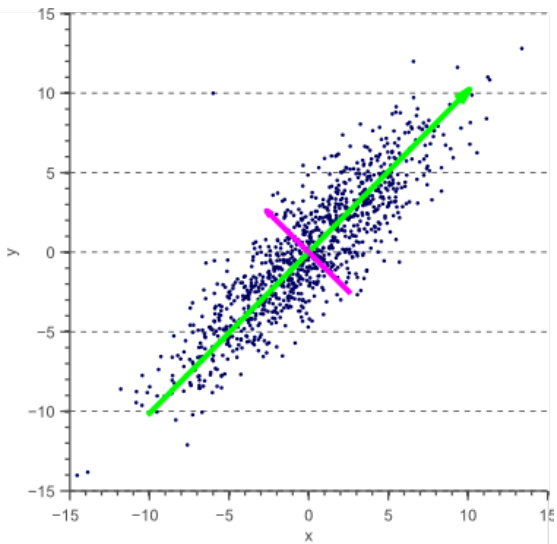
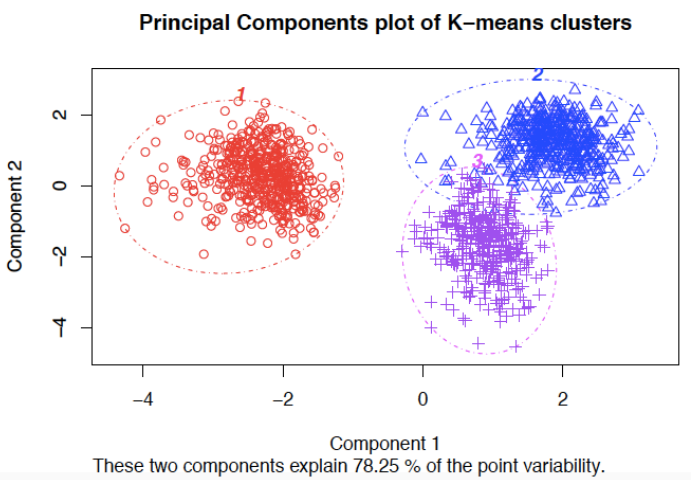
# CS 467, Introduction to Machine Learning

University of Southern California

M. R. Rajati, PhD

# Lesson 9
# Unsupervised Learning



# Unsupervised Learning

*Unsupervised vs Supervised Learning:*

- Most of this course focuses on *supervised learning* methods such as regression and classification.
- In that setting we observe both a set of features $X_1, X_2, \ldots, X_p$ for each object, as well as a response or outcome variable $Y$. The goal is then to predict $Y$ using $X_1, X_2, \ldots, X_p$.

# Unsupervised Learning

*Unsupervised vs Supervised Learning:*

- Here we instead focus on *unsupervised learning*, we where observe only the features $X_1, X_2, \ldots, X_p$.
- We are not interested in prediction, because we do not have an associated response variable $Y$ .

# The Goals of Unsupervised Learning

- The goal is to discover interesting things about the measurements: is there an informative way to visualize the data? Can we discover subgroups among the variables or among the observations?
- We discuss two methods:
  - *principal components analysis*, a tool used for data visualization or data pre-processing before supervised techniques are applied, and
  - *clustering*, a broad class of methods for discovering unknown subgroups in data.

# The Challenge of Unsupervised Learning

- Unsupervised learning is more subjective than supervised learning, as there is no simple goal for the analysis, such as prediction of a response.
- But techniques for unsupervised learning are of growing importance in a number of fields:
  - subgroups of breast cancer patients grouped by their gene expression measurements,
  - groups of shoppers characterized by their browsing and purchase histories,
  - movies grouped by the ratings assigned by movie viewers.

# Another advantage

- It is often easier to obtain *unlabeled data* — from a lab instrument or a computer — than *labeled data*, which can require human intervention.
- For example it is difficult to automatically assess the overall sentiment of a movie review: is it favorable or not?

# Clustering

- *Clustering* refers to a very broad set of techniques for finding *subgroups*, or *clusters*, in a data set.
- We seek a partition of the data into distinct groups so that the observations within each group are quite similar to each other.

Mutually exclusive
collectively exhaustive

# Clustering

- We must define what it means for two or more observations to be *similar* or *different*.
- Indeed, this is often a domain-specific consideration that must be made based on knowledge of the data being studied.

# Clustering for Market Segmentation

- Suppose we have access to a large number of measurements (e.g. median household income, occupation, distance from nearest urban area, and so forth) for a large number of people.

# Clustering for Market Segmentation

- Our goal is to perform *market segmentation* by identifying subgroups of people who might be more receptive to a particular form of advertising, or more likely to purchase a particular product.

# Clustering for Market Segmentation

- The task of performing market segmentation amounts to clustering the people in the data set.

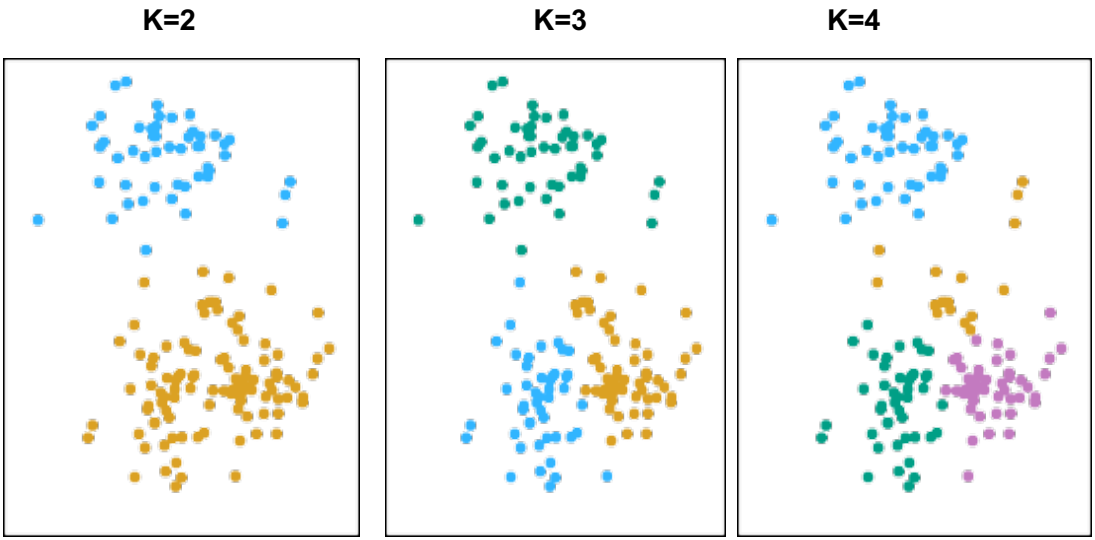
# Two clustering methods

- In *K-means clustering*, we seek to partition the observations into a pre-specified number of clusters.

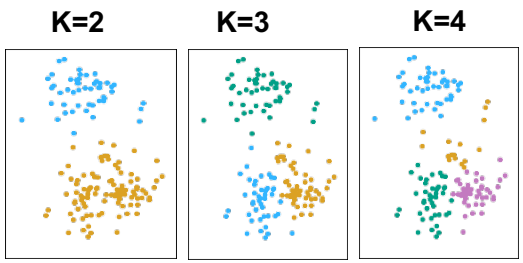
# Two clustering methods

- In *hierarchical clustering*, we do not know in advance how many clusters we want; in fact, we end up with a tree-like visual representation of the observations, called a *dendrogram*, that allows us to view at once the clusterings obtained for each possible number of clusters, from 1 to $n$.

# $K$-means clustering



# $K$-means clustering




A simulated data set with 150 observations in 2-dimensional space. Panels show the results of applying K-means clustering with different values of *K*, the number of clusters. The color of each observation indicates the cluster to which it was assigned using the K-means clustering algorithm. Note that there is no ordering of the clusters, so the cluster coloring is arbitrary. These cluster labels were not used in clustering; instead, they are the outputs of the clustering procedure.

We have a pre-specified # of clusters, K.

# Details of *K*-means clustering

Let $C_1, \ldots, C_K$ denote sets containing the indices of the observations in each cluster. These sets satisfy two properties:

Collectively Exhaustive

1. $C_1 \cup C_2 \cup \ldots \cup C_K = \{1, \ldots, n\}$. In other words, each observation belongs to at least one of the $K$ clusters.

Mutually exclusive

2. $C_k \cap C_{k'} = \emptyset$ for all distinct $k$ and $k'$. In other words, the clusters are non-overlapping: no observation belongs to more than one cluster.

$k \neq k'$

For instance, if the $i^{\text{th}}$ observation is in the $k^{\text{th}}$ cluster, then $i \in C_k$.

# Details of *K*-means clustering: continued

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

# Details of *K*-means clustering: continued

- The within-cluster variation for cluster $C_k$ is a measure WCV($C_k$) of the amount by which the observations within a cluster differ from each other.
- Hence we want to solve the problem

$$\underset{C_1,\ldots,C_K}{\text{minimize}} \left\{ \sum_{k=1}^{K} \text{WCV}(C_k) \right\}$$

# Details of *K*-means clustering: continued

- In words, this formula says that we want to partition the observations into *K* clusters such that the total within-cluster variation, summed over all *K* clusters, is as small as possible.

# How to define within-cluster variation?

- Typically we use Euclidean distance

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

where $|C_k|$ denotes the number of observations in the $k^{\text{th}}$ cluster.

# How to define within-cluster variation?

- Combining the above equations gives the optimization problem that defines K-means clustering.

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

NP Time

$\Rightarrow$ GREEDY

$\| x_i - x_{i'} \|_2^2$

# *K*-Means Clustering Algorithm

1. Randomly assign a number, from 1 to *K*, to each of the observations. These serve as initial cluster assignments for the observations.
2. Iterate until the cluster assignments stop changing:
   1. For each of the *K* clusters, compute the cluster *centroid*.
      The $k^{th}$ cluster centroid is the vector of the *p* feature means for the observations in the $k^{th}$ cluster.
   2. Assign each observation to the cluster whose centroid is closest (where *closest* is defined using Euclidean distance).

EM Algorithm

The Chicken & The Egg

# Properties of the Algorithm

- This algorithm is guaranteed to decrease the value of the objective function at each step. *Why?* Note that

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

$= 2\sum_{i\in C_k}\|x_i-\bar{x}_k\|_2^2$

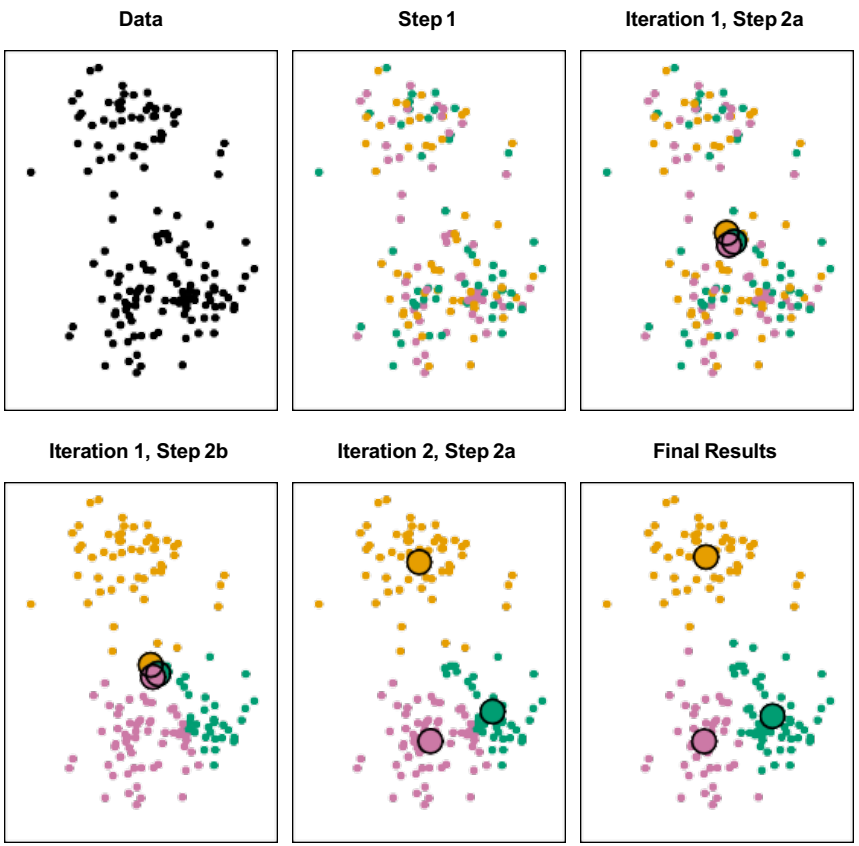
$\bar{x}_k$ → the center of the cluster

where $\bar{x}_{kj}=\frac{1}{|C_k|}\sum_{i\in C_k}x_{ij}$ is the mean for feature $j$ in cluster $C_k$.

- however it is not guaranteed to give the global minimum. *Why not?*

Repeat the algorithm using different initial conditions

# Example


Data
Step 1
Iteration 1, Step 2a
Iteration 1, Step 2b
Iteration 2, Step 2a
Final Results

# Details of Previous Figure

The progress of the K-means algorithm with $K=3$.

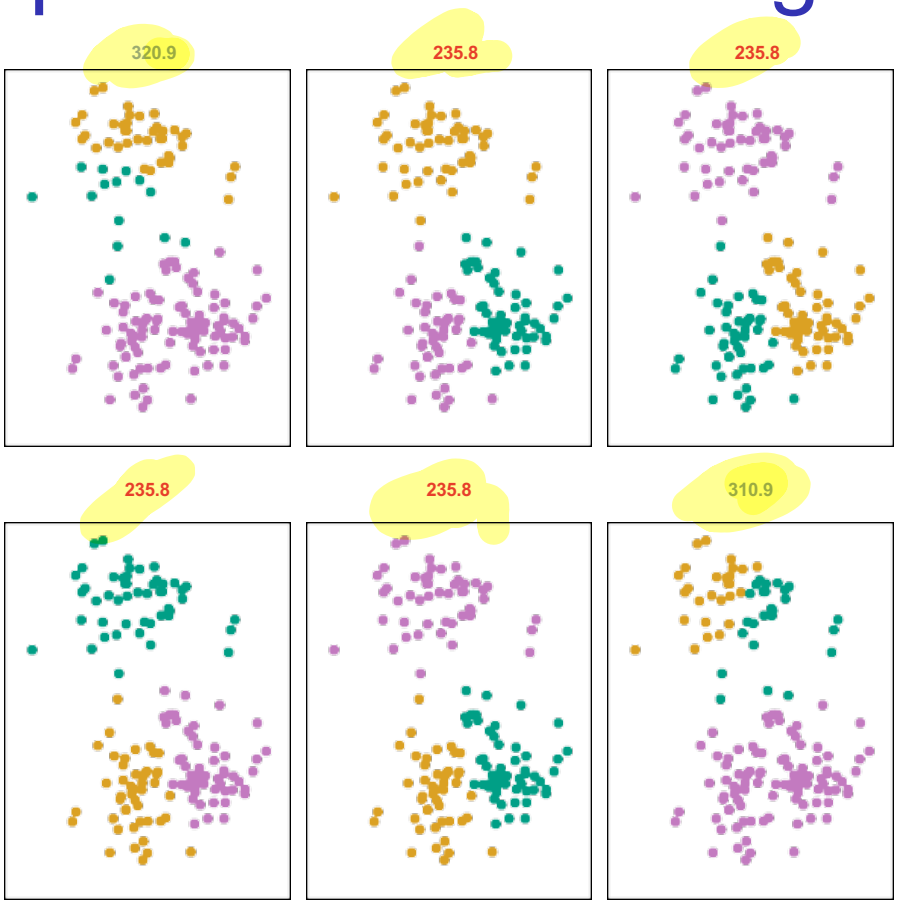
- *Top left:* The observations are shown.
- *Top center:* In Step 1 of the algorithm, each observation is randomly assigned to a cluster.
- *Top right:* In Step 2(a), the cluster centroids are computed. These are shown as large colored disks. Initially the centroids are almost completely overlapping because the initial cluster assignments were chosen at random.

# Details of Previous Figure

The progress of the K-means algorithm with $K=3$.

- *Bottom left:* In Step 2(b), each observation is assigned to the nearest centroid.
- *Bottom center:* Step 2(a) is once again performed, leading to new cluster centroids.
- *Bottom right:* The results obtained after 10 iterations.

# Example: different starting values



# Details of Previous Figure

*K*-means clustering performed six times on the data from previous figure with *K* = 3, each time with a different random assignment of the observations in Step 1 of the *K*-means algorithm.

# Details of Previous Figure

Above each plot is the value of the objective $\underset{C_1,\ldots,C_K}{\text{minimize}} \left\{ \sum_{k=1}^{K} \frac{1}{|C_k|} \sum_{i,i' \in C_k} \sum_{j=1}^{p} (x_{ij} - x_{i'j})^2 \right\}$

Three different local optima were obtained, one of which resulted in a smaller value of the objective and provides better separation between the clusters.

Those labeled in red all achieved the same best solution, with an objective value of 235.8

# K-Medoids Clustering

- Similar to K-means, but in each step we do not compute the centroid of each cluster.
- We compute the medoid, which is a data point that has the smallest average dissimilarity with other data points in the cluster.
- This way you limit the center of your cluster to be one of your data points.

