Week 6: SVMs and Unsupervised Learning

Slava Mikhaylov

POLS3003 Data Science and Big Data Analytics

Week 6 Outline

Support Vector Machines

Unsupervised Learning

- Singular-Value Decomposition
- Principal Components Analysis
- Clustering

Support Vector Machines

Support Vector Machines

Here we approach the two-class classification problem in a direct way:
We try and find a plane that separates the classes in feature space.
If we cannot, we get creative in two ways:

- We soften what we mean by "separates", and
- We enrich and enlarge the feature space so that separation is possible.

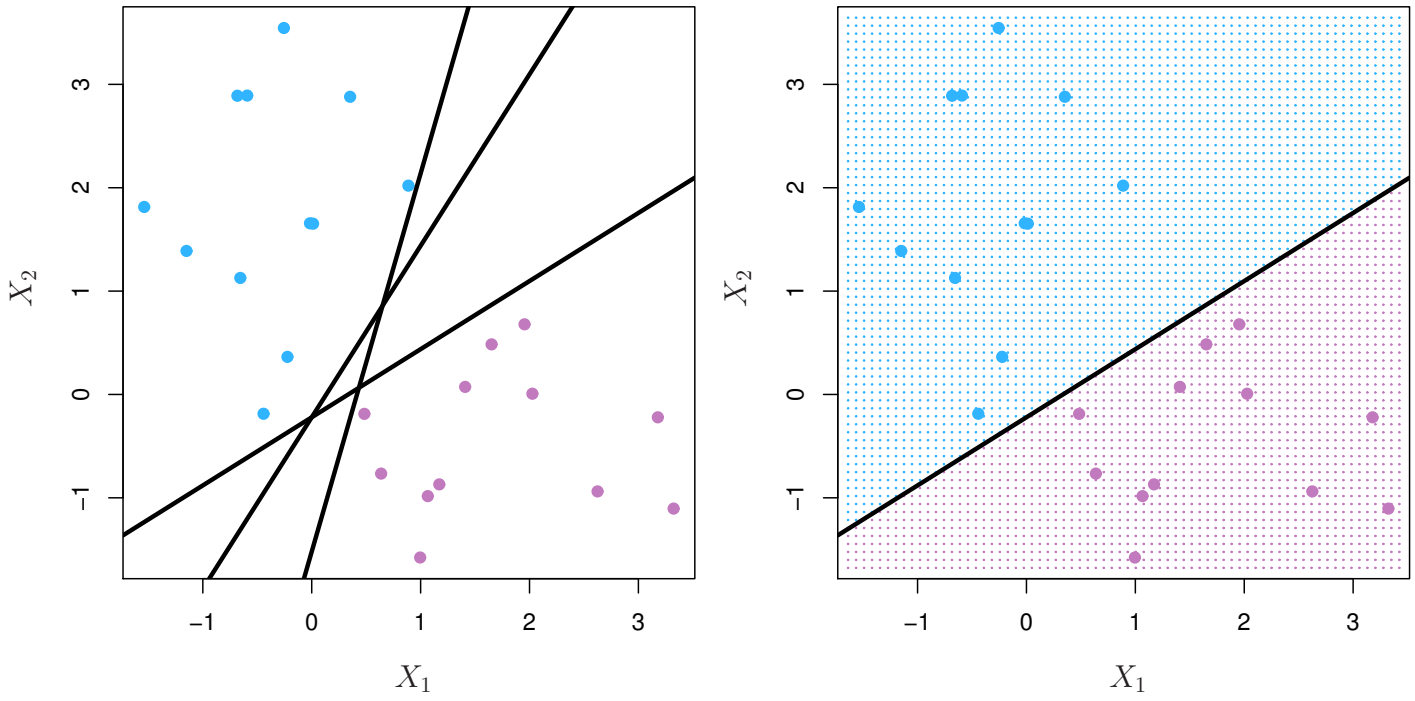
What is a Hyperplane?

- A hyperplane in p dimensions is a flat affine subspace of dimension $p - 1$.
- In general the equation for a hyperplane has the form

$$\beta_0 + \beta_1 X_1 + \beta_2 X_2 + \cdots + \beta_p X_p = 0$$

- In $p = 2$ dimensions a hyperplane is a line.
- If $\beta_0 = 0$, the hyperplane goes through the origin, otherwise not.
- The vector $\beta = (\beta_1, \beta_2, \ldots, \beta_p)$ is called the normal vector – it points in a direction orthogonal to the surface of a hyperplane.

