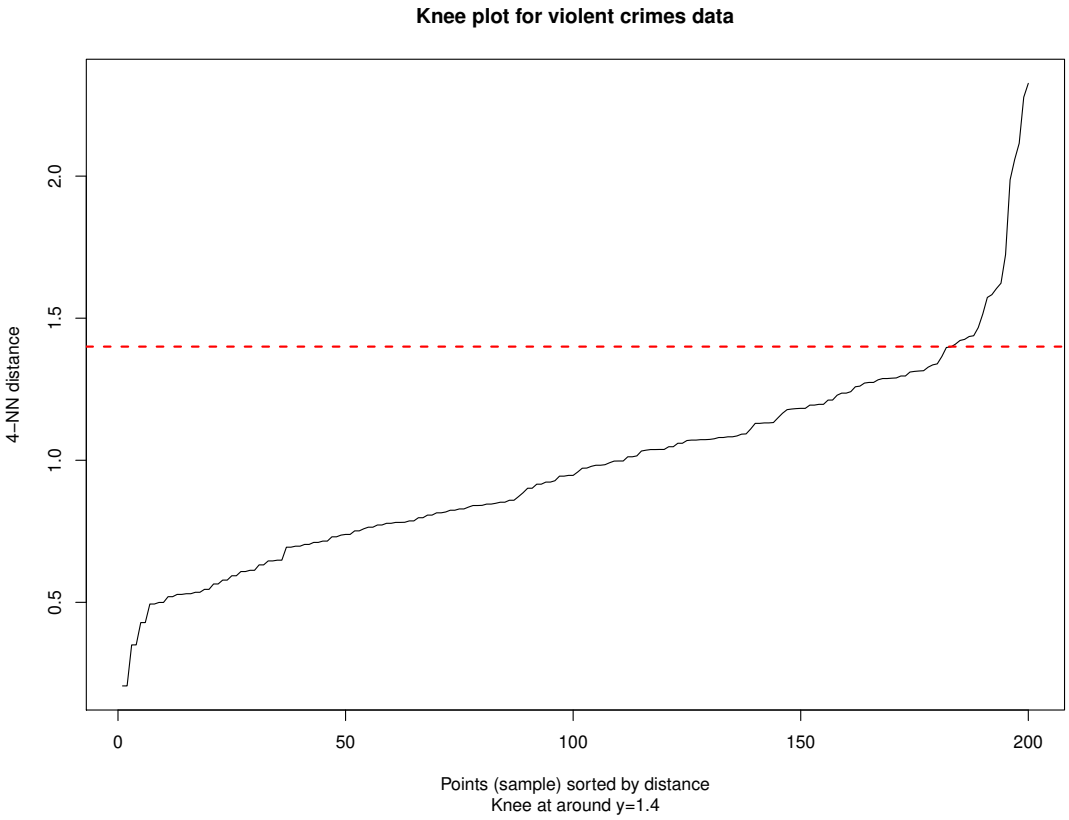
Refer to Ester et al. for the algorithm details, as well as the optimizing the choice of ϵ.