# Hierarchical Clustering

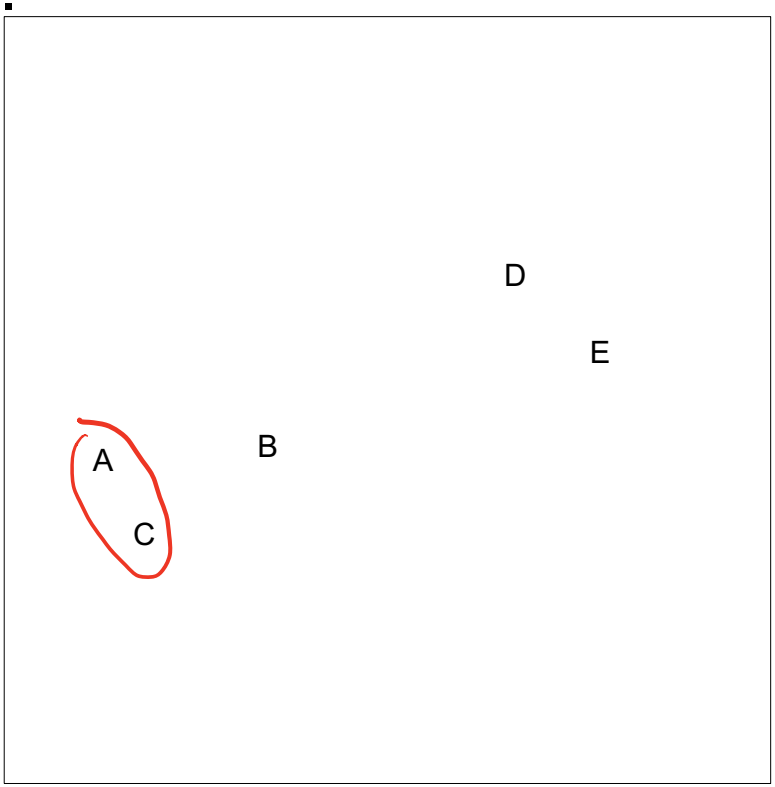
- *K*-means clustering requires us to pre-specify the number of clusters *K*. This can be a disadvantage (later we discuss strategies for choosing *K*)
- *Hierarchical clustering* is an alternative approach which does not require that we commit to a particular choice of *K*.

# Hierarchical Clustering

- In this section, we describe *bottom-up* or *agglomerative* clustering. This is the most common type of hierarchical clustering, and refers to the fact that a dendrogram is built starting from the leaves and combining clusters up to the trunk.

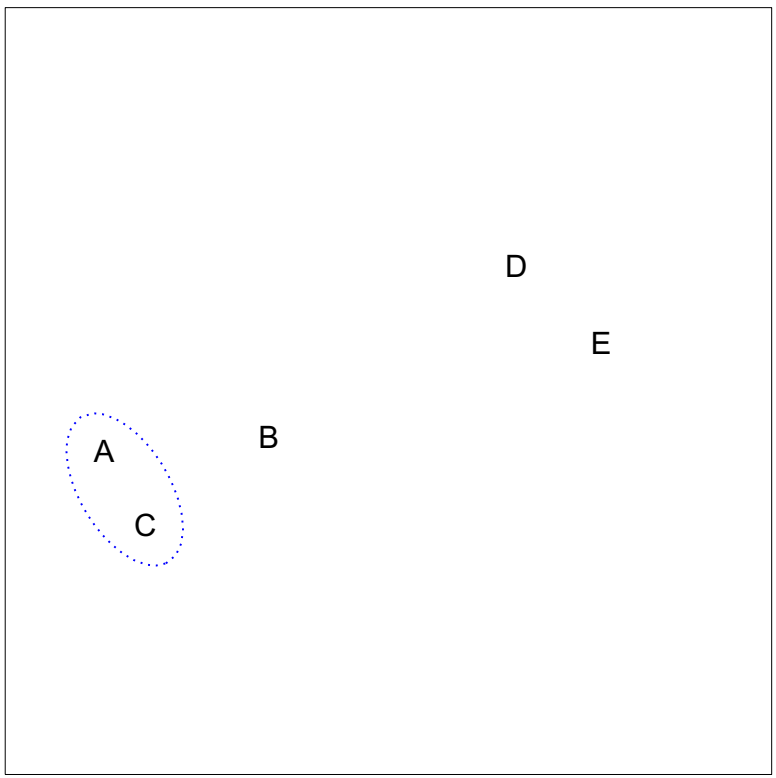
# Hierarchical Clustering: the idea

Builds a hierarchy in a “bottom-up” fashion...



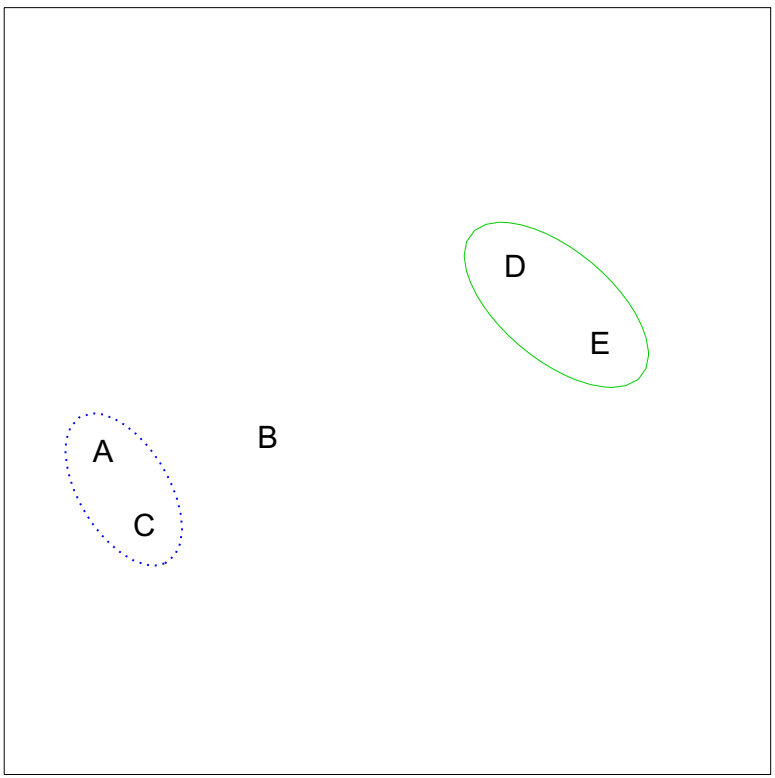
# Hierarchical Clustering: the idea

Builds a hierarchy in a “bottom-up” fashion...



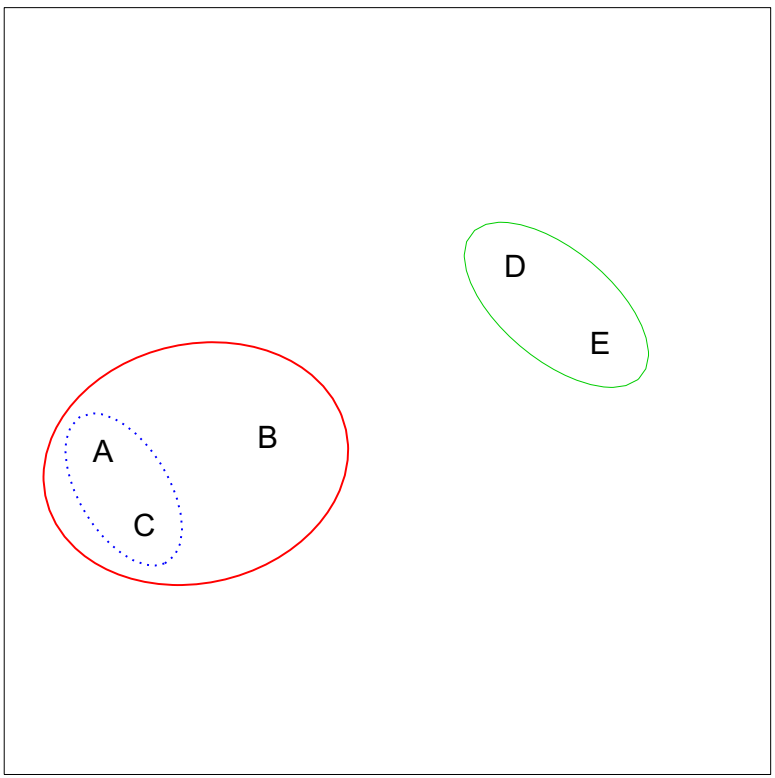
# Hierarchical Clustering: the idea

Builds a hierarchy in a “bottom-up” fashion...



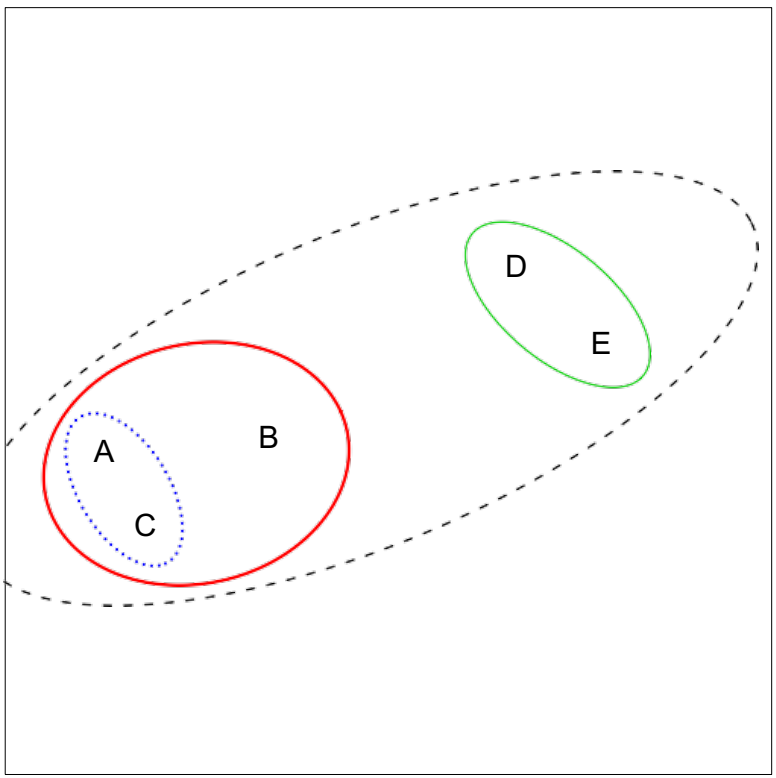
# Hierarchical Clustering: the idea

Builds a hierarchy in a “bottom-up” fashion...



# Hierarchical Clustering: the idea

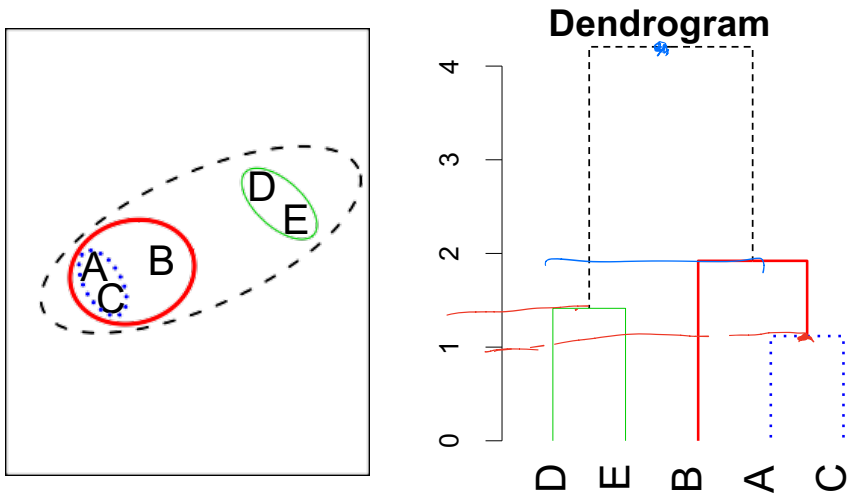
Builds a hierarchy in a “bottom-up” fashion...



# Hierarchical Clustering Algorithm

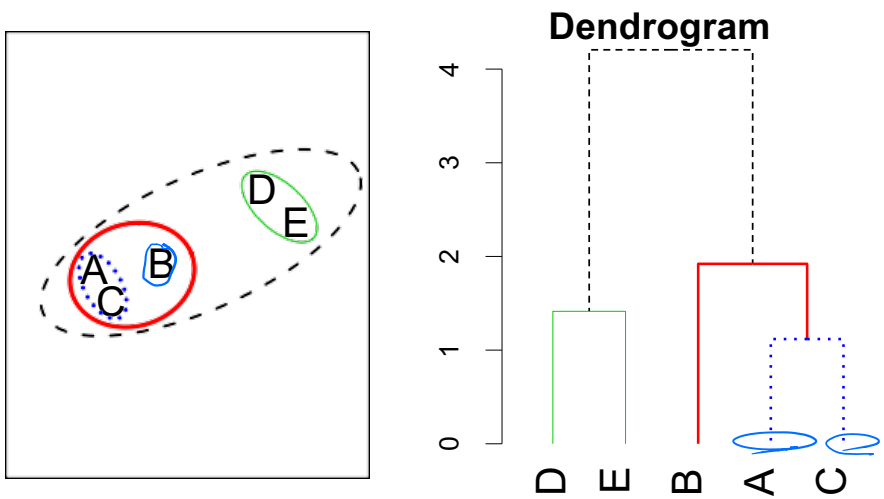
The approach in words:

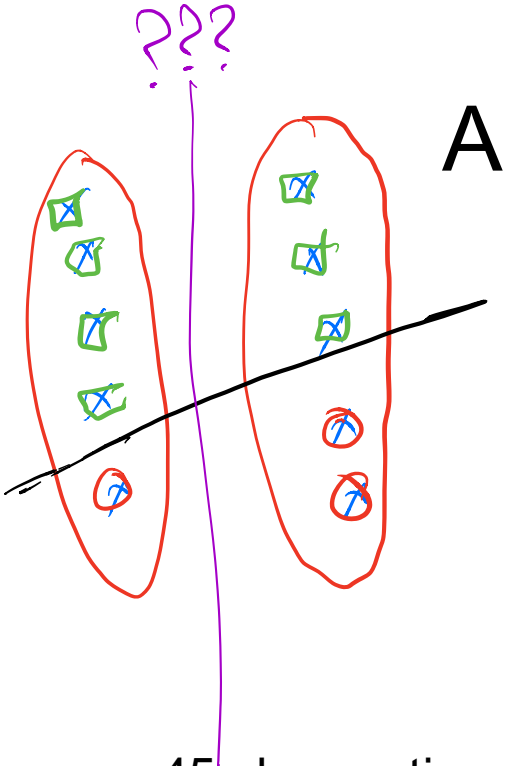
- Start with each point in its own cluster.
- Identify the closest two clusters and merge them.
- Repeat.
- Ends when all points are in a single cluster.



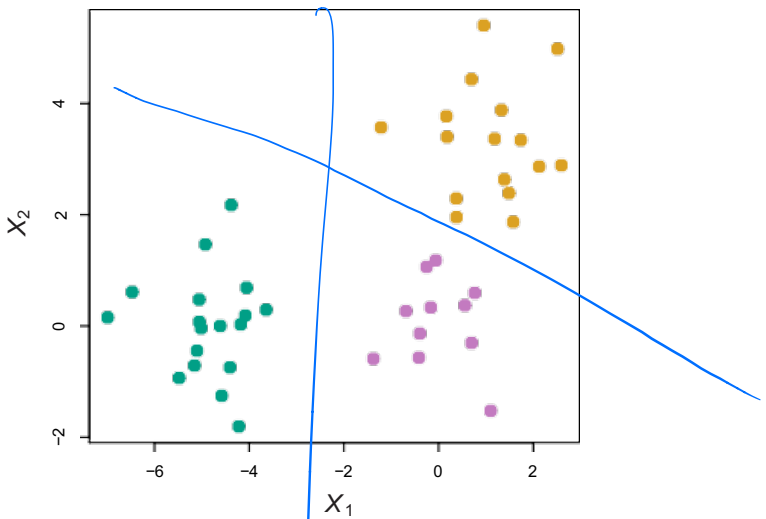
# Hierarchical Clustering Algorithm

The height of each node in the plot is proportional to the value of the intergroup dissimilarity between its two daughters





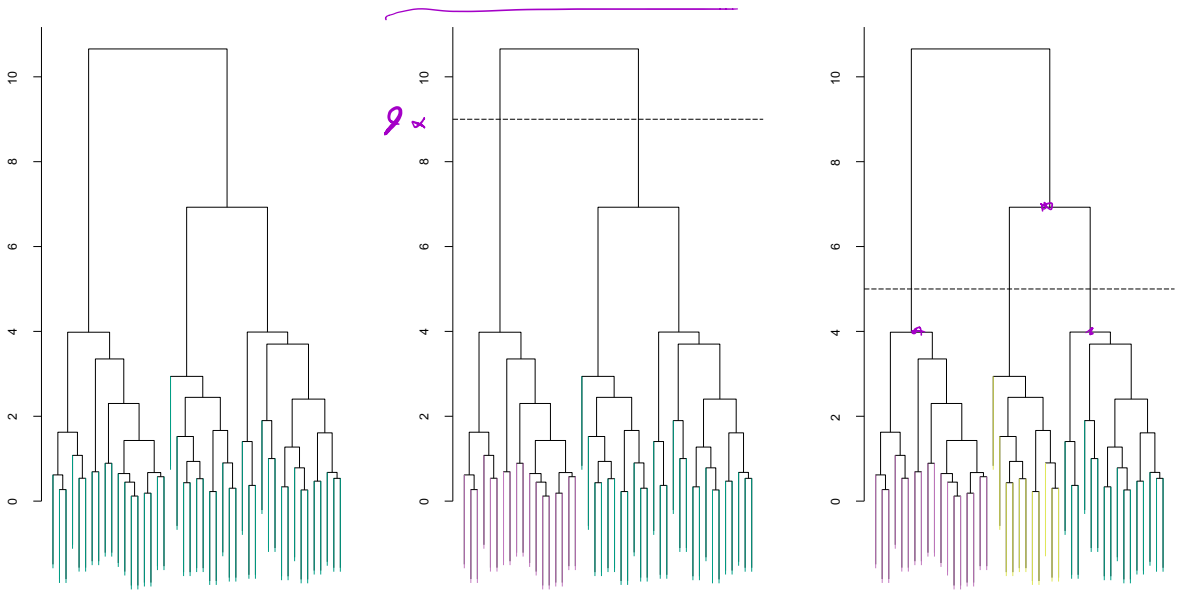

# An Example




45 observations generated in 2-dimensional space. In reality there are three distinct classes, shown in separate colors.