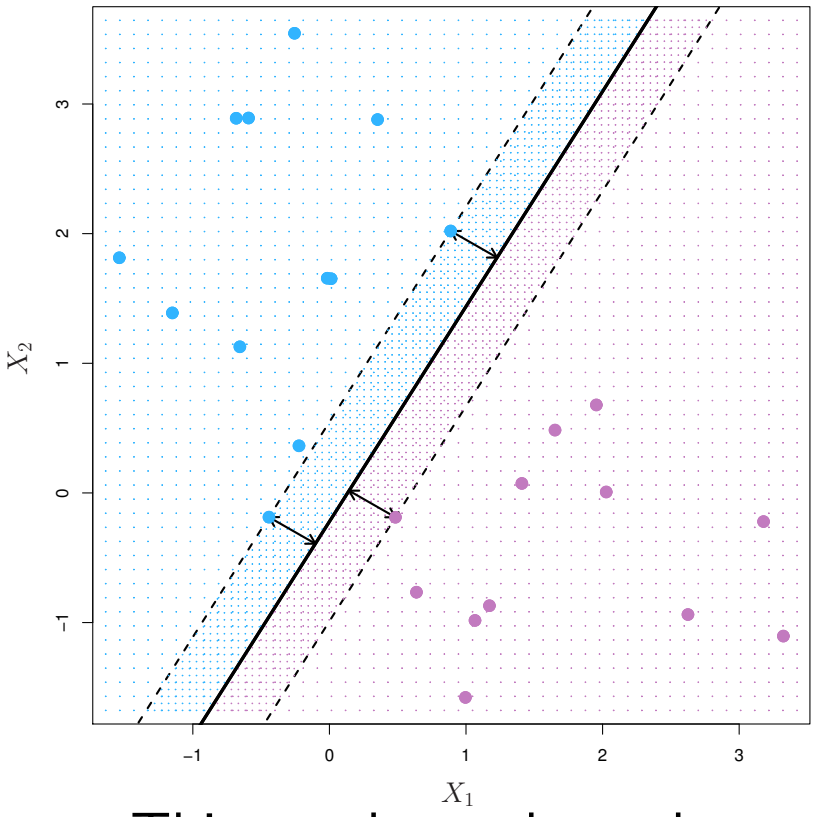
Separating Hyperplanes



- If $f(X) = \beta_0 + \beta_1 X_1 + \cdots + \beta_p X_p$, then $f(X) > 0$ for points on one side of the hyperplane, and $f(X) < 0$ for points on the other.
- If we code the colored points as $Y_i = +1$ for blue, say, and $Y_i = -1$ for purple, then if $Y_i \cdot f(X_i) > 0$ for all i, $f(X) = 0$ defines a separating hyperplane.

Maximal Margin Classifier

Among all separating hyperplanes, find the one that makes the biggest gap or margin between the two classes.

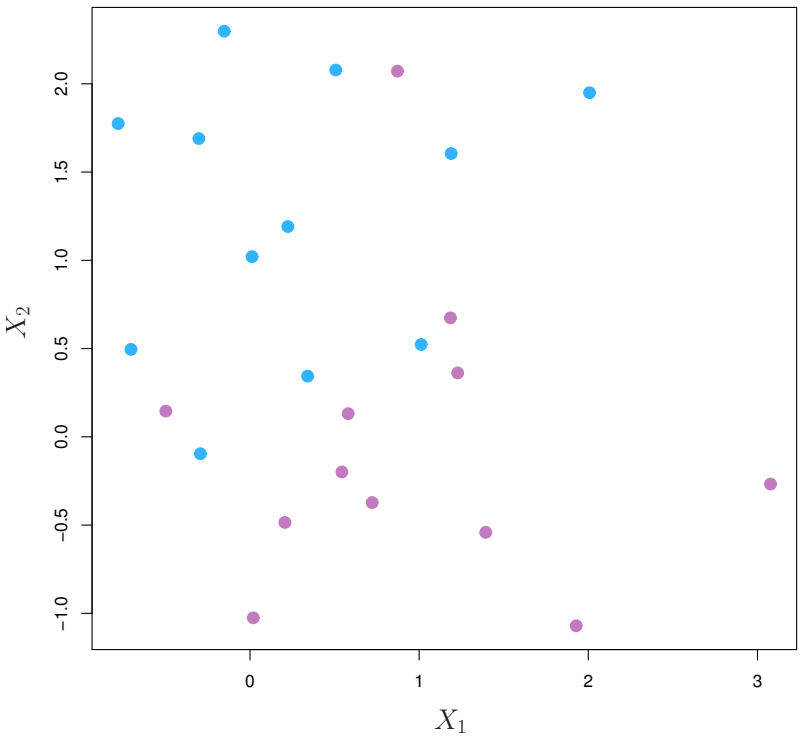


Constrained optimization problem

- $\underbrace{\textit{maximize } M}_{\beta_0,\beta_1,\ldots,\beta_p}$ subject to $\sum_{j=1}^{p} \beta_j^2 = 1$
- $y_i(\beta_0 + \beta_1 x_{i1} + \cdots + \beta_p x_{ip}) \geq M$ for all $i = 1, \ldots, N$.