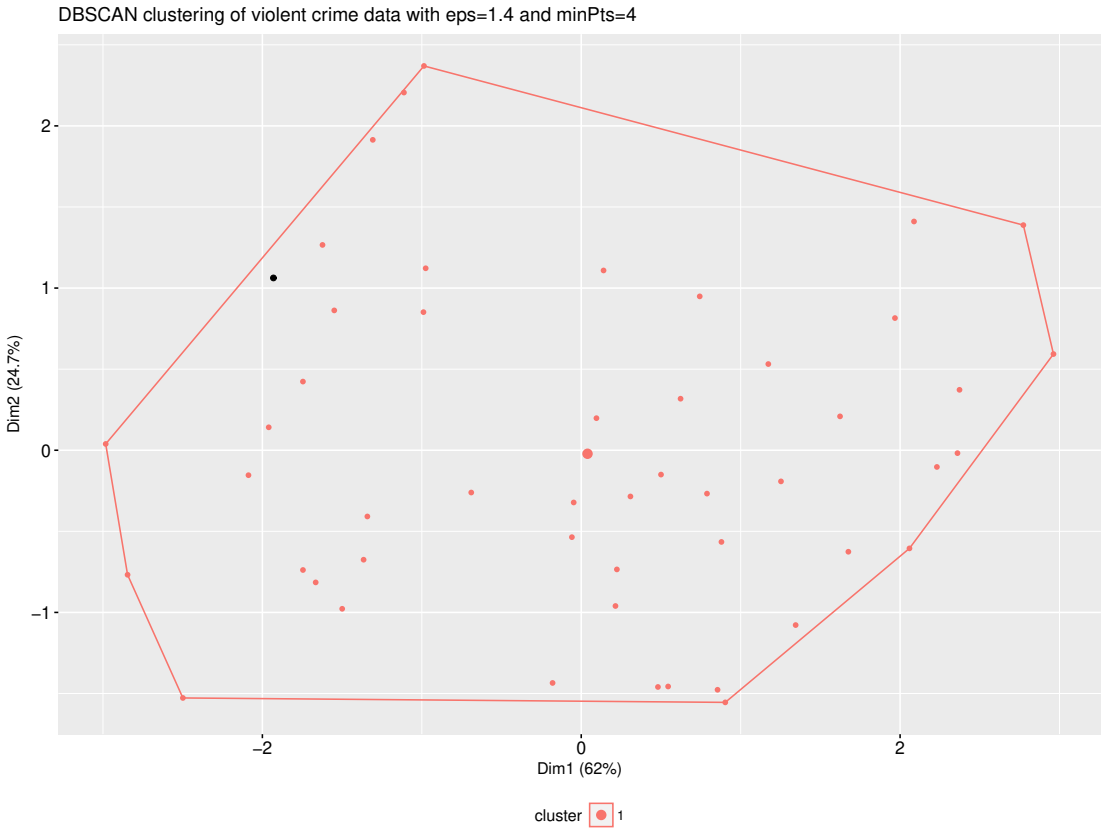

Knee plot for multishapes data
5−NN distance
0.0
0.1
0.2
0.3
0.4
0.5
0.6
0
1000
2000
3000
4000
5000
Points (sample) sorted by distance
Knee at around y=0.15


DBSCAN clustering of multishape data with eps=0.15 and minPts=5
y value
1
0
−1
−2
−3
−1
0
1
x value
cluster 1 2 3 4 5

Application to violent crime data:

- Chose $\epsilon = 1.4$ on standardized data based on knee plot, and `MinPts`$= 4$.
- Resulted in 1 density-based cluster, and 1 outlier.


Knee plot for violent crimes data
4−NN distance
0.5
1.0
1.5
2.0
0
50
100
150
200
Points (sample) sorted by distance
Knee at around y=1.4


DBSCAN clustering of violent crime data with eps=1.4 and minPts=4
Dim2 (24.7%)
2
1
0
−1
−2
0
2
Dim1 (62%)
cluster
1

Now for something really different: Chernoff faces

An unusual graphical method to cluster data:

- Each observation is a face
- Each variable is a facial feature
- Invented mostly as a joke!