However, we will treat these class labels as unknown and will seek to cluster the observations in order to discover the classes from the data.

# Application of hierarchical clustering



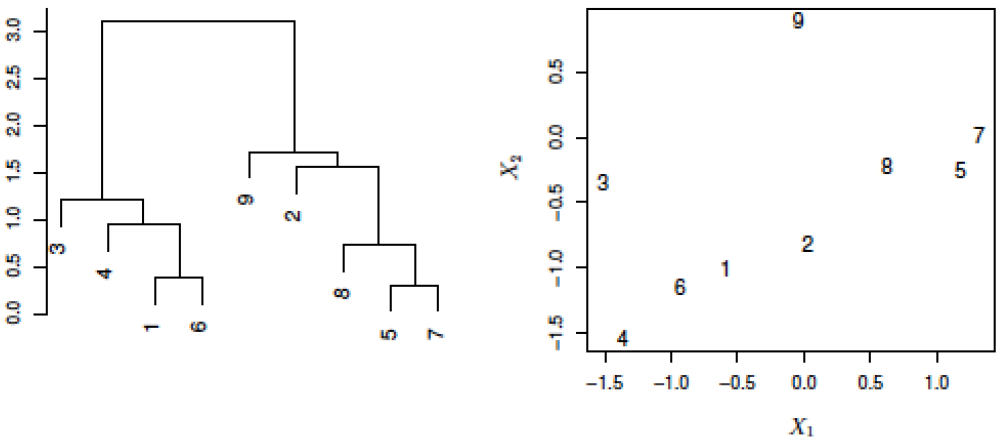
# Details of previous figure

- *Left:* Dendrogram obtained from hierarchically clustering the data from previous slide, with complete linkage and Euclidean distance.
- *Center:* The dendrogram from the left-hand panel, cut at a height of 9 (indicated by the dashed line). This cut results in two distinct clusters, shown in different colors.

# Details of previous figure

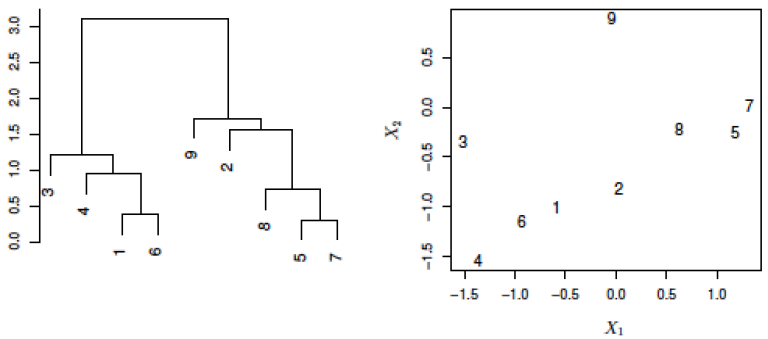
- *Right:* The dendrogram from the left-hand panel, now cut at a height of 5. This cut results in three distinct clusters, shown in different colors.
- Note that the colors were not used in clustering, but are simply used for display purposes in this figure

# Another Example



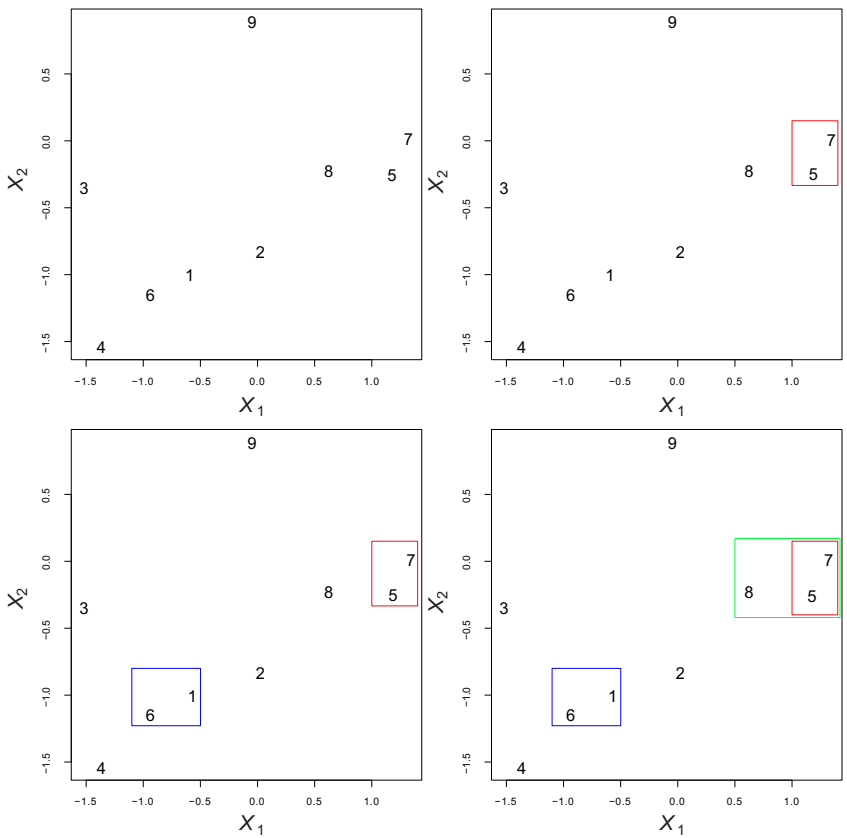

- An illustration of how to properly interpret a dendrogram with nine observations in two-dimensional space. The raw data on the right was used to generate the dendrogram on the left.
- Observations 5 and 7 are quite similar to each other, as are observations 1 and 6.

# Another Example




- However, observation 9 is *no more similar to* observation 2 than it is to observations 8, 5, and 7, even though observations 9 and 2 are close together in terms of horizontal distance.
- This is because observations 2, 8, 5, and 7 all fuse with observation 9 at the same height, approximately 1.8.

# Merges in previous example



# Algorithm

**Algorithm 10.2** *Hierarchical Clustering*

1. Begin with $n$ observations and a measure (such as Euclidean distance) of all the $\binom{n}{2} = n(n-1)/2$ 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. The dissimilarity between these two clusters indicates the height in the dendrogram at which the fusion should be placed.

   (b) Compute the new pairwise inter-cluster dissimilarities among the $i-1$ remaining clusters.

# Nested Clusters

- The term hierarchical refers to the fact that clusters obtained by cutting the dendrogram at a given height are necessarily nested within the clusters obtained by cutting the dendrogram at any greater height.
- However, on an arbitrary data set, this assumption of hierarchical structure might be unrealistic
- Hierarchical clustering can sometimes yield worse (i.e. less accurate) results than K -means clustering for a given number of clusters

# Dissimilarity between Groups

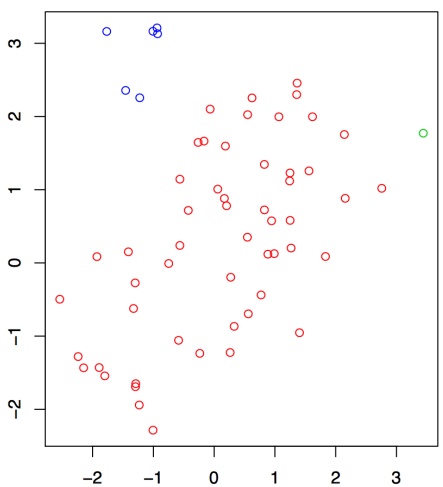
- The concept of dissimilarity between a pair of observations needs to be extended to a pair of groups of observations .
- This extension is achieved by developing the notion of linkage, which defines the dissimilarity between two groups of observations. The four most common types of linkage—complete, average, single, and centroid

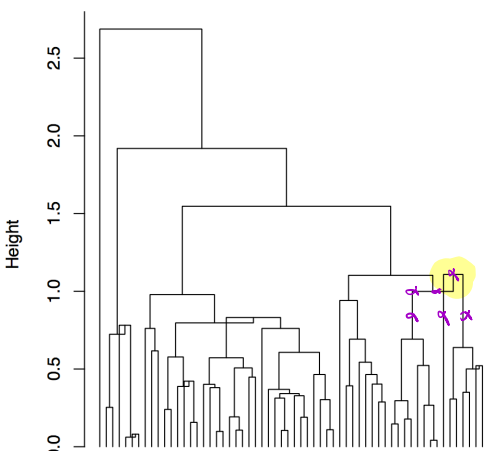
# Types of Linkage

| *Linkage* | *Description* |
|---|---|
| Complete | Maximal inter-cluster dissimilarity. Compute all pairwise dissimilarities between the observations in cluster A and the observations in cluster B, and record the *largest* of these dissimilarities. |
| Single | Minimal inter-cluster dissimilarity. Compute all pairwise dissimilarities between the observations in cluster A and the observations in cluster B, and record the *smallest* of these dissimilarities. |
| Average | Mean inter-cluster dissimilarity. Compute all pairwise dissimilarities between the observations in cluster A and the observations in cluster B, and record the *average* of these dissimilarities. |
| Centroid | Dissimilarity between the centroid for cluster A (a mean vector of length *p*) and the centroid for cluster B. Centroid linkage can result in undesirable *inversions*. |

# Types of Linkage

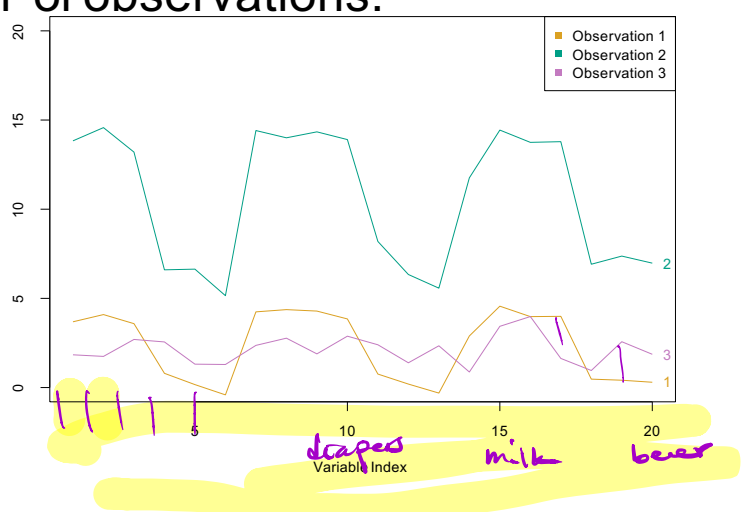
- For centroid dissimilarity, inversion can occur, whereby two clusters are fused at a height below either of the individual clusters in the dendrogram. This can lead to difficulties in visualization as well as in interpretation of the dendrogram.



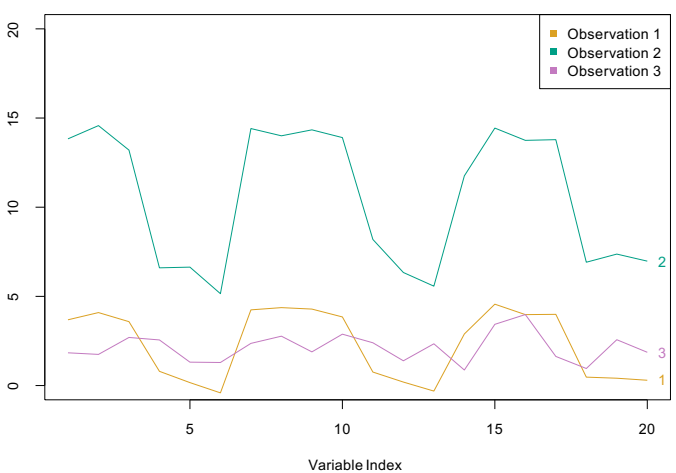
# Choice of Dissimilarity Measure

- So far have used Euclidean distance.
- An alternative is *correlation-based distance* which considers two observations to be similar if their features are highly correlated.
- This is an unusual use of correlation, which is normally computed between variables; here it is computed between the observation profiles for each pair of observations.



# Choice of Dissimilarity Measure

- Correlation-based distance focuses on the shapes of observation profiles rather than their magnitudes.
- Observations 1, 3 have small Euclidean distance but weak correlation (low correlation similarity).
- Observations 1, 2 have large Euclidean distance but strong correlation (high correlation similarity).



# Practical issues

- Should the observations or features first be standardized in some way? For instance, maybe the variables should be centered to have mean zero and scaled to have standard deviation one.
- In the case of hierarchical clustering,
  - What dissimilarity measure should be used?
  - What type of linkage should be used?
- How many clusters to choose? (in both *K*-means or hierarchical clustering). Difficult problem. No agreed-upon method. See Elements of Statistical Learning, chapter 13 for more details.

# How many clusters?

- Sometimes, we might have no problem specifying the number of clusters K ahead of time, e.g.,
  - Segmenting a client database into K clusters for *K* salesmen
- Other times, *K* is implicitly defined by cutting a hierarchical clustering tree at a given height
- In most exploratory applications, *K* is unknown.
- What is the “right” value of K?

# This is a hard problem

Determining the number of clusters is a hard problem!

Why is it hard?

- Determining the number of clusters is a hard task for humans to perform (unless the data are low-dimensional). Not only that, it's just as hard to explain what it is we're looking for. Usually, statistical learning is successful when at least one of these is possible

# This is a hard problem

- Why is it important?
  - E.g., it might mean a big difference scientifically if we were convinced that there were $K = 2$ subtypes of breast cancer vs. $K = 3$ subtypes
  - One of the (larger) goals of data mining/statistical learning is automatic inference; choosing $K$ is certainly part of this.

# Reminder: within-cluster variation

We focus on K-means, but most ideas apply to other settings

Recall: given the number of clusters *K*, the *K*-means algorithm approximately minimizes the within-cluster variation:

$$W = \sum_{k=1}^{K} \sum_{C(i)=k} \|X_i - \bar{X}_k\|_2^2$$

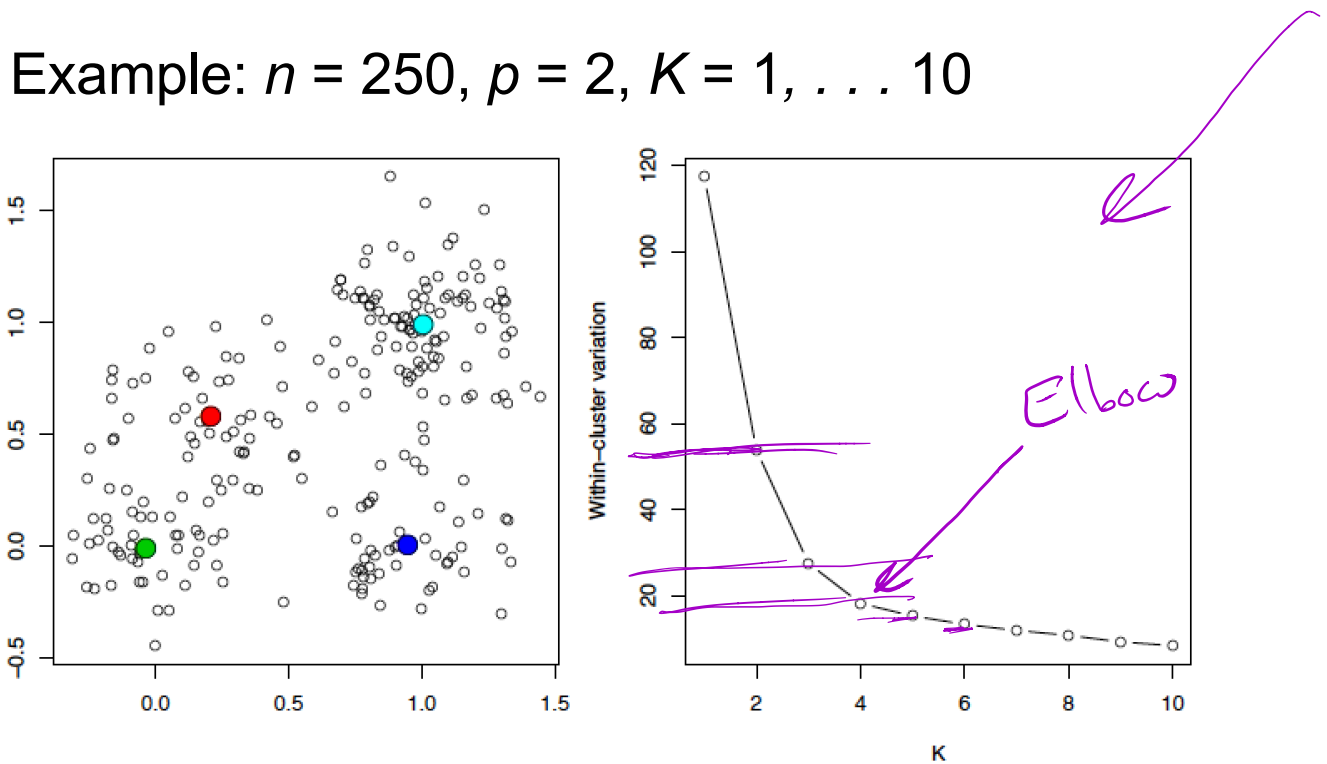
over clustering assignments *C*, where $\overline{X}_k$ is the average of points in group *k*

Clearly a lower value of W is better. So why not just run *K*-means for a bunch of different values of K, and choose the value of K that gives the smallest *W* (*K*)?

# That's not going to work

Problem: within-cluster variation just keeps decreasing: scree plot

Example: $n = 250$, $p = 2$, $K = 1, \ldots 10$



4/29/20

# Between-cluster variation

Within-cluster variation measures how tightly grouped the clusters are. As we increase the number of clusters $K$, this just keeps going down. What are we missing?