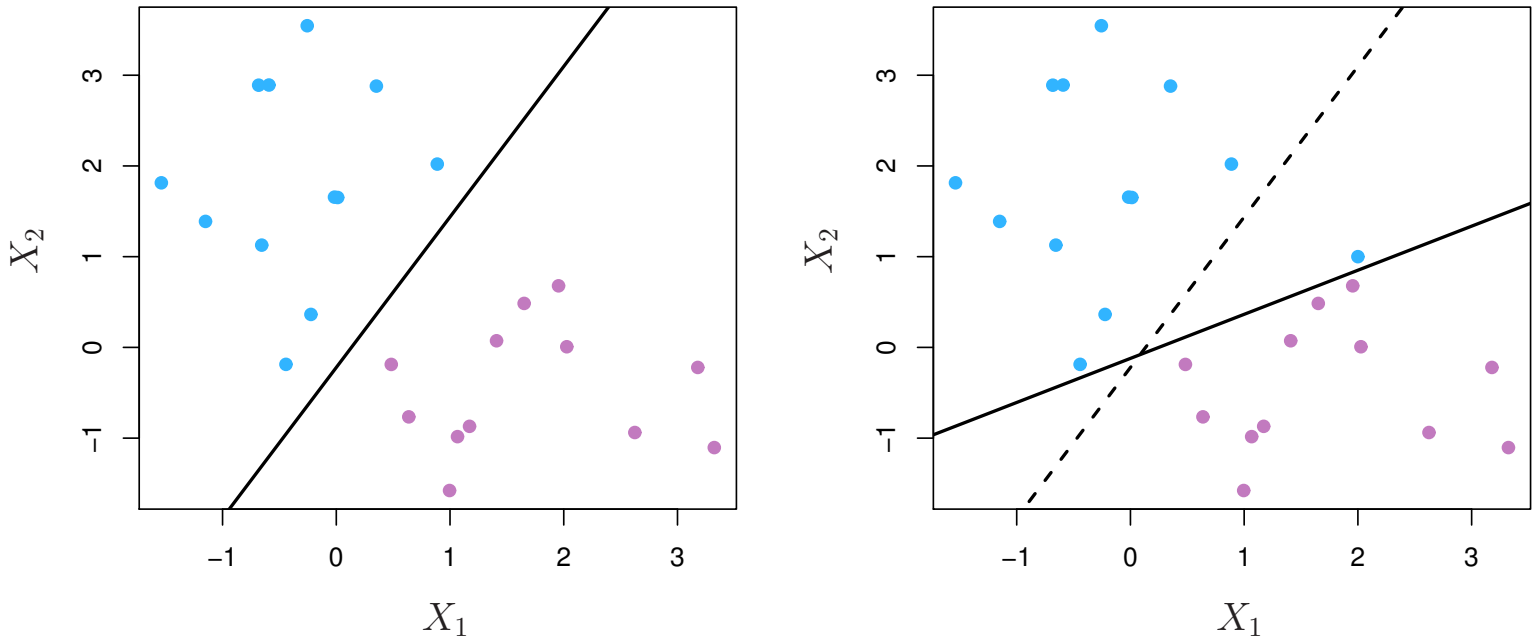
This can be rephrased as a convex quadratic program, and solved efficiently. The function *svm()* in package *e1071* solves this problem efficiently.

Non-separable Data



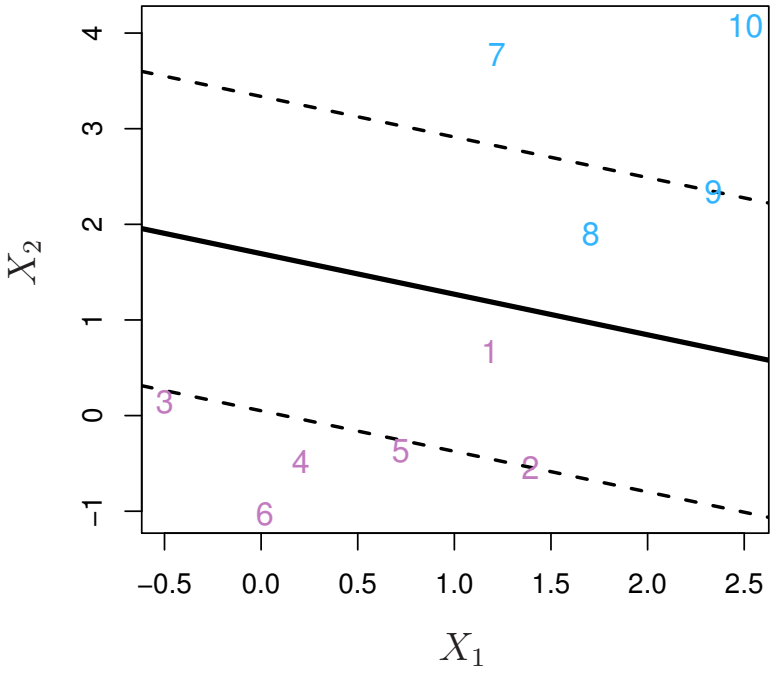
- The data on the left are not separable by a linear boundary.
- This is often the case, unless $N < p$.