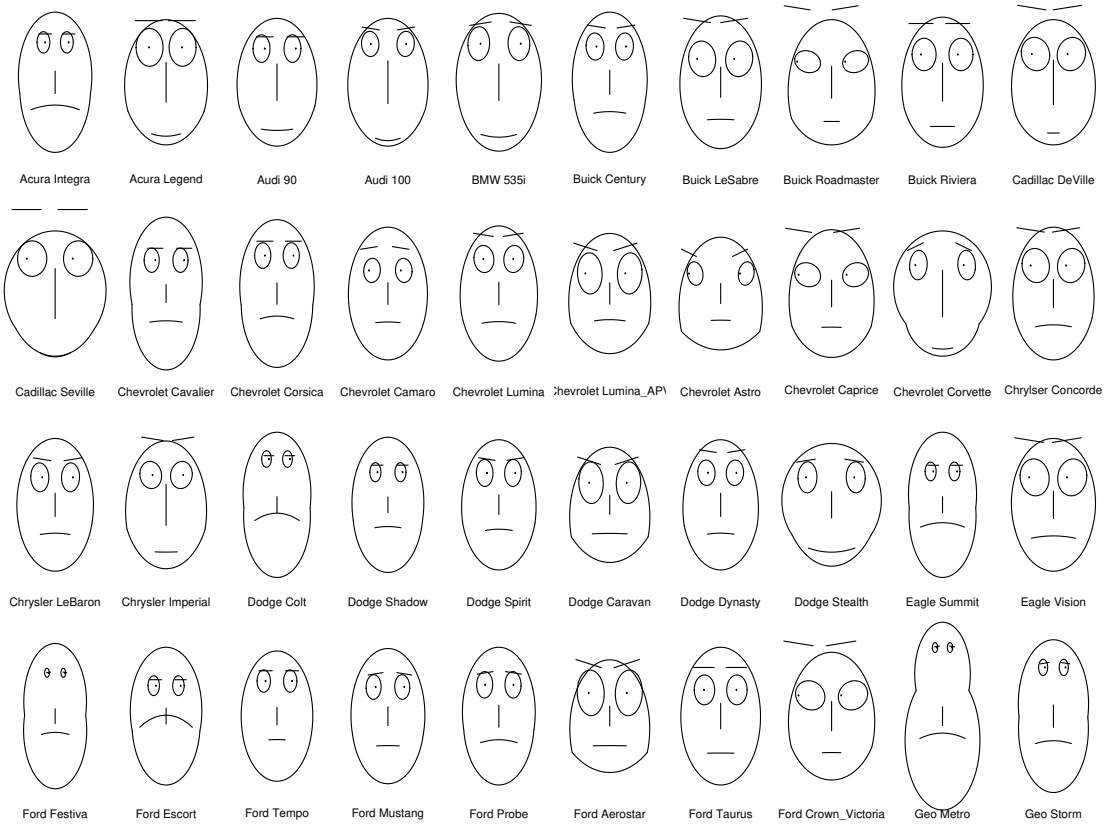
Car data: Data on model year 1993 cars

```
> summary(car.mat)
     Weight        MPG.city      MPG.highway      Horsepower      EngineSize
 Min.   :1695   Min.   :15.00   Min.   :20.00   Min.   : 55.0   Min.   :1.000
 1st Qu.:2620   1st Qu.:18.00   1st Qu.:26.00   1st Qu.:103.0   1st Qu.:1.800
 Median :3040   Median :21.00   Median :28.00   Median :140.0   Median :2.400
 Mean   :3073   Mean   :22.37   Mean   :29.09   Mean   :143.8   Mean   :2.668
 3rd Qu.:3525   3rd Qu.:25.00   3rd Qu.:31.00   3rd Qu.:170.0   3rd Qu.:3.300
 Max.   :4105   Max.   :46.00   Max.   :50.00   Max.   :300.0   Max.   :5.700
   Min.Price        Price         Max.Price         RPM        Rev.per.mile
 Min.   : 6.70   Min.   : 7.40   Min.   : 7.9   Min.   :3800   Min.   :1320
 1st Qu.:10.80   1st Qu.:12.20   1st Qu.:14.7   1st Qu.:4800   1st Qu.:1985
 Median :14.70   Median :17.70   Median :19.6   Median :5200   Median :2340
 Mean   :17.13   Mean   :19.51   Mean   :21.9   Mean   :5281   Mean   :2332
 3rd Qu.:20.30   3rd Qu.:23.30   3rd Qu.:25.3   3rd Qu.:5750   3rd Qu.:2565
 Max.   :45.40   Max.   :61.90   Max.   :80.0   Max.   :6500   Max.   :3755
 Fuel.tank.capacity   Passengers        Length        Wheelbase
 Min.   : 9.20      Min.   :2.000   Min.   :141.0   Min.   : 90.0
 1st Qu.:14.50      1st Qu.:4.000   1st Qu.:174.0   1st Qu.: 98.0
 Median :16.40      Median :5.000   Median :183.0   Median :103.0
 Mean   :16.66      Mean   :5.086   Mean   :183.2   Mean   :103.9
 3rd Qu.:18.80      3rd Qu.:6.000   3rd Qu.:192.0   3rd Qu.:110.0
 Max.   :27.00      Max.   :8.000   Max.   :219.0   Max.   :119.0
```

Example Chernoff faces: Car data

- Width of face – vehicle weight
- Height of face split – City MPG
- Length of face – Highway MPG
- Width of top half of face – Horsepower
- Width of bottom half of face – Engine size

Other facial features: Length of nose, curvature of mouth, size of eyes, angle of eyebrows, etc.


Acura Integra
Acura Legend
Audi 90
Audi 100
BMW 535i
Buick Century
Buick LeSabre
Buick Roadmaster
Buick Riviera
Cadillac DeVille
Cadillac Seville
Chevrolet Cavalier
Chevrolet Corsica
Chevrolet Camaro
Chevrolet Lumina
Chevrolet Lumina_APV
Chevrolet Astro
Chevrolet Caprice
Chevrolet Corvette
Chrylser Concorde
Chrysler LeBaron
Chrysler Imperial
Dodge Colt
Dodge Shadow
Dodge Spirit
Dodge Caravan
Dodge Dynasty
Dodge Stealth
Eagle Summit
Eagle Vision
Ford Festiva
Ford Escort
Ford Tempo
Ford Mustang
Ford Probe
Ford Aerostar
Ford Taurus
Ford Crown_Victoria
Geo Metro
Geo Storm