Between-cluster variation measures how spread apart the groups are from each other:

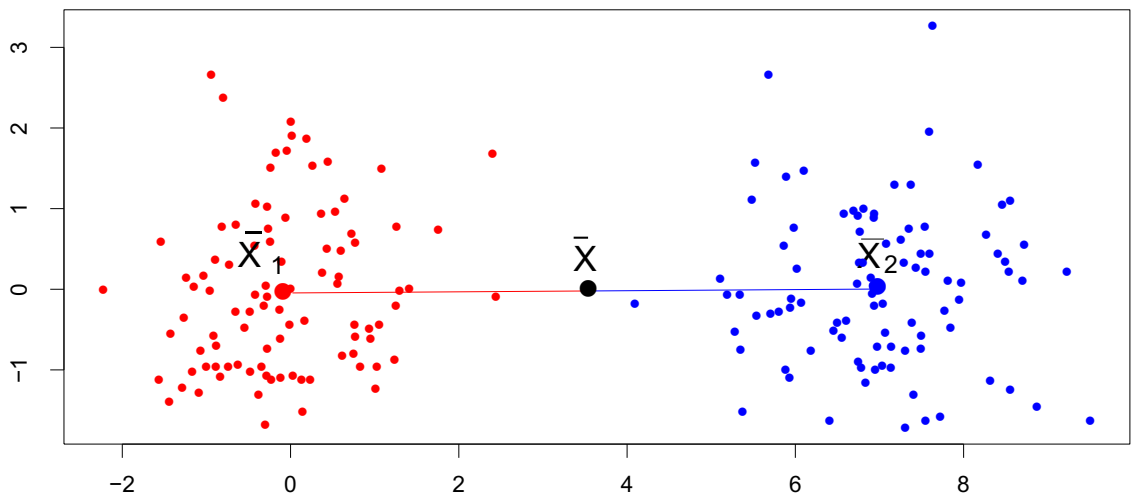
$$B = \sum_{k=1}^{K} n_k \, \|\bar{X}_k - \bar{X}\|_2^2$$

where as before $\bar{X}_k$ is the average of points in group $k$, and $\bar{X}$ is the overall average, i.e.

$$\bar{X}_k = \frac{1}{n_k} \sum_{C(i)=k} X_i \quad \text{and} \quad \bar{X} = \frac{1}{n} \sum_{i=1}^{n} X_i$$

# Example: between-cluster variation

Example: $n = 100$, $p = 2$, $K = 2$




$$B = n_1 \|\bar{X}_1 - \bar{X}\|_2^2 + n_2 \|\bar{X}_2 - \bar{X}\|_2^2$$

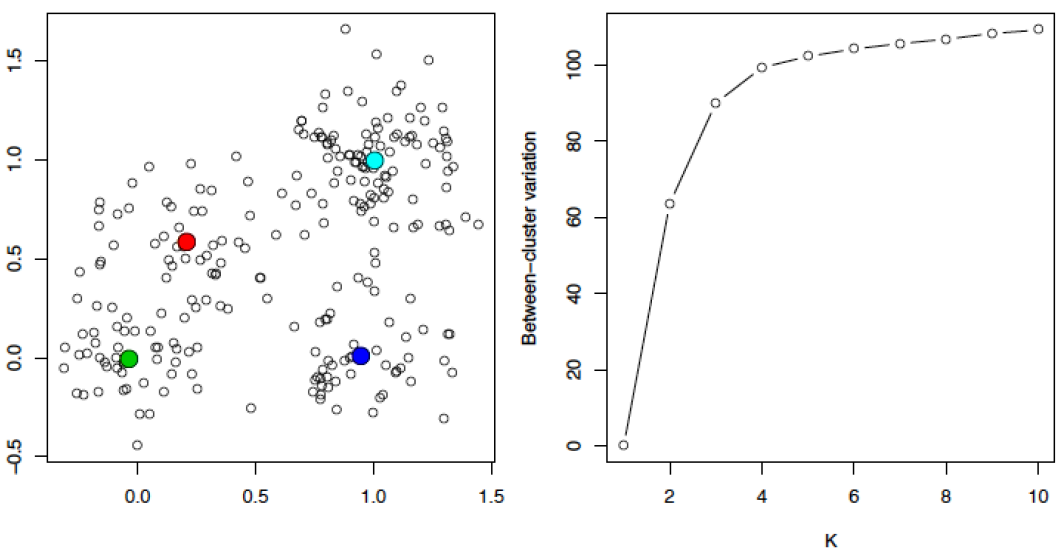
$$W = \sum_{C(i)=1} \|X_i - \bar{X}_1\|_2^2 + \sum_{C(i)=2} \|X_i - \bar{X}_2\|_2^2$$

# Still not going to work

Bigger B is better, can we use it to choose $K$?

Problem: between- cluster variation just keeps increasing

Running example: $n = 250$, $p = 2$, $K = 1, \ldots 10$



# CH index

- Ideally we'd like our clustering assignments $C$ to simultaneously have a small W and a large $B$

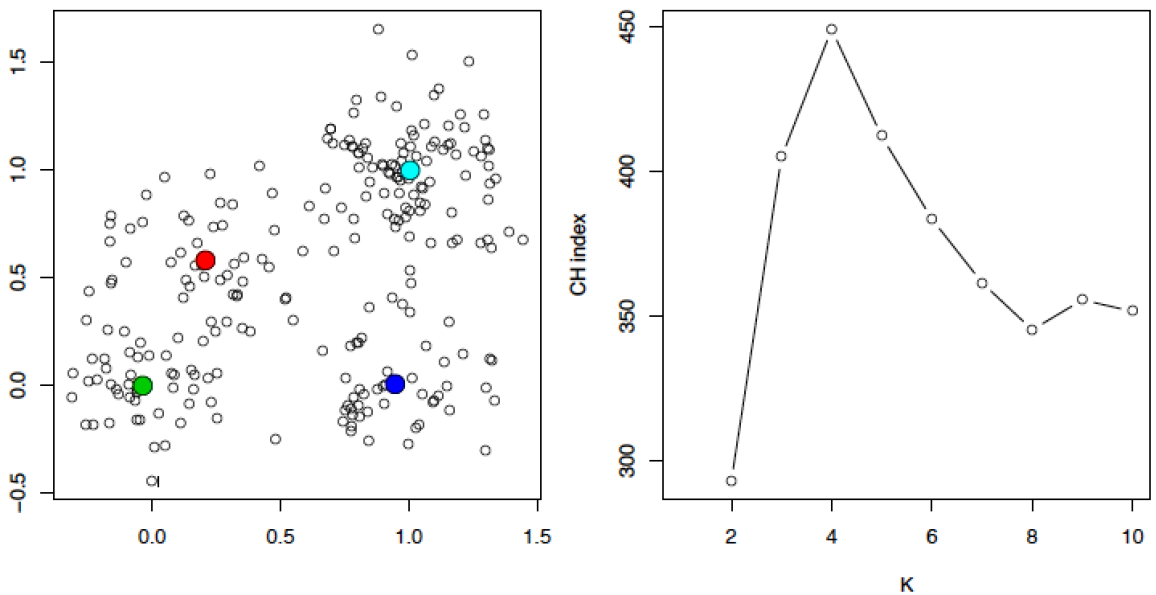
- This is the idea behind the CH index proposed by Calinski and Harabasz (1974), in "A dendrite method for cluster analysis"
- For clustering assignments coming from K clusters, we record CH score:

$$\text{CH}(K) = \frac{B(K)/(K-1)}{W(K)/(n-K)}$$

- To choose $K$, just pick some maximum number of clusters to be considered $K_{\max}$ (e.g., $K = 20$), and choose the value of $K$ with the largest score CH($K$).

# Example: CH index

Running example: $n = 250$, $p = 2$, $K = 2, \ldots 10$.




We would choose $K = 4$ clusters, which seems reasonable

General problem: the CH index is not defined for $K = 1$. We could never choose just one cluster (the null model)!

# Gap statistic

It's true that W(K) keeps dropping, but how much it drops at any one *K* should be informative

The gap statistic was introduced by Tibshirani et al. (2001), "Estimating the number of clusters in a data set via the gap statistic"

# Gap statistic

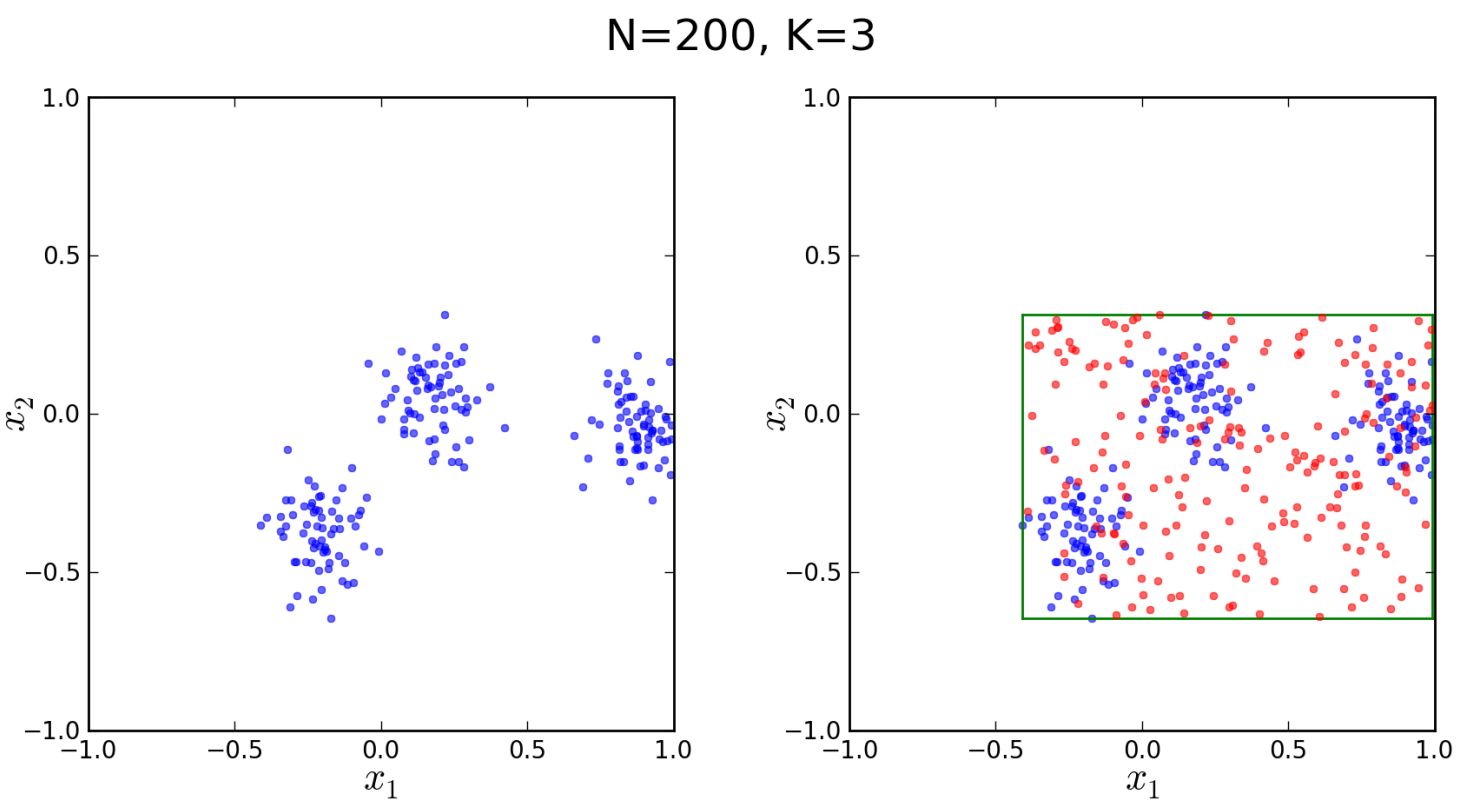
It is based on this idea. We compare the observed within-cluster variation $W(K)$ to $W_{unif}(K)$, the within-cluster variation we'd see if we instead had points distributed uniformly (over an *encapsulating box*). The gap for $K$ clusters is defined as

$$\text{Gap}(K) = \log W_{unif}(K) - \log W(K)$$

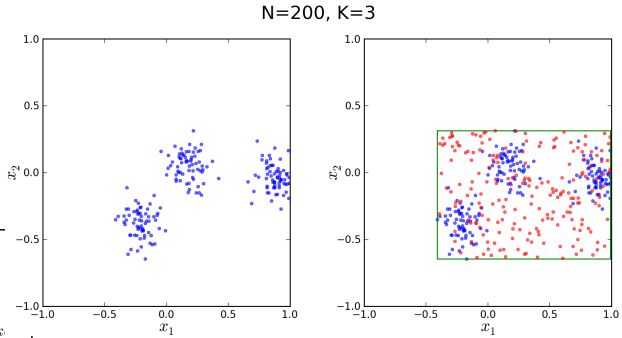
# Gap statistic

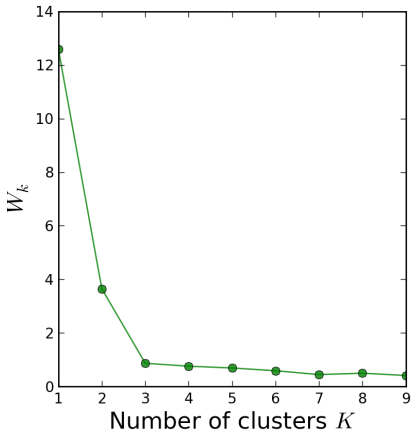
The reference datasets are in our case generated by sampling uniformly from the original dataset's bounding box (see green box in the upper right plot of the figures below). To obtain the estimate log $W_{\text{unif}}(K)$, we compute the average of $B$ copies of log $W(K)$, each of which is generated from the uniform sample, and their standard deviation $s(K)$.

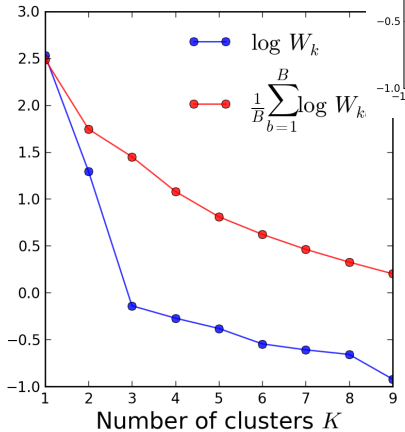
# Gap statistic

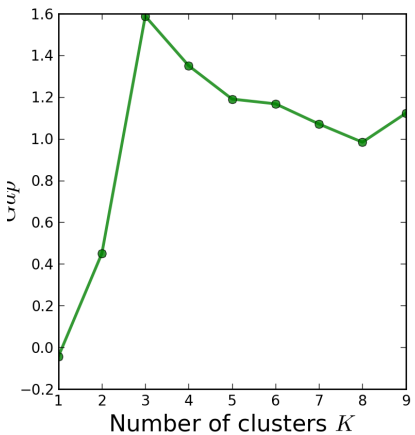


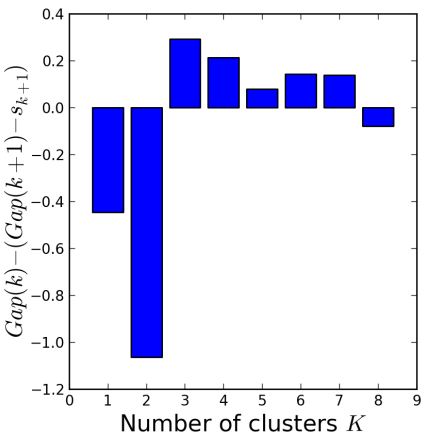
# Gap statistic



# Gap statistic

Then we choose K by:

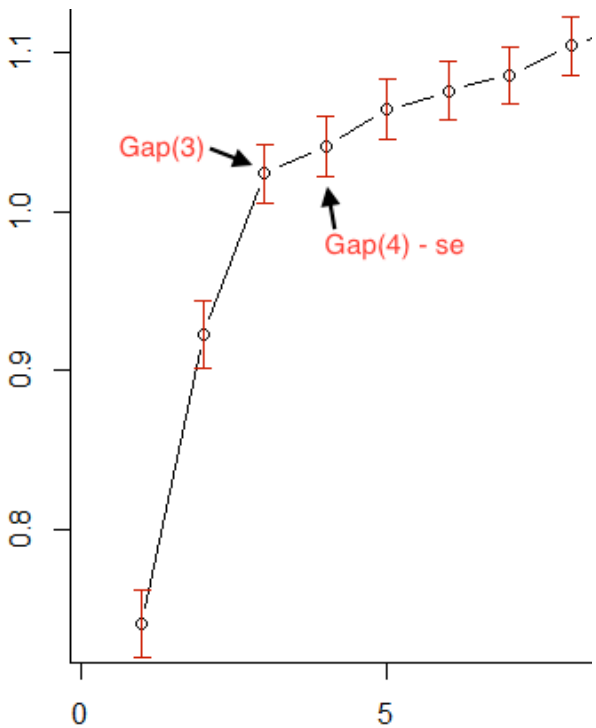
$$\hat{K} = \min_{K \in \{1, \ldots K_{max}\}} : \text{Gap}(K) \geq \text{Gap}(K+1) - s(K+1),$$

# Gap statistic

This means:

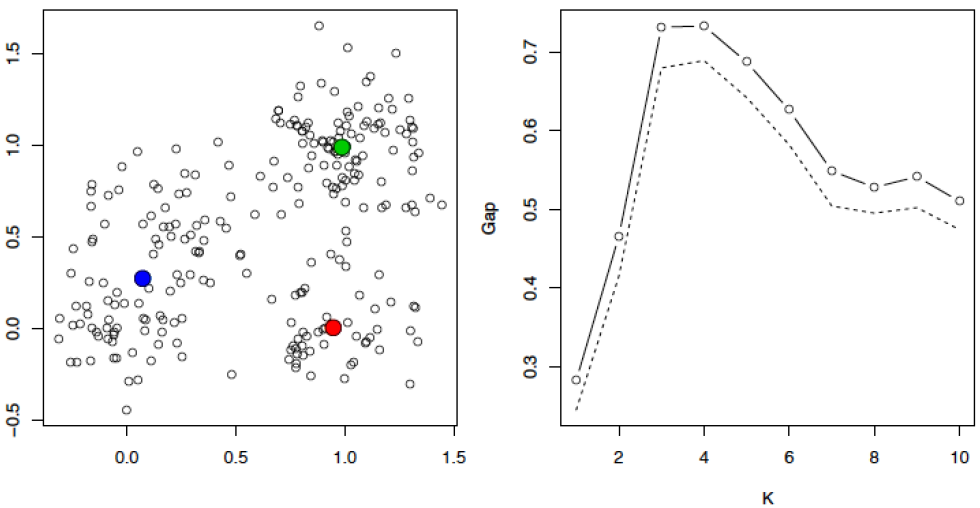
Choose the cluster size $\hat{k}$ to be the smallest k such that

$$Gap(k) \geq Gap(k+1) - s(k+1)$$



# Example: gap statistic

Running example: $n = 250$, $p = 2$, $K = 1, \ldots 10$




We would choose K = 3 clusters, which is also reasonable

The gap statistic does especially well when the data fall into one cluster. (Why? Hint: think about the null distribution that it uses)

# CH index and gap statistic in R

The CH index can be computed using the kmeans function in the base distribution, which returns both the within-cluster variation and the between-cluster varation
E.g.,

```
k = 5
km = kmeans(x, k, alg="Lloyd")
names(km)
# Now use some of these return items to compute ch
```

The gap statistic is implemented by the function gap in the package lga, and by the function gap in the package SAGx. (Beware: these functions are poorly documented ... it's unclear what clustering method they're using)

# Silhouette Analysis

- A method of interpretation and validation of consistency within clusters of data
- Provides a succinct graphical representation of how well each object lies within its cluster

# Silhouette Analysis

- Assume the data have been clustered via any technique, such as k-means, into *k* clusters.
- For each sample $\mathbf{x}_i$, let $a_i$ be the average distance between $\mathbf{x}_i$ and all other data within the same cluster.
- Can interpret $a_i$ as a measure of how well $\mathbf{x}_i$ is assigned to its cluster (the smaller the value, the better the assignment).