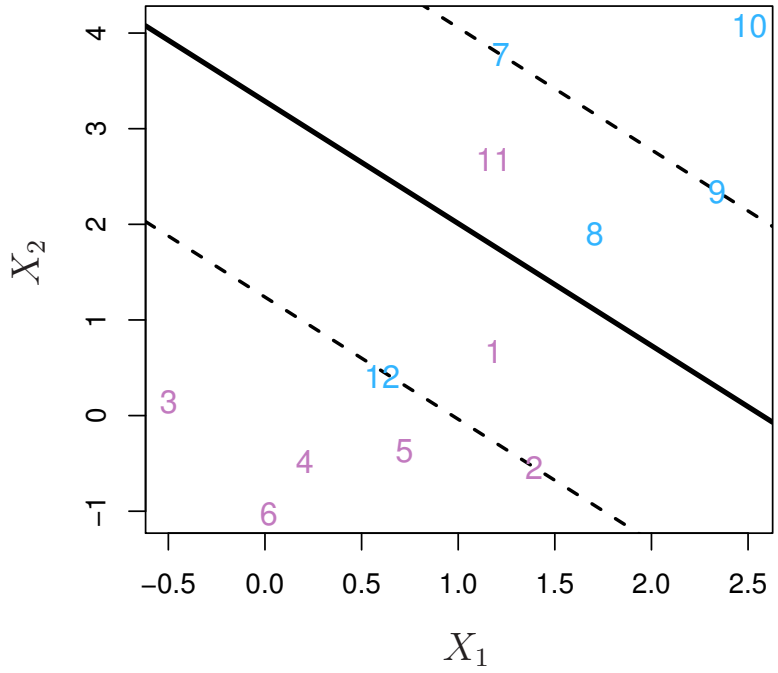
Noisy Data



- Sometimes the data are separable, but noisy.
- This can lead to a poor solution for the maximal-margin classifier.
- The support vector classifier maximizes a soft margin.

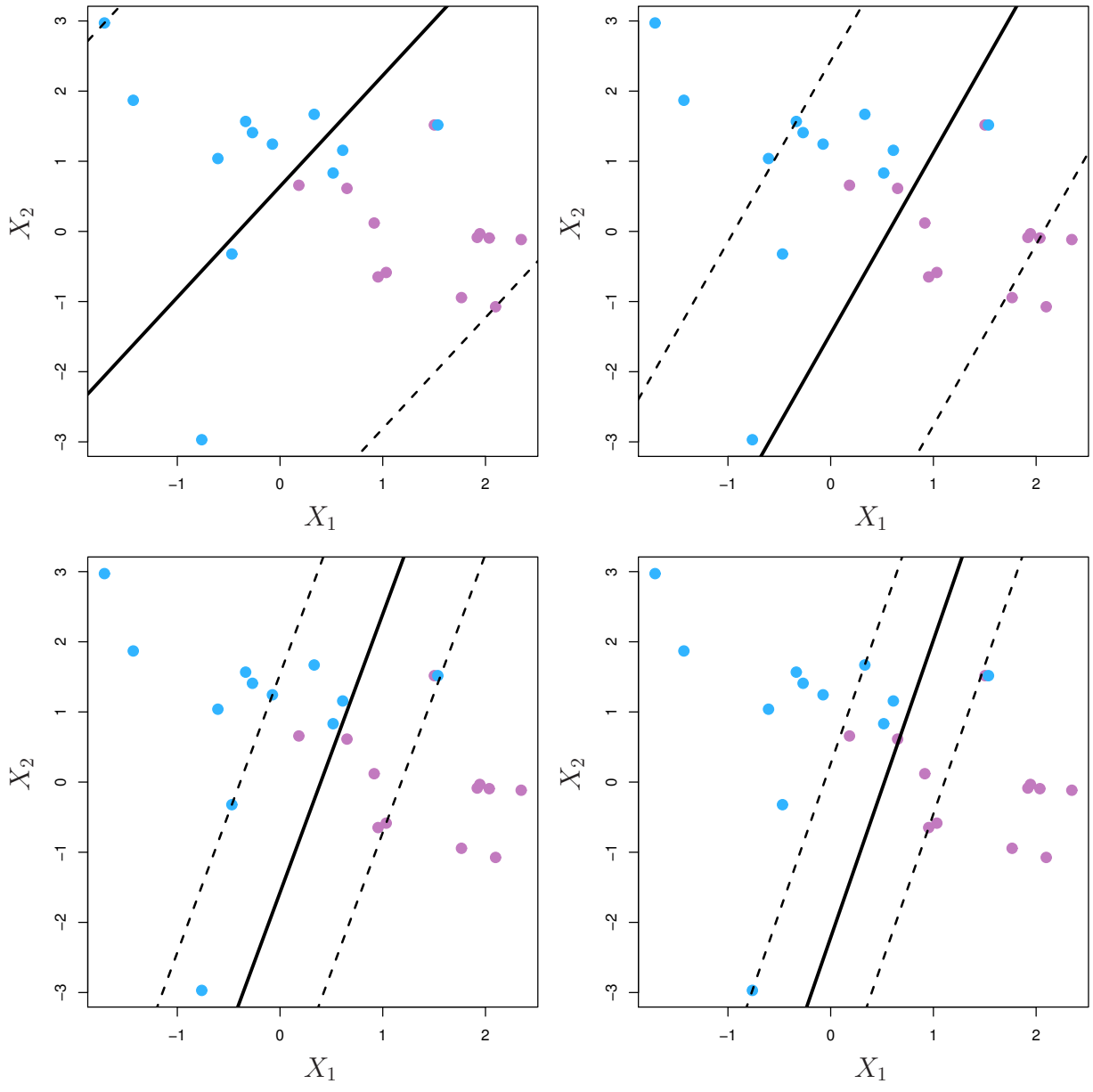
Support Vector Classifier



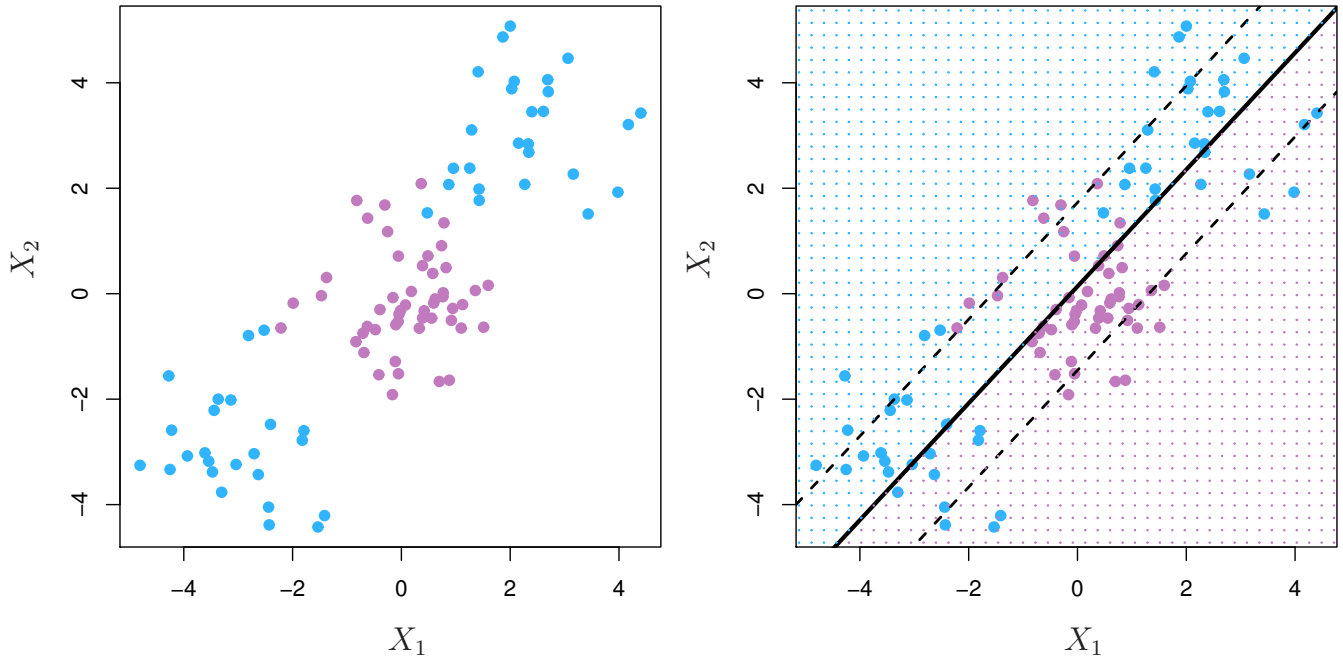


- $\underbrace{\textit{maximize } M}_{\beta_0,\beta_1,\ldots,\beta_p,\epsilon_1,\ldots,\epsilon_n}$ subject to $\sum_{j=1}^{p} \beta_j^2 = 1$
- $y_i(\beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \cdots + \beta_p x_{ip}) \geq M(1 - \epsilon_i)$
- $\epsilon_i \geq 0$, $\sum_{i=1}^{n} \epsilon_i \leq C$.

C is a regularization parameter



Linear boundary can fail



- Sometime a linear boundary simply won't work, no matter what value of C.
- The example on the left is such a case.
- What to do?

Feature Expansion

- Enlarge the space of features by including transformations; e.g. $X_1^2, X_1^3, X_1X_2, X_1X_2^2, \ldots$. Hence go from a p-dimensional space to a $M > p$ dimensional space.
- Fit a support-vector classifier in the enlarged space.
- This results in non-linear decision boundaries in the original space.