# Silhouette Analysis

- We then define the average dissimilarity of point $\mathbf{x}_i$ to a cluster $c$ as the average of the distance from $\mathbf{x}_i$ i to all points in $c$.
- Let $b_i$ be the lowest average distance of $\mathbf{x}_i$ to all points in any other cluster, of which $\mathbf{x}_i$ is not a member.

# Silhouette Analysis

- The cluster with this lowest average dissimilarity is said to be the "neighboring cluster" of $\mathbf{x}_i$, because it is the next best fit cluster for point $\mathbf{x}_i$. We now define a silhouette:

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

- or

$$s_i = \begin{cases} 1 - a_i / b_i & a_i < b_i \\ 0 & a_i = b_i \\ b_i / a_i - 1 & a_i > b_i \end{cases}$$

# Silhouette Analysis

- From

$$s_i = \begin{cases} 1 - a_i / b_i & a_i < b_i \\ 0 & a_i = b_i \\ b_i / a_i - 1 & a_i > b_i \end{cases}$$

it is obvious that $-1 \leq s_i \leq 1$.

# Silhouette Analysis

- $s_i$ close to 1 requires $a_i \ll b_i$.
- $a_i$ is a measure of dissimilarity of $\mathbf{x}_i$ to its own cluster
- Thus a small $a_i$ means $\mathbf{x}_i$ is well matched.
- Large $b_i$ implies that $\mathbf{x}_i$ is badly matched to its neighboring cluster.
- Thus an $s_i$ close to one means that $\mathbf{x}_i$ is appropriately clustered.

# Silhouette Analysis

- $s_i$ close to negative one, by the same logic means that $\mathbf{x}_i$ would be more appropriate if it was clustered in its neighboring cluster.
- An $s_i$ near zero means that $\mathbf{x}_i$ is on the border of two natural clusters.

# Silhouette Analysis

- The average $s_i$ over a cluster measures how tightly grouped all the data in the cluster are.
- Thus the average $s_i$ over the entire dataset is a measure of how appropriately the data have been clustered.

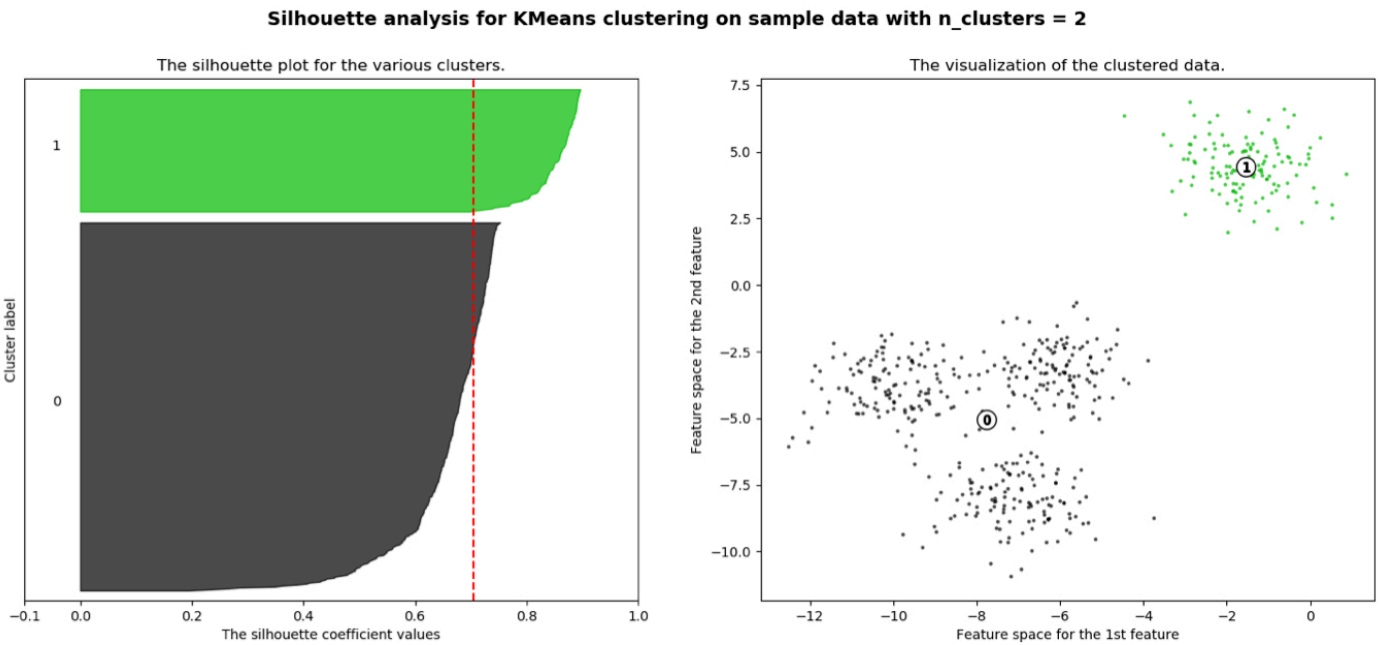
# Silhouette Analysis

- If there are too many or too few clusters, some of the clusters will display narrower silhouettes than the rest.
- So silhouette plots and averages may be used to determine the natural number of clusters within a dataset.

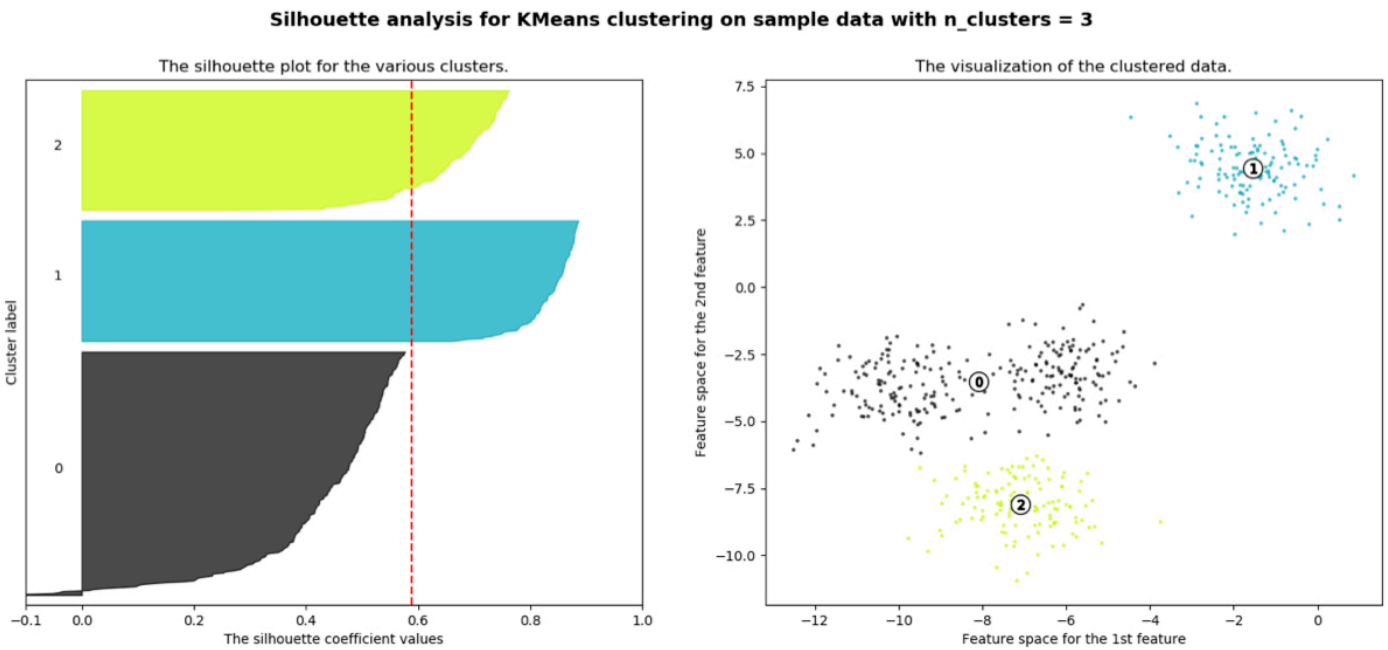
# Silhouette Analysis

- One can also increase the likelihood of the silhouette being maximized at the correct number of clusters by re-scaling the data using feature weights that are cluster specific.

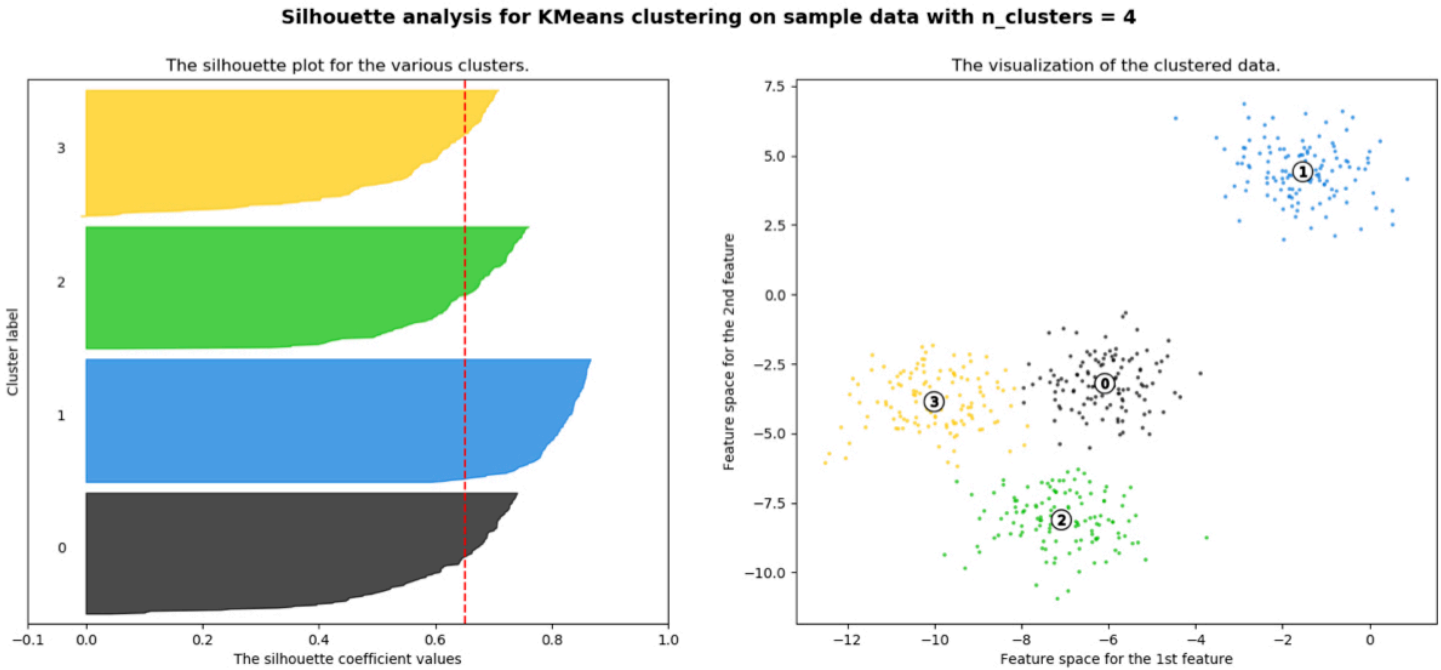
# Silhouette Plot



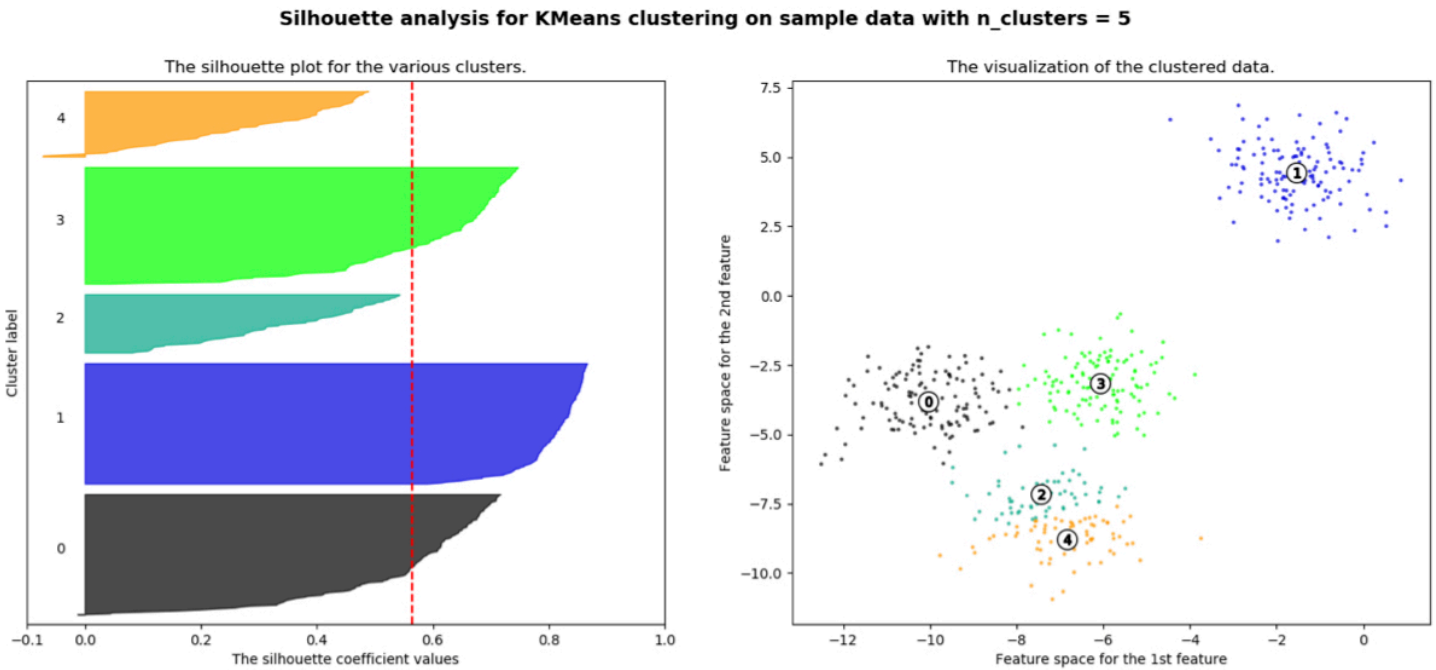
# Silhouette Plot



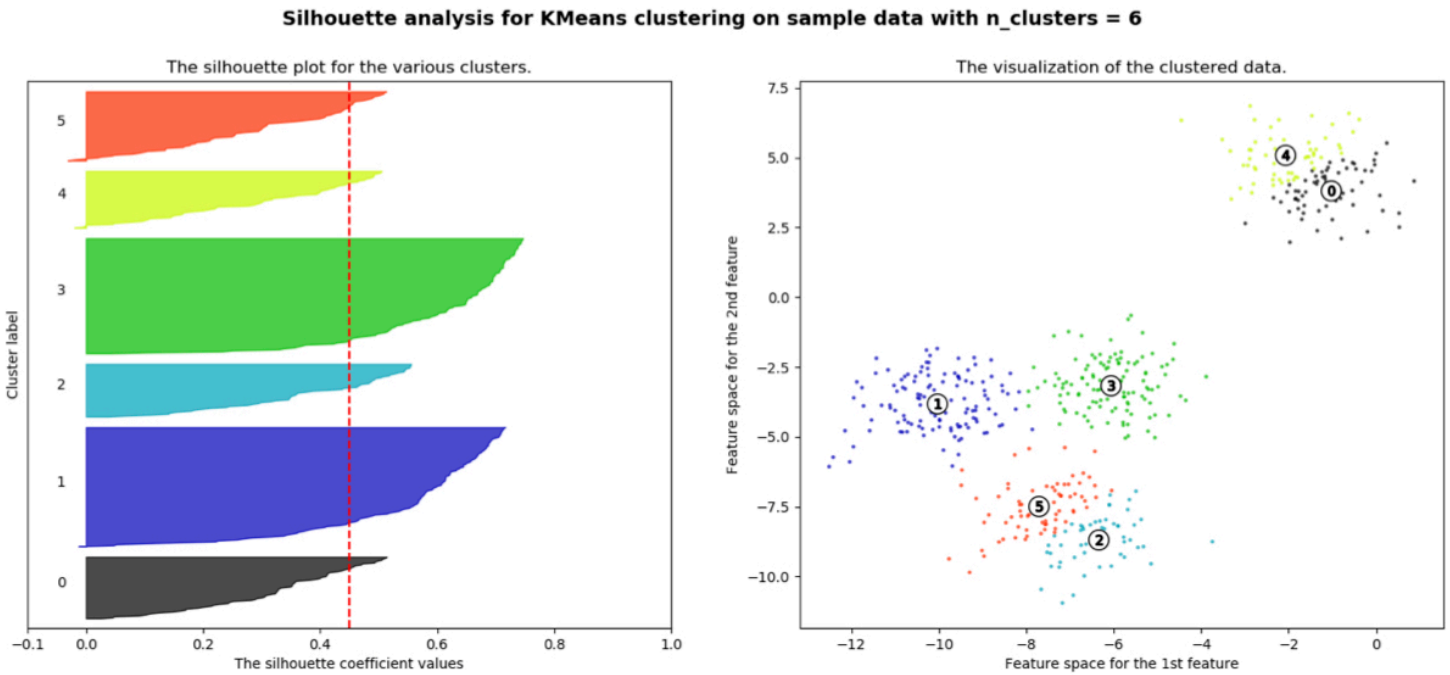
# Silhouette Plot



# Silhouette Plot



# Silhouette Plot



# Silhouette Plot

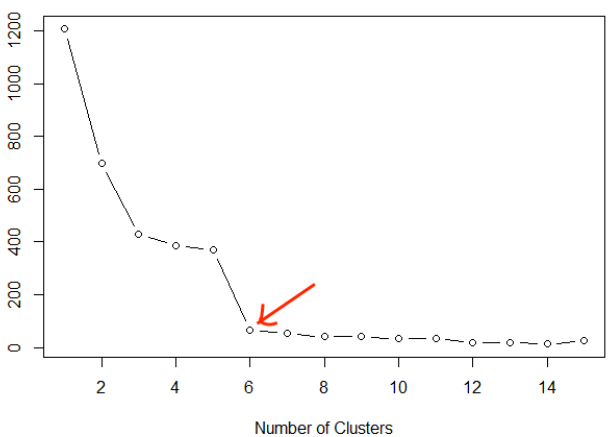
- The silhouette plot shows that k= 3, 5 and 6 are a bad pick for the given data due to the presence of clusters with below average silhouette scores and also due to wide fluctuations in the size of the silhouette plots.
- Silhouette analysis is more ambivalent in deciding between 2 and 4.