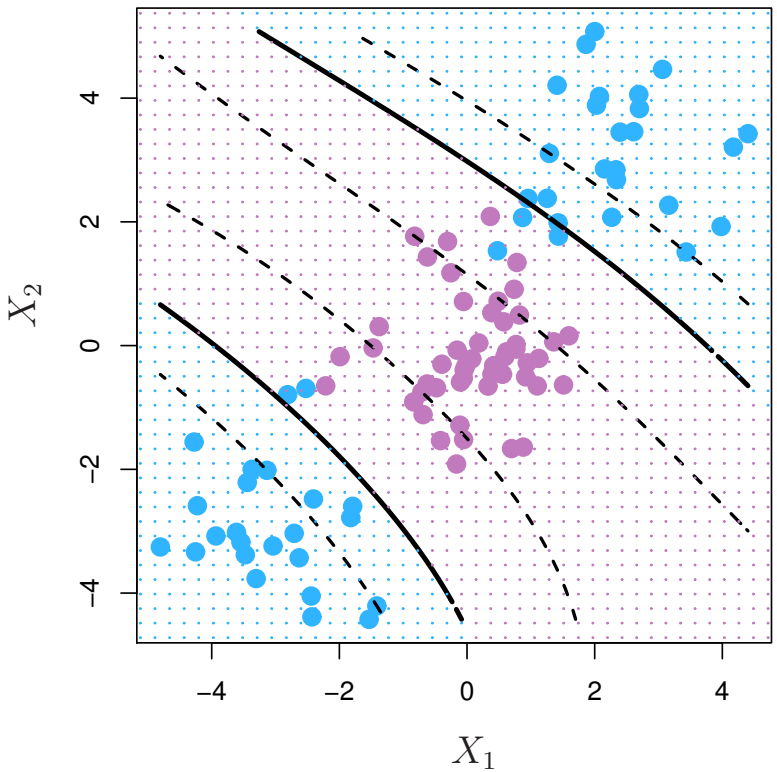
Feature Expansion: Example

- Suppose we use $(X_1, X_2, X_1^2, X_2^2, X_1X_2)$ instead of just (X_1, X_2).
- Then the decision boundary would be of the form

$$\beta_0 + \beta_1 X_1 + \beta_2 X_2 + \beta_3 X_1^2 + \beta_4 X_2^2 + \beta_5 X_1 X_2 = 0$$

- This leads to nonlinear decision boundaries in the original space (quadratic conic sections).

Cubic Polynomials



- Here we use a basis expansion of cubic polynomials.
- From 2 variables to 9
- The support-vector classifier in the enlarged space solves the problem in the lower-dimensional space.

$$\beta_0 + \beta_1 X_1 \beta_2 X_2 + \beta_3 X_1^2 + \beta_4 X_2^2 + \beta_5 X_1 X_2 + \beta_6 X_1^3 + \beta_7 X_2^3 + \beta_8 X_1 X_2^2 + \beta_9 X_1^2 X_2 = 0$$

Nonlinearities and Kernels

- Polynomials (especially high-dimensional ones) get wild rather fast.
- There is a more elegant and controlled way to introduce nonlinearities in support-vector classifiers – through the use of kernels.
- Before we discuss these, we must understand the role of inner products in support-vector classifiers.

Inner products and support vectors

$\langle x_i, x_{i'} \rangle = \sum_{j=1}^{p} x_{ij} x_{i'j}$ – inner product between vectors.

- The linear support vector classifier can be represented as

$$f(x) = \beta_0 + \sum_{i=1}^{n} \alpha_i \langle x, x_i \rangle - \quad n \text{ parameters}$$

- To estimate the parameters $\alpha_1, \ldots, \alpha_n$ and β_0, all we need are the $\binom{n}{2}$ inner products $\langle x_i, x_{i'} \rangle$ between all pairs of training observations.
- It turns out tha most of the $\hat{\alpha}_i$ can be zero:

$$f(x) = \beta_0 + \sum_{i \in S} \hat{\alpha}_i \langle x, x_i \rangle$$

$\mathcal{S}$ is the support set of indices i such that $\hat{\alpha}_i > 0$.

Kernels and Support Vector Machines

- If we can compute inner-products between observations, we can fit a SV classifier.
- Some special kernel functions can do this for us. E.g.

$$K(x_i, x_{i'}) = \left(1 + \sum_{j=1}^{p} x_{ij} x_{i'j}\right)^d$$

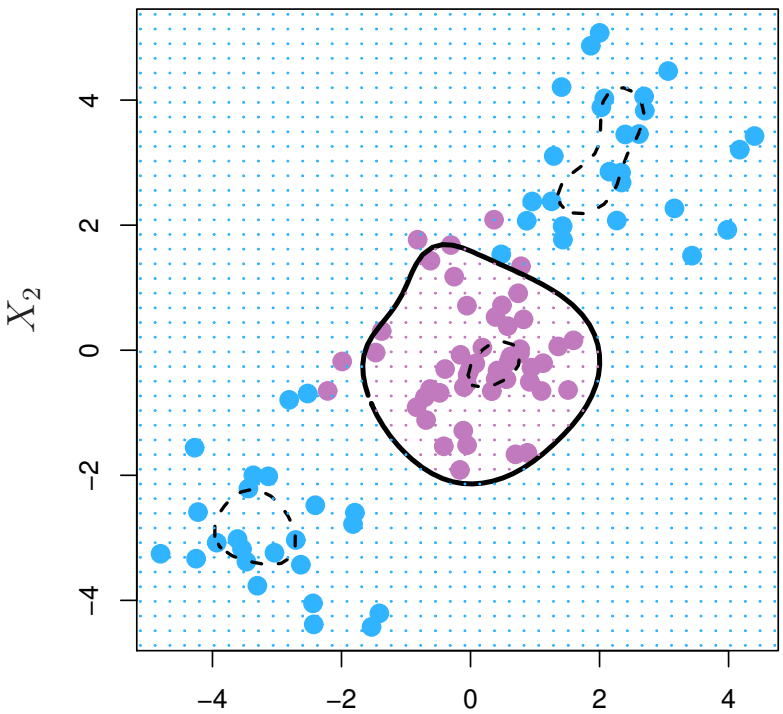
computes the inner-products needed for d dimensional polynomials – $\binom{p+d}{d}$ basis functions.