# Cross Validation

- The data is partitioned into $v$ folds.
- Each of the folds is then held out at turn as a test set, a clustering model computed on the other $v - 1$ training sets
- The value of an objective function is calculated for the test set.
- Example of objective function: the sum of the squared distances to the centroids for $k$-means

# Cross Validation

- These *v* values are calculated and averaged for each alternative number of clusters *c*, and the cluster number selected such that further increase in number of clusters leads to only a small reduction in the objective function (scree plots)



# More

- https://stackoverflow.com/questions/15376075/cluster-analysis-in-r-determine-the-optimal-number-of-clusters/15376462#15376462

# More

- https://stats.stackexchange.com/questions/3685/where-to-cut-a-dendrogram

# Once again, it really is a hard problem

Background

Just How Many Clusters are there in the Galaxy Data?

- Galaxy Data from Postman *et al.* (1986): measurements of velocities in $10^3$ km/sec of 82 galaxies from a survey of the Corona Borealis region.
- Roeder (1990): at least 3, no more than 7 modes (Confidence set)
- Others are in consensus

| | | | | | |
|---|---|---|---|---|---|
| 9172 | 9350 | 9483 | 9558 | 9775 | 10227 |
| 10406 | 16084 | 16170 | 18419 | 18552 | 18600 |
| 18927 | 19052 | 19070 | 19330 | 19343 | 19349 |
| 19440 | 19473 | 19529 | 19541 | 19547 | 19663 |
| 19846 | 19856 | 19863 | 19914 | 19918 | 19973 |
| 19989 | 20166 | 20175 | 20179 | 20196 | 20215 |
| 20221 | 20415 | 20629 | 20795 | 20821 | 20846 |
| 20875 | 20986 | 21137 | 21492 | 21701 | 21814 |
| 21921 | 21960 | 22185 | 22209 | 22242 | 22249 |
| 22314 | 22374 | 22495 | 22746 | 22747 | 22888 |
| 22914 | 23206 | 23241 | 23263 | 23484 | 23538 |
| 23542 | 23666 | 23706 | 23711 | 24129 | 24285 |
| 24289 | 24366 | 24717 | 24990 | 25633 | 26960 |
| 26995 | 32065 | 32789 | 34279 | | |

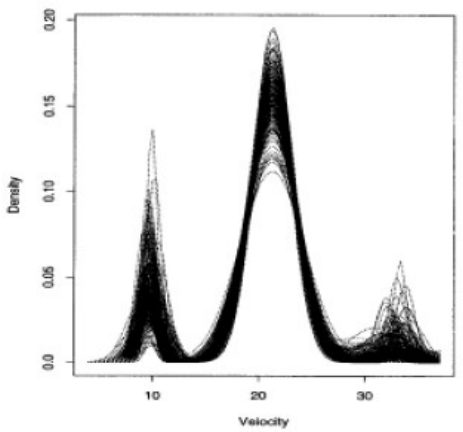
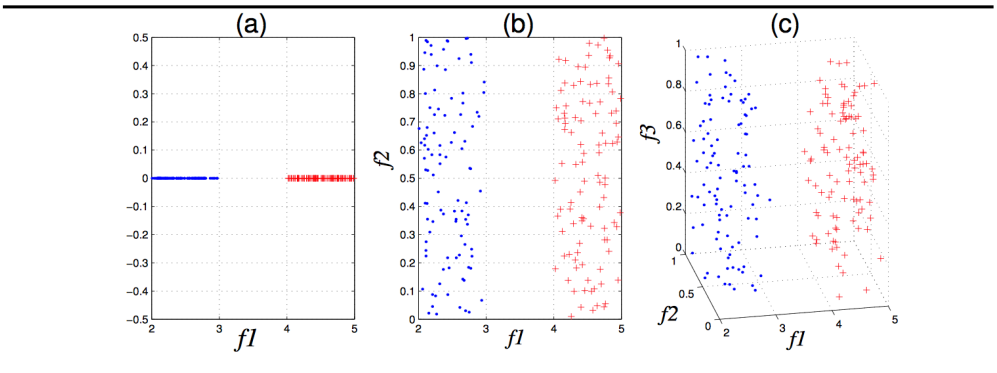



*Figure 1. Densities Obtained From the Markov Chain Monte Carlo Sampler Using the Astronomy Data From Roeder (1992).*

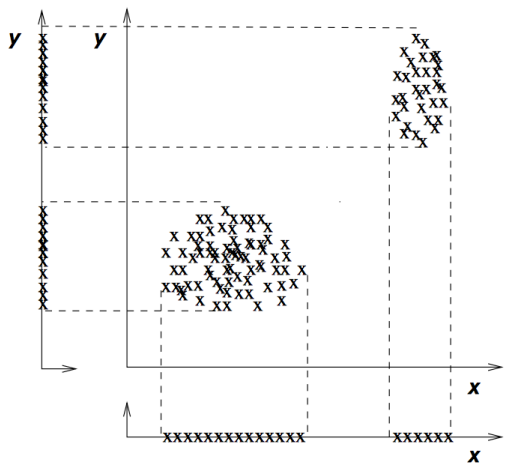
- Histogram from Roeder and Wasserman (1997)

# Variable Selection for Clustering



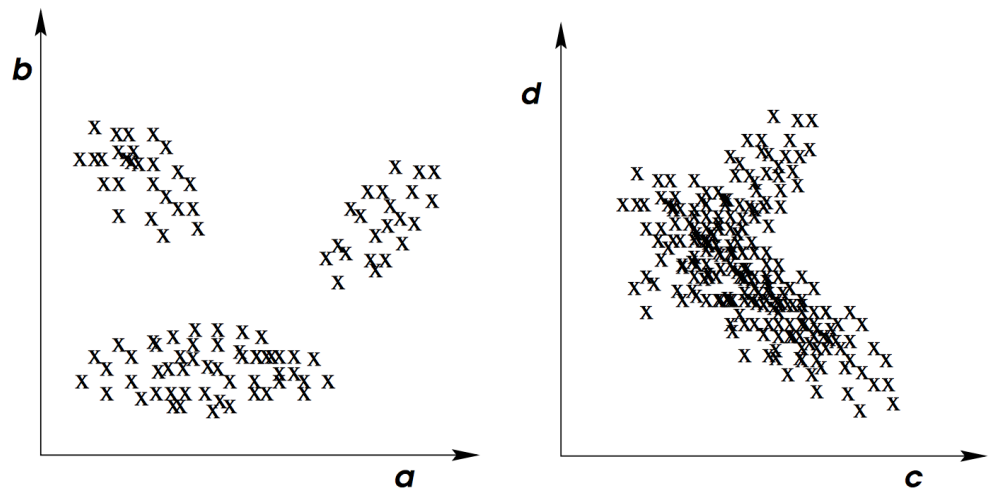

Feature $f_1$ is relevant while $f_2$ and $f_3$ are irrelevant. We are able to distinguish the two clusters from $f_1$ only. Thus, removing $f_2$ and $f_3$ will not effect the accuracy of clustering.

# Variable Selection for Clustering




In this example, features $x$ and $y$ are redundant, because feature $x$ provides the same information as feature $y$ with regard to discriminating the two clusters.

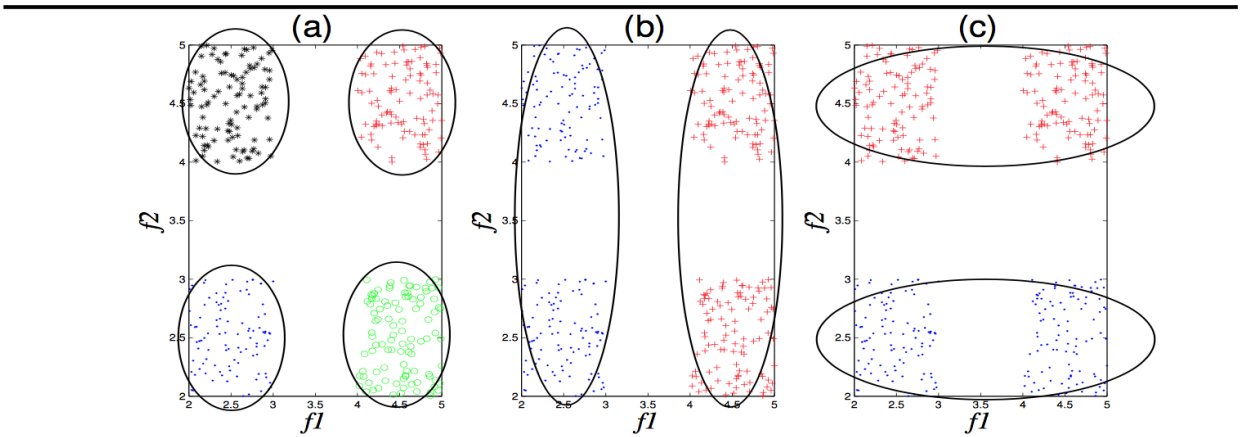
# Variable Selection for Clustering




(a) (b)

A more complex example. Figure a is the scatterplot of the data on features $a$ and $b$. Figure b is the scatterplot of the data on features $c$ and $d$.

# Variable Selection for Clustering




Different sets of features may produce different clustering.

# Variable Selection for Clustering

- The goal of feature selection for unsupervised learning is to find the smallest feature subset that best uncovers “interesting natural” groupings (clusters) from data according to the chosen criterion.

# Subset Selection

- Two paradigms:
  - feature subset selection criteria should be the criteria used for clustering
  - The two criteria need not be the same

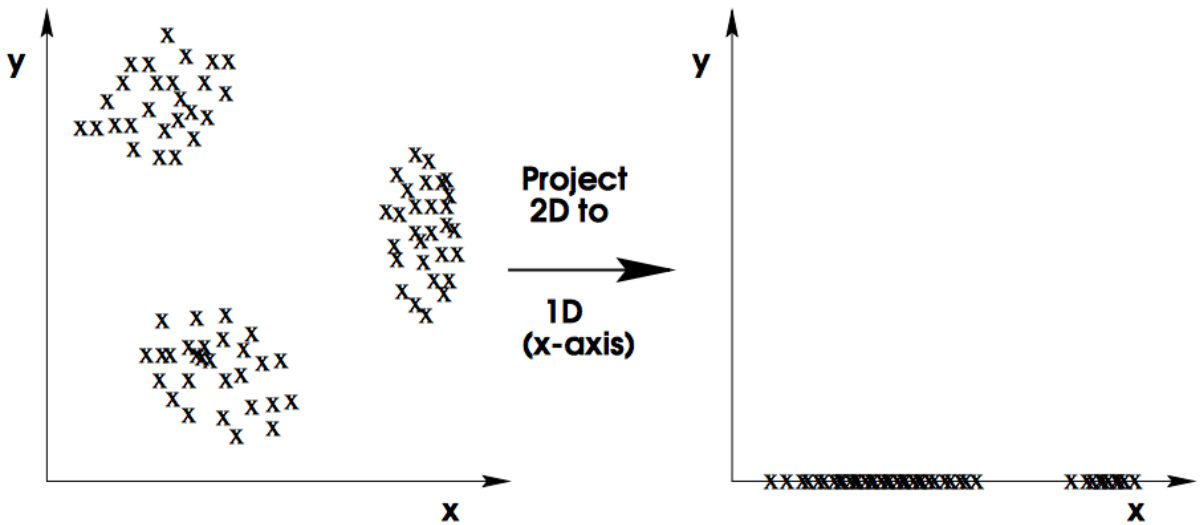
# Subset Selection

Searching for the best subset of features, we run into a new problem:
*k depends on the feature subset*.

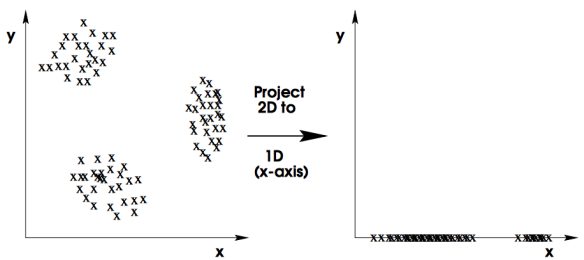
Therefore, in each step of subset selection, the best *k* has to be found.

Many ways are conceivable for best subset selection for clustering. The details are left to the students.

# Subset Selection



# Subset Selection




Two dimensions: three clusters
One-dimension: only two clusters.

A fixed number of clusters for all feature sets does not model the data in the respective subspace correctly.

# K-Means

Large number of k-means variations were proposed to handle feature selection
Most start cluster the data into *k* clusters. Then, assign weight to each feature.

The feature that minimizes the within-cluster distance / maximizes between-cluster distance is preferred, hence, gets higher weight.

# Sparse Methods Based on L1 Norm

Witten and Tibshirani formulate clustering using a parameter vector w, and use L1 penalty and optimization to obtain a sparse set of features.

See https://www.ncbi.nlm.nih.gov/pmc/articles/PMC2930825/pdf/nihms201124.pdf

# Principal Components Analysis

- PCA produces a low-dimensional representation of a dataset. It finds a sequence of linear combinations of the variables that have maximal variance, and are mutually uncorrelated.
- Apart from producing derived variables for use in supervised learning problems, PCA also serves as a tool for data visualization.

# Principal Components Analysis: details

- The *first principal component* of a set of features $X_1, X_2, \ldots, X_p$ is the normalized linear combination of the features

$$Z_1 = \varphi_{11}X_1 + \varphi_{21}X_2 + \ldots + \varphi_{p1}X_p$$

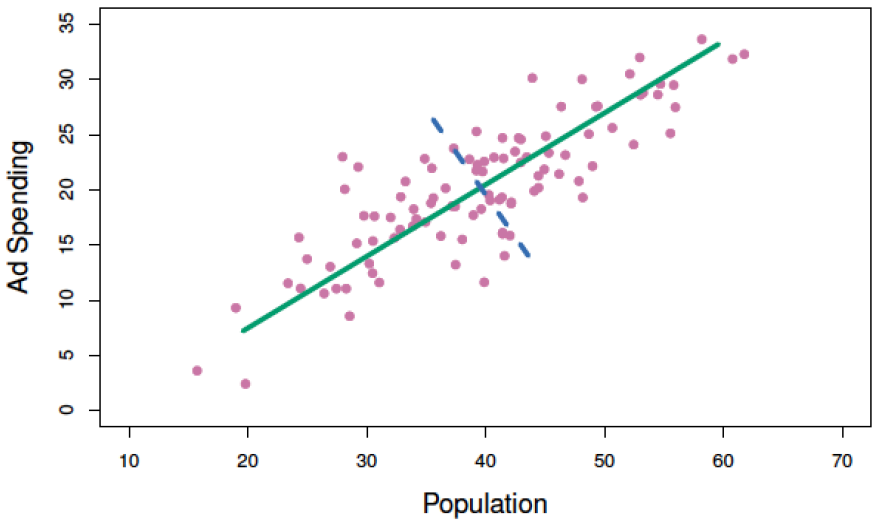
that has the largest variance. By normalized, we mean that

$$\sum_{j=1}^{p} \phi_{j1}^2 = 1$$

# Principal Components Analysis: details

- We refer to the elements $\varphi_{11}, \ldots, \varphi_{p1}$ as the loadings of the first principal component; together, the loadings make up the principal component loading vector, $\varphi_1 = (\varphi_{11}\ \varphi_{21} \ldots \varphi_{p1})^T$.
- We constrain the loadings so that their sum of squares is equal to one, since otherwise setting these elements to be arbitrarily large in absolute value could result in an arbitrarily large variance.

# PCA: example




The population size (pop) and ad spending (ad) 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.

# Computation of Principal Components

- Suppose we have a $n \times p$ data set. Since we are only interested in variance, we assume **X** that each of the variables in **X** has been centered to have mean zero (that is, the column means of **X** are zero).
- We then look for the linear combination of the sample feature values of the form $z_{i1} = \varphi_{11}x_{i1} + \varphi_{21}x_{i2} + \ldots + \varphi_{p1}x_{ip}$

  for i = 1, . . . , n that has largest sample variance, subject to the constraint that

$$\sum_{j=1}^{p} \phi_{j1}^2 = 1$$

# Computation of Principal Components

- Since each of the $x_{ij}$ has mean zero, then so does $z_{i1}$ (for any values of $\varphi_{j1}$). Hence the sample variance of the $z_{i1}$ can be written as

$$\frac{1}{n}\sum_{i=1}^{n} z_{i1}^2.$$

# Computation: continued

- Therefore, the first principal component loading vector solves the optimization problem

$$\underset{\phi_{11},\ldots,\phi_{p1}}{\text{maximize}} \frac{1}{n}\sum_{i=1}^{n}\left(\sum_{j=1}^{p}\phi_{j1}x_{ij}\right)^2 \text{ subject to } \sum_{j=1}^{p}\phi_{j1}^2 = 1$$

- This problem can be solved via a singular-value decomposition of the matrix **X**, a standard technique in linear algebra.
- We refer to $Z_1$ as the first principal component, with realized values $z_{11}, \ldots, z_{n1}$

# Geometry of PCA

- The loading vector $\varphi_1$ with elements $\varphi_{11}, \varphi_{21}, \ldots, \varphi_{p1}$ defines a direction in feature space along which the data vary the most.
- If we project the $n$ data points $x_1, \ldots, x_n$ onto this direction, the projected values are the principal component scores $z_{11}, \ldots, z_{n1}$ themselves.

# Further principal components

- The second principal component is the linear combination of $X_1, \ldots, X_p$ that has maximal variance among all linear combinations that are *uncorrelated* with $Z_1$.
- The second principal component scores $z_{12}, z_{22}, \ldots, z_{n2}$ take the form

$$z_{i2} = \varphi_{12}x_{i1} + \varphi_{22}x_{i2} + \ldots + \varphi_{p2}x_{ip},$$

where $\varphi_2$ is the second principal component loading vector, with elements $\varphi_{12}, \varphi_{22}, \ldots, \varphi_{p2}$.

# Further principal components: continued

- It turns out that constraining $Z_2$ to be uncorrelated with $Z_1$ is equivalent to constraining the direction $\varphi_2$ to be orthogonal (perpendicular) to the direction $\varphi_1$. And so on.

# Further principal components: continued

- The principal component directions $\varphi_1$, $\varphi_2$, $\varphi_3$, . . . are the ordered sequence of right singular vectors of the matrix **X**, and the variances of the components are $1/n$ times the squares of the singular values. There are at most $\min(n - 1, p)$ principal components.

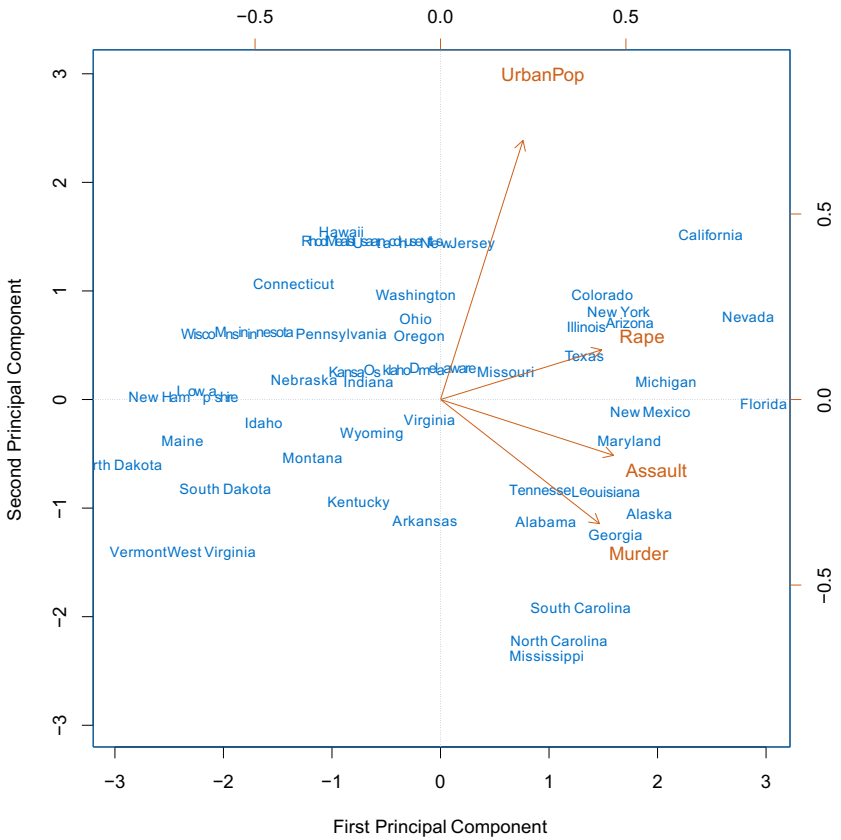
# Illustration

- USAarrests data: For each of the fifty states in the United States, the data set contains the number of arrests per 100, 000 residents for each of three crimes: Assault, Murder, and Rape. We also record UrbanPop (the percent of the population in each state living in urban areas).

# Illustration

- The principal component score vectors have length $n = 50$, and the principal component loading vectors have length $p = 4$.
- PCA was performed after standardizing each variable to have mean zero and standard deviation one.

# USAarrests data: PCA plot



# Figure details

The first two principal components for the USArrests data.

- The blue state names represent the scores for the first two principal components.

# PCA loadings

| | PC1 | PC2 |
|---|---|---|
| Murder | 0.5358995 | -0.4181809 |
| Assault | 0.5831836 | -0.1879856 |
| UrbanPop | 0.2781909 | 0.8728062 |
| Rape | 0.5434321 | 0.1673186 |

The contribution of each variable in each of the principal components PC1 and PC2 is shown as a vector. For example, Assault is shown as $[0.58,-0.19]^T$

# Figure details

- The orange arrows indicate the first two principal component loading vectors (with axes on the top and right). For example, the loading for Rape on the first component is 0.54, and its loading on the second principal component 0.17 [the word Rape is centered at the point (0.54, 0.17)].

# Figure details

- This figure is known as a *biplot*, because it displays both the principal component scores and the principal component loadings.

# Figure details

- The first loading vector places approximately equal weight on Assault, Murder, and Rape, with much less weight on UrbanPop.
- Hence this component roughly corresponds to a measure of overall rates of serious crimes.

# Figure details

- The second loading vector places most of its weight on UrbanPop and much less weight on the other three features.
- Hence, this component roughly corresponds to the level of urbanization of the state.

# Figure details

- The crime-related variables (Murder, Assault , and Rape) are located close to each other
- The UrbanPop variable is far from the other three.

# Figure details

- The crime-related variables are correlated with each other: states with high murder rates tend to have high assault and rape rates
- The UrbanPop variable is less correlated with the other three.

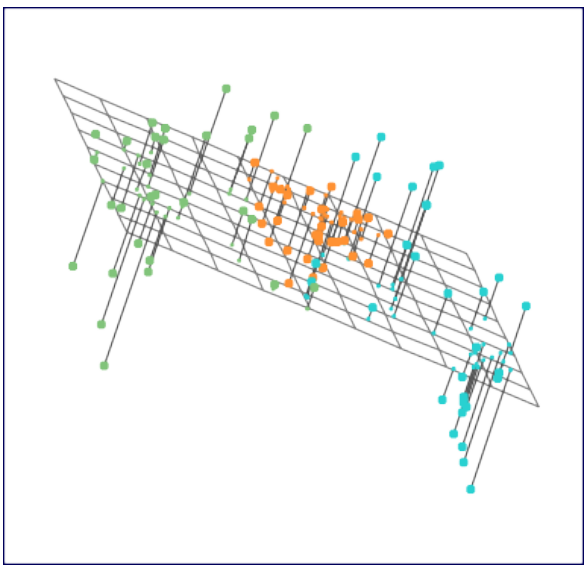
# Figure details

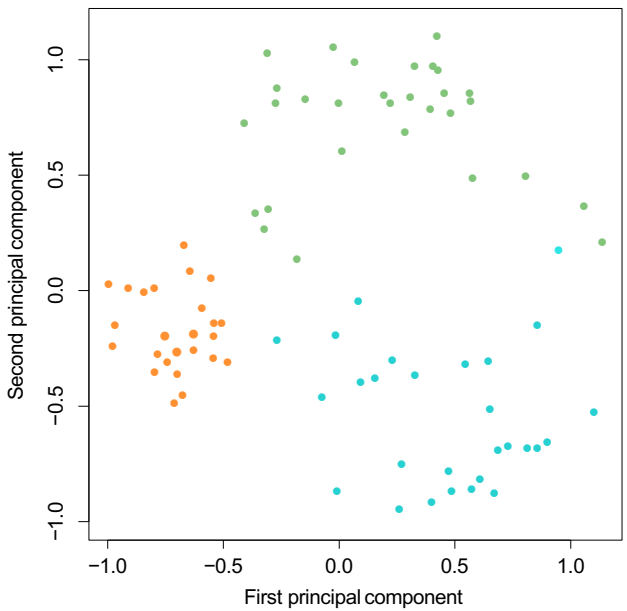
- States with large positive scores on the first component, such as California, Nevada and Florida, have high crime rates
- States like North Dakota, with negative scores on the first component, have low crime rates.

# Figure details

- California has a high score on the second component, hence a high level of urbanization
- The opposite is true for states like Mississippi.
- States close to zero on both components, such as Virginia, have approximately average levels of both crime and urbanization.

# Another Interpretation of Principal Components






The first three principal components of a data set span the three-dimensional hyperplane that is closest to the $n$ observations,

# PCA find the hyperplane closest to the observations

- The first principal component loading vector has a very special property: it defines the line in *p*-dimensional space that is *closest* to the *n* observations (using average squared Euclidean distance as a measure of closeness)

# PCA find the hyperplane closest to the observations

- The notion of principal components as the dimensions that are closest to the *n* observations extends beyond just the first principal component.
- For instance, the first two principal components of a data set span the plane that is closest to the *n* observations, in terms of average squared Euclidean distance.

# Scaling of the variables matters

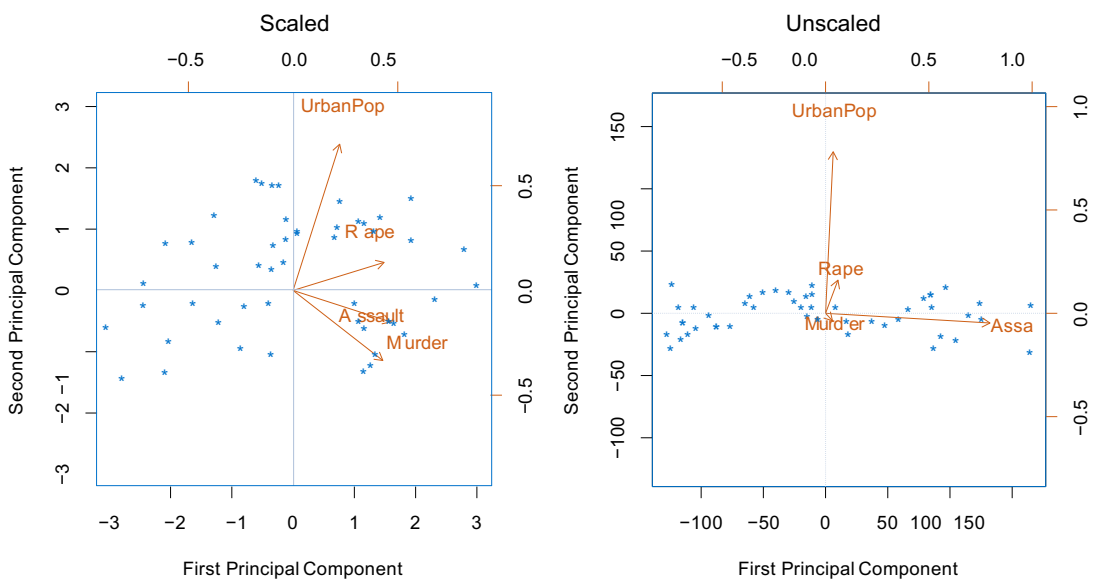
- Murder, Rape, Assault, and UrbanPop have variance 18. 97, 87. 73, 6945. 16, and 209. 5, respectively.
- In PCA on the unscaled variables, the first principal component loading vector will have a very large loading for Assault

# Scaling of the variables matters

- This is simply a consequence of the scales of the variables.
  - For instance, if Assault were measured in units of the number of occurrences per 100 people (rather than number of occurrences per 100,000 people), then this would amount to dividing all of the elements of that variable by 1,000. Then the variance of the variable would be tiny

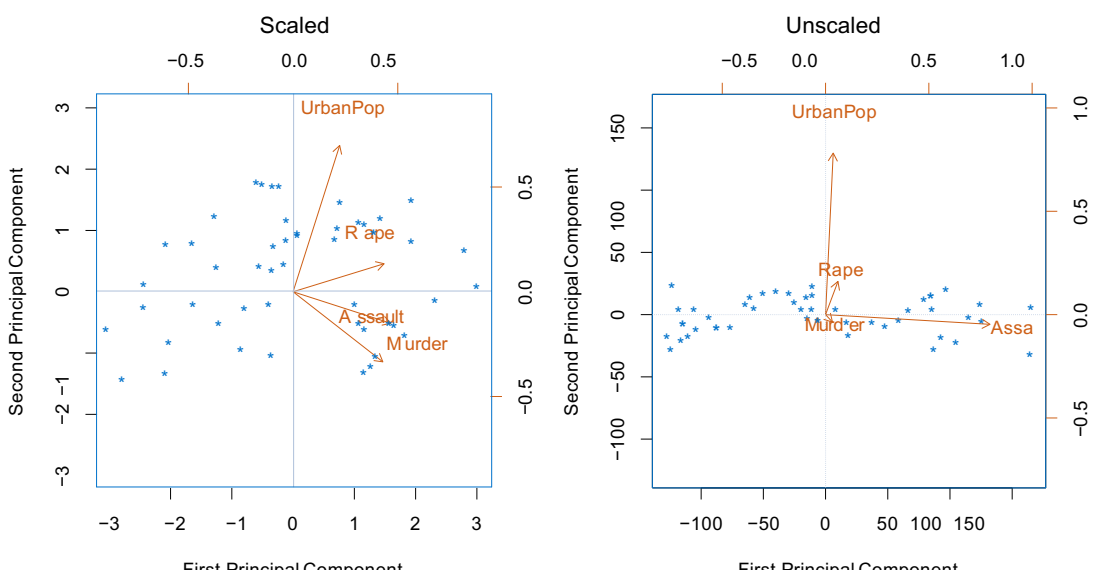
# Scaling of the variables matters

- It is undesirable for the principal components obtained to depend on an arbitrary choice of scaling
- We typically scale each variable to have standard deviation one before we perform PCA.



# Scaling of the variables matters

- If the variables are in different units, scaling each to have standard deviation equal to one is recommended.
- If they are in the same units, you might or might not scale the variables.



# Proportion Variance Explained

- We can now ask a natural question: how much of the information in a given data set is lost by projecting the observations onto the first few principal components?
- That is, how much of the variance in the data is not contained in the first few principal components?

# Proportion Variance Explained

- To understand the strength of each component, we are interested in knowing the proportion of variance explained (PVE) by each one.
- The *total variance* present in a data set (assuming that the variables have been centered to have mean zero) is defined as

$$\sum_{j=1}^{p} \mathrm{Var}(X_j) = \sum_{j=1}^{p} \frac{1}{n} \sum_{i=1}^{n} x_{ij}^2$$

# Proportion Variance Explained

The variance explained by the *m*th principal component is

$$\mathrm{Var}(Z_m) = \frac{1}{n}\sum_{i=1}^{n} z_{im}^2$$

- It can be shown that

$$\sum_{j=1}^{p} \mathrm{Var}(X_j) = \sum_{m=1}^{M} \mathrm{Var}(Z_m)$$

with $M = \min(n - 1, p)$.

# Proportion Variance Explained: continued

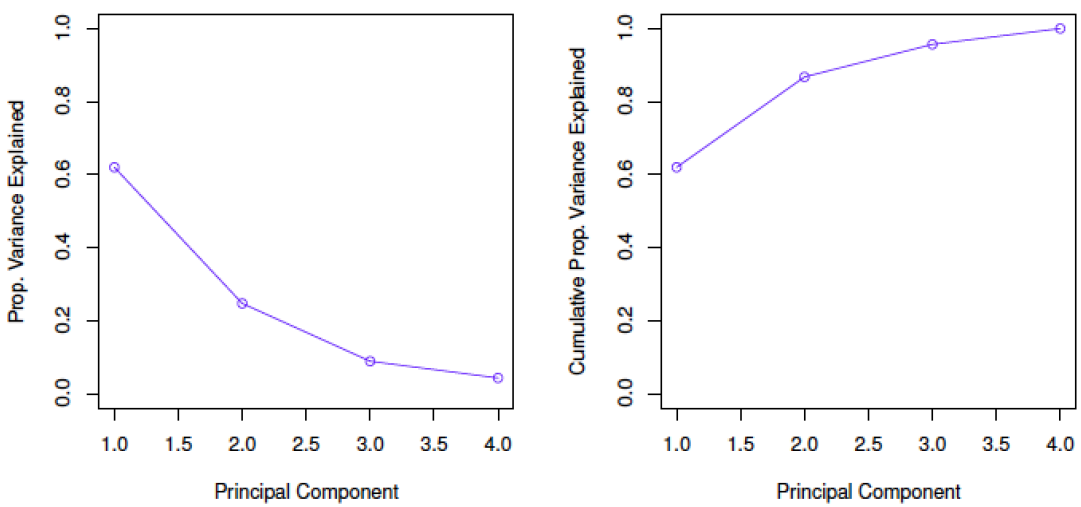
- Therefore, the PVE of the $m^{th}$ principal component is given by the positive quantity between 0 and 1

$$\frac{\sum_{i=1}^{n} z_{im}^2}{\sum_{j=1}^{p} \sum_{i=1}^{n} x_{ij}^2}$$


Principal Component

# Proportion Variance Explained: continued

- The PVEs sum to one. We sometimes display the cumulative PVEs.



# How many principal components should we use?

If we use principal components as a summary of our data, how many components are sufficient?

- No simple answer to this question, as cross-validation is not available for this purpose.
  - *Why not?*

# How many principal components should we use?

- When could we use cross-validation to select the number of components?
  - the “scree plot” on the previous slide can be used as a guide: we look for an “elbow”.

# How many principal components should we use?

- If we compute principal components in a supervised analysis, then we can treat the number of principal component score vectors to be used as a tuning parameter to be selected via cross-validation

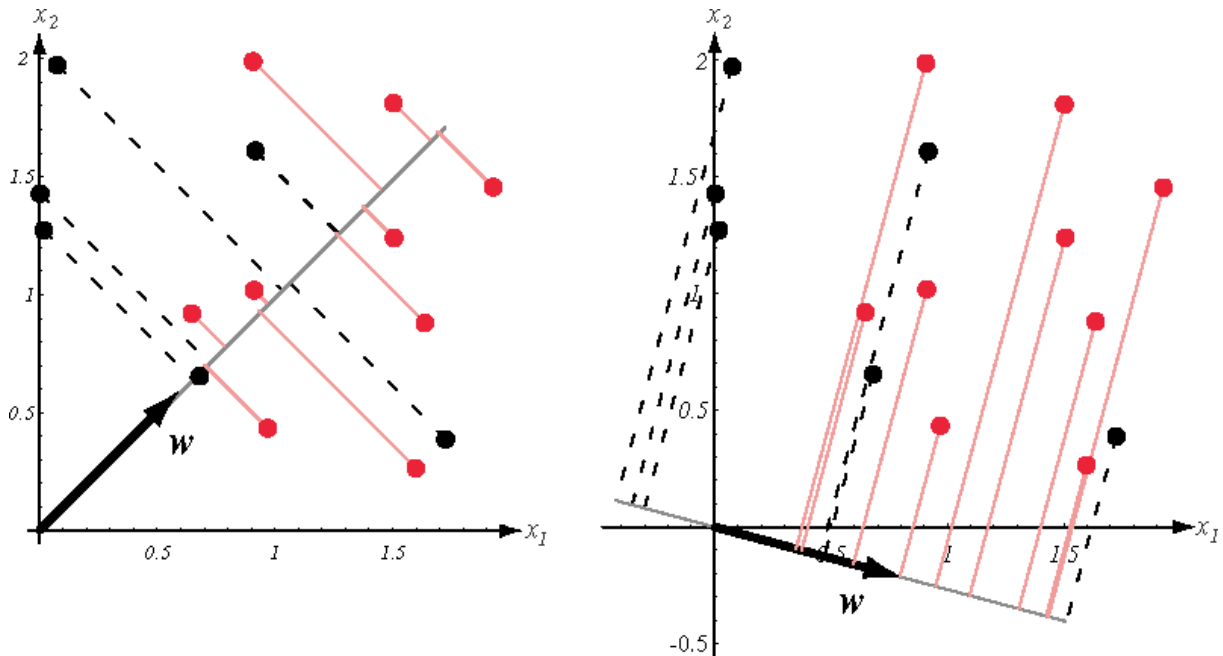
# PCA vs Clustering

- PCA looks for a low-dimensional representation of the observations that explains a good fraction of the variance.
- Clustering looks for homogeneous subgroups among the observations.

# Fisher's Linear Discriminant Analysis

- Whereas PCA seeks directions that are efficient for representation, discriminant analysis seeks directions that are efficient for discrimination.
- It is in some sense a supervised version of PCA.

# Fisher's Linear Discriminant Analysis




Projection of the same set of samples onto two different lines in the directions marked as **β**. The figure on the right shows greater separation between the red and black projected points.

# Fisher's Linear Discriminant Analysis

- Given $\mathbf{x}_1, \ldots, \mathbf{x}_n \in R^p$ divided into two subsets $\mathcal{D}_1$ and $\mathcal{D}_2$ corresponding to the classes 1 and 2, respectively, the goal is to find a projection onto a line defined as

$$y = \boldsymbol{\beta}^T \mathbf{x}$$

where the points corresponding to $\mathcal{D}_1$ and $\mathcal{D}_2$ are well separated.

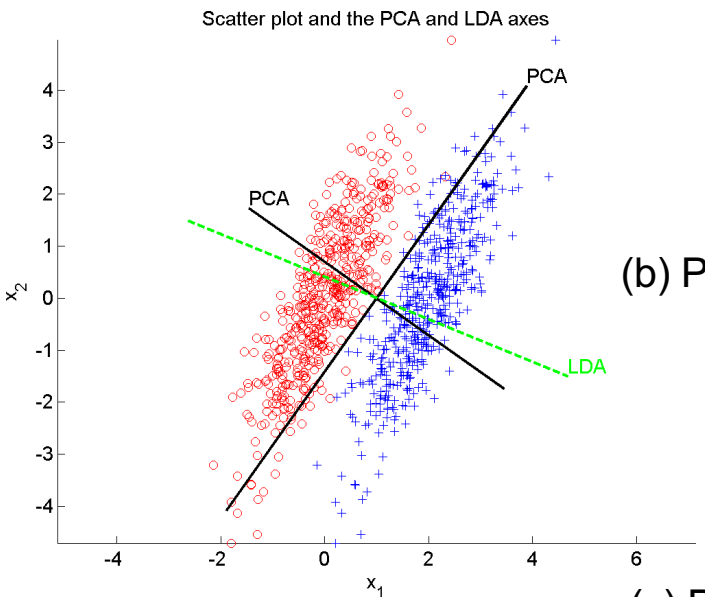
# Fisher's Linear Discriminant Analysis

- This is called the *Fisher's linear discriminant* with the geometric interpretation that the best projection makes the difference between the means as large as possible relative to the variance.

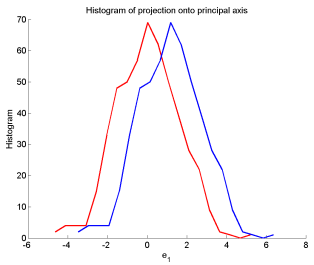
# Fisher's Linear Discriminant Analysis

- Once the transformation from the *p*-dimensional original feature space to a lower dimensional subspace is done using PCA or Fisher's LDA, classification methods can be used to train pertinent classifiers.

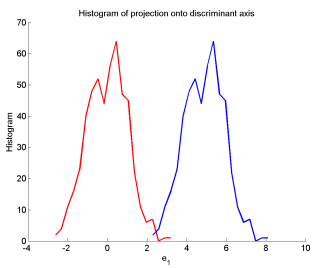
# Fisher's Linear Discriminant Analysis




(a) Scatter plot.



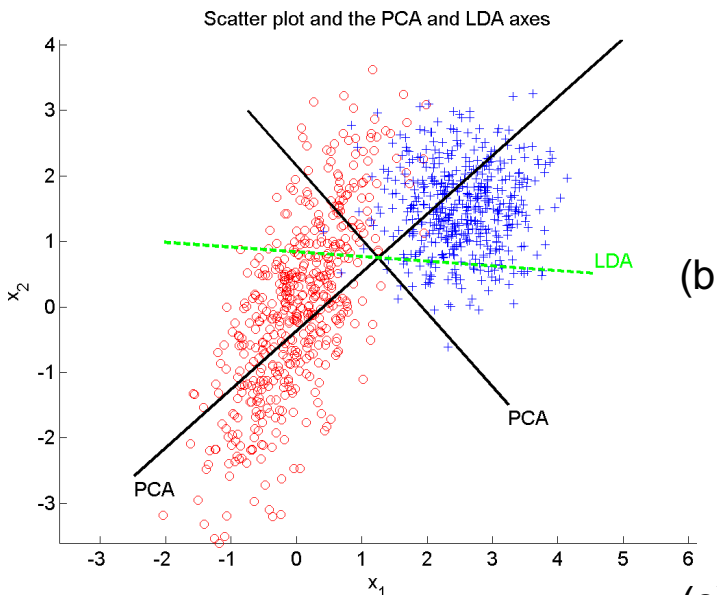

(b) Projection onto the first PCA axis.



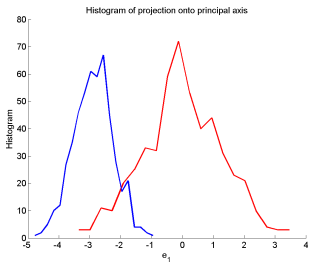

(c) Projection onto the first LDA axis.

Scatter plot and the PCA and LDA axes for a bivariate sample with two classes. Histogram of the projection onto the first LDA axis shows better separation than the projection onto the first PCA axis.

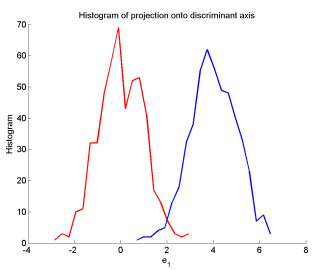
# Fisher's Linear Discriminant Analysis




(a) Scatter plot.




(b) Projection onto the first PCA axis.




(c) Projection onto the first LDA axis.

Scatter plot and the PCA and LDA axes for a bivariate sample with two classes. Histogram of the projection onto the first LDA axis shows better separation than the projection onto the first PCA axis.

# Conclusions

- *Unsupervised learning* is important for understanding the variation and grouping structure of a set of unlabeled data, and can be a useful pre-processor for supervised learning
- It is intrinsically more difficult than *supervised learning* because there is no gold standard (like an outcome variable) and no single objective (like test set accuracy)

# Conclusions

- It is an active field of research, with many recently developed tools such as *self-organizing maps*, *independent components analysis* and *spectral clustering*.

  See *The Elements of Statistical Learning*, chapter 14.

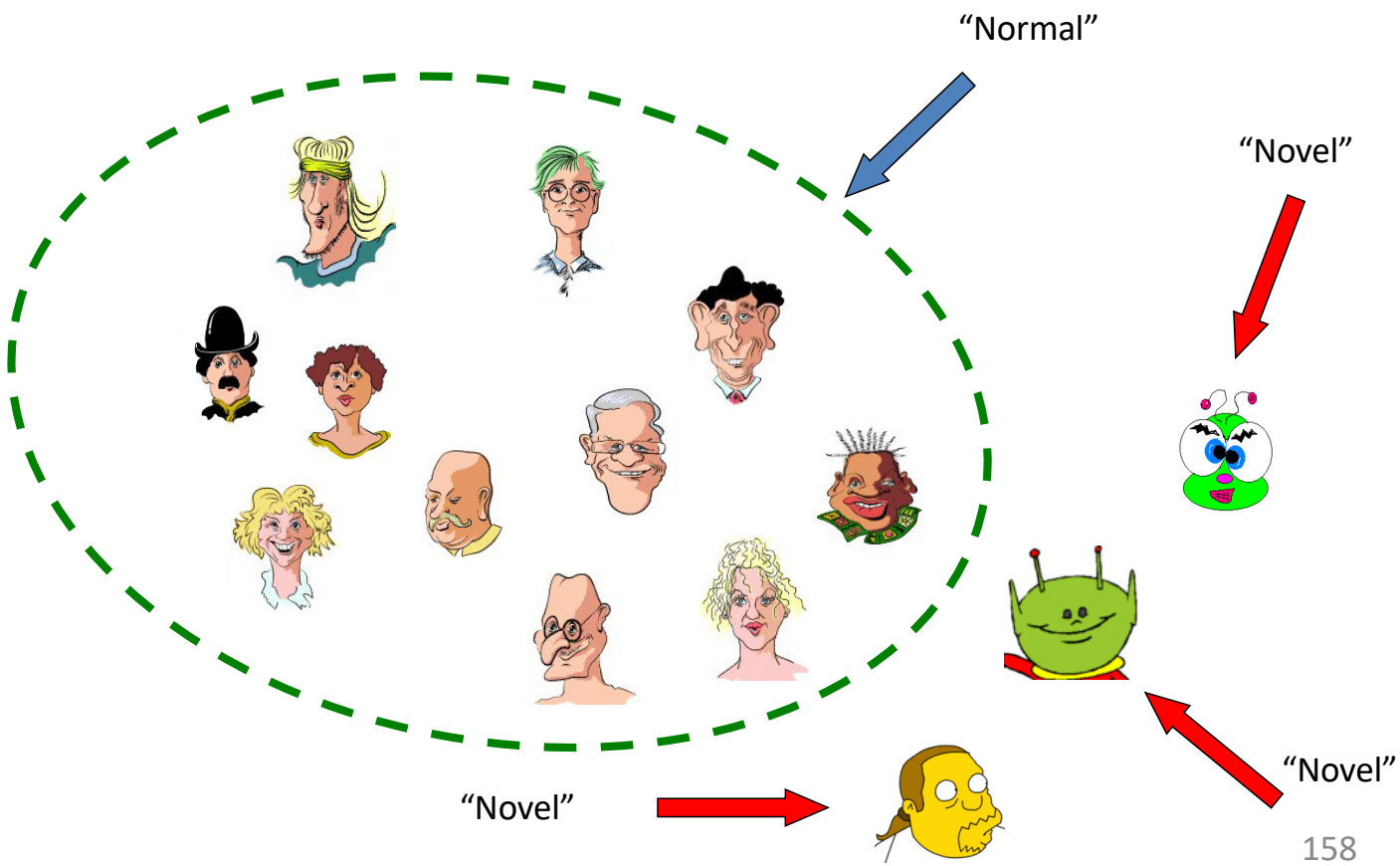
# Appendix 1

Novelty/Anomaly/Outlier Detection

# Novelty Detection is

- An unsupervised learning problem (data unlabeled)
- About the identification of new or unknown data or signal that a machine learning system is not aware of during training

# Example 1



So what's seems to be the problem?

It's a 2-Class problem.
"Normal vs. "Novel"

**Wrong!**

# The Problem is

- That “All positive examples are alike but each negative example is negative in its own way”.

# Example 2

Suppose we want to build a classifier that recognizes web pages about “jigsaw puzzles”.

How can we collect a training data?

We can surf the web and pretty easily assemble a sample to be our collection of positive examples.

# Example 2

What about negative examples ?

The negative examples are… the rest of the web. That is ~(“pickup sticks web page”)

So the negative examples come from an unknown # of negative classes.

# Applications

Many exist

- Intrusion detection
- Fraud detection
- Fault detection
- Robotics
- Medical diagnosis
- E-Commerce
- And more…

# Possible Approaches

Density Estimation:

- Estimate a *density* based on training data
- Threshold the estimated density for test points

Quantile Estimation:

- Estimate a *quantile* of the distribution underlying the training data: for a fixed constant $\alpha \in (0,1]$, attempt to find a small set $S$ such that $\Pr(x \in S) = \alpha$
- Check whether test points are inside or outside $S$

# One Class Support Vector Machine (OCSVM) Algorithm

Maps input data into a high dimensional feature space via kernels

Iteratively finds the maximal margin in the hyperplane which best separates the training data from the origin

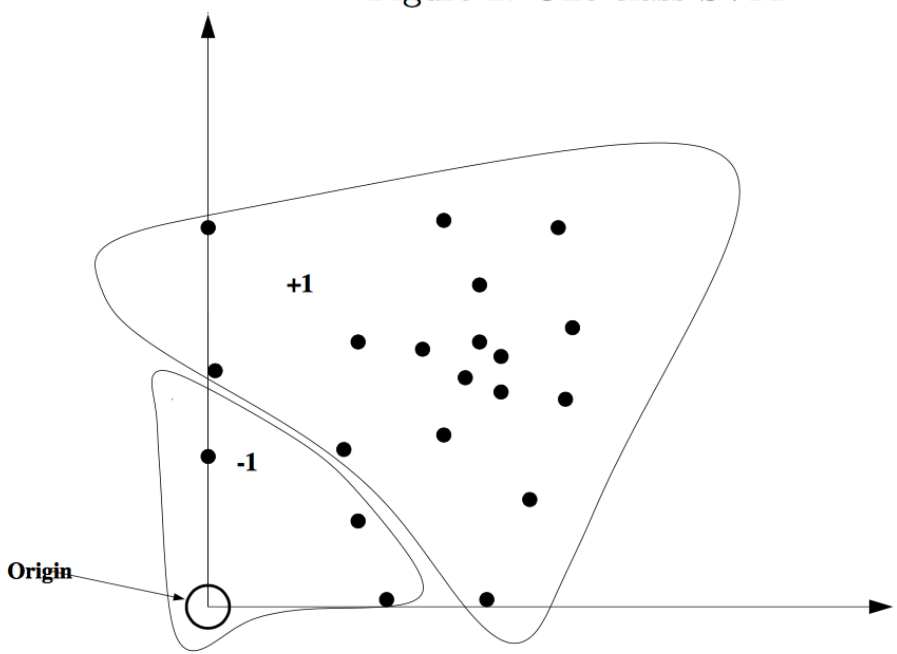
Solves optimization problem to find rule *f* with maximal margin

$f(\mathbf{x})=\beta_0+\beta_1 x_1+\ldots+\beta_p x_p$

If $f(\mathbf{x})<0$ , label **x** as anomalous

# OCSVM

Figure 1: One-class SVM




One-Class SVM Classifier. The origin is the only original member of the second class.

# Efficiency

Complexity of OCSVM

- Time: $O(pL^3)$
- Space: $O(p(L+T))$
  - $p$ – number of dimensions
  - L – number of records in training dataset
  - T – number of records in test dataset