- The solution has the form

$$f(x) = \beta_0 + \sum_{i \in S} \hat{\alpha}_i K(x, x_i).$$

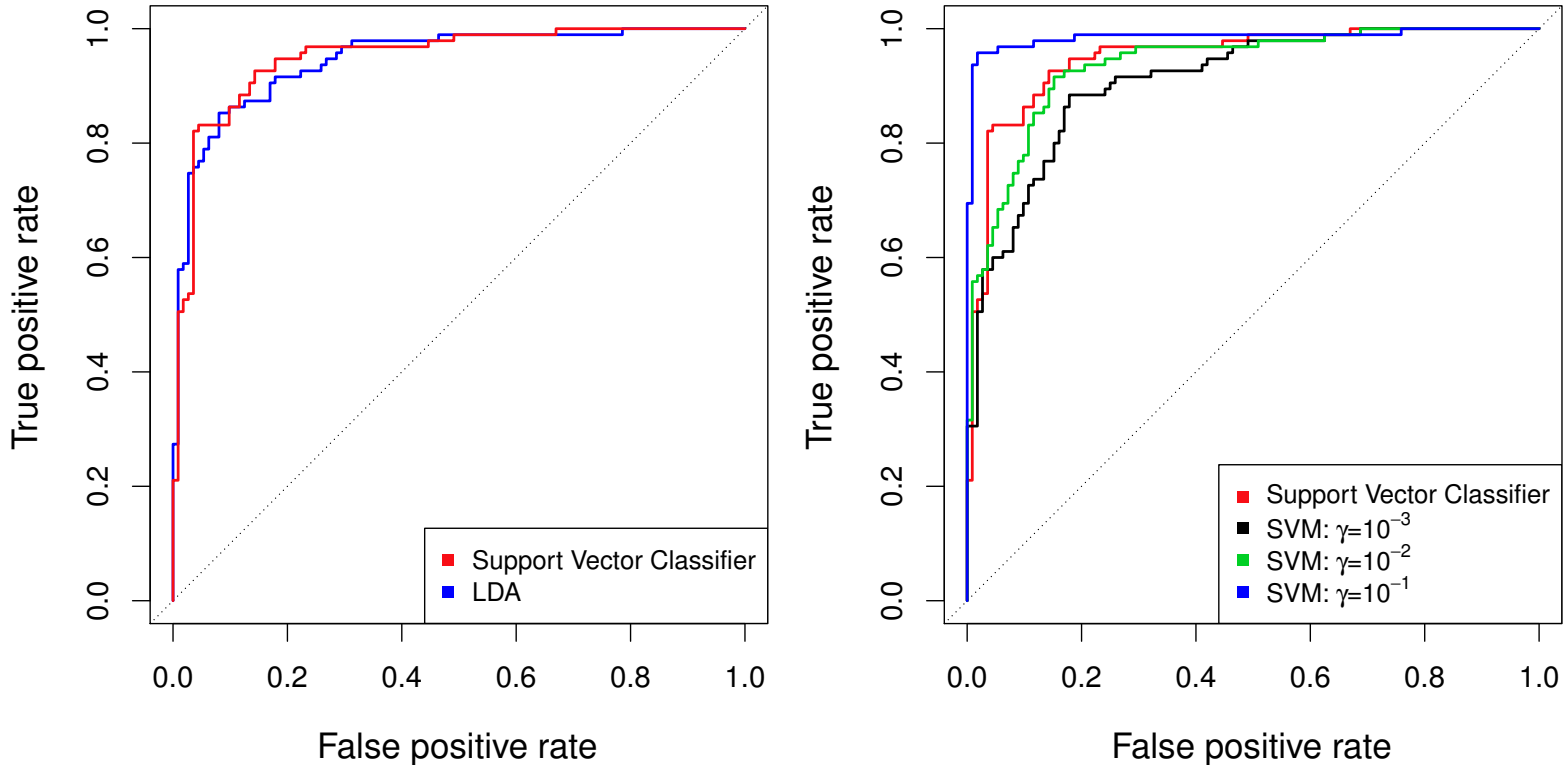
Radial Kernel

$$K(x_i, x_{i'}) = exp(-\gamma \sum_{j=1}^{p} (x_{ij} - x_{i'j})^2).$$



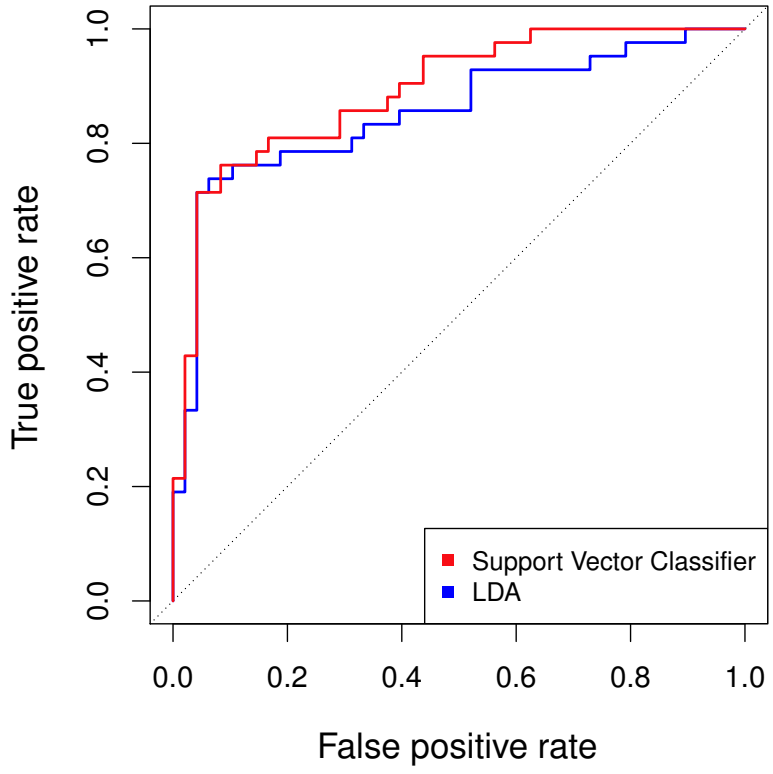
- $f(x) = \beta_0 + \sum_{i \in S} \hat{\alpha}_i K(x, x_i)$
- Implicit feature space; very high dimensional.
- Controls variance by squashing down most dimensions severely.

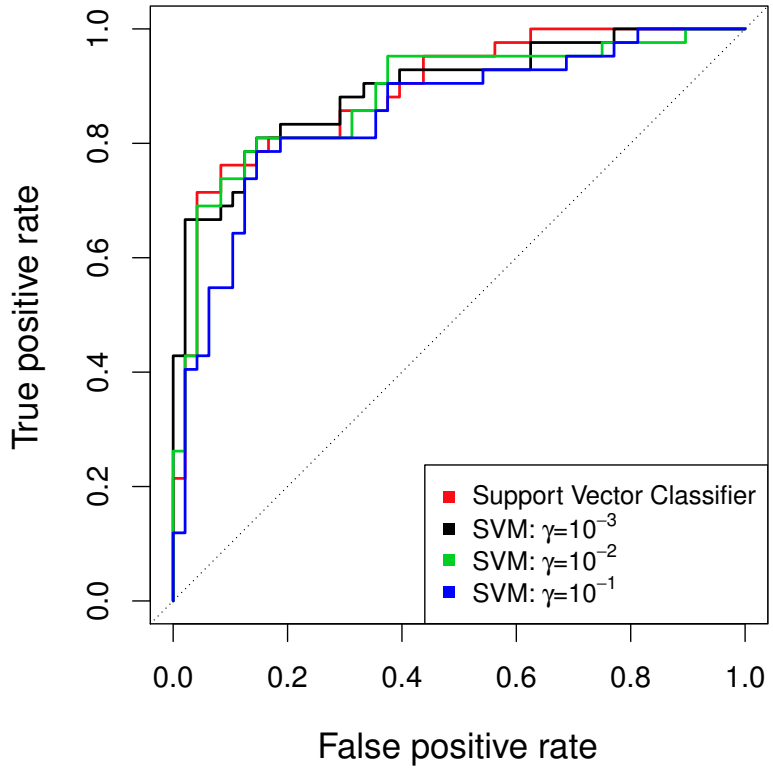
Example: Heart Data



- ROC curve is obtained by changing the threshold 0 to threshold t in $f(X) > t$, and recording false positive and true positive rates as t varies.
- Here we see ROC curves on training data.

Example continued: Heart Test Data





SVMs: more than 2 classes?

The SVM as defined works for $K = 2$ classes. What do we do if we have $K > 2$ classes?

OVA One versus All. Fit K different 2-class SVM classifiers $\hat{f}_k(x),\ k = 1, \ldots, K$; each class versus the rest. Classify x^* to the class for which $\hat{f}_k(x^*)$ is largest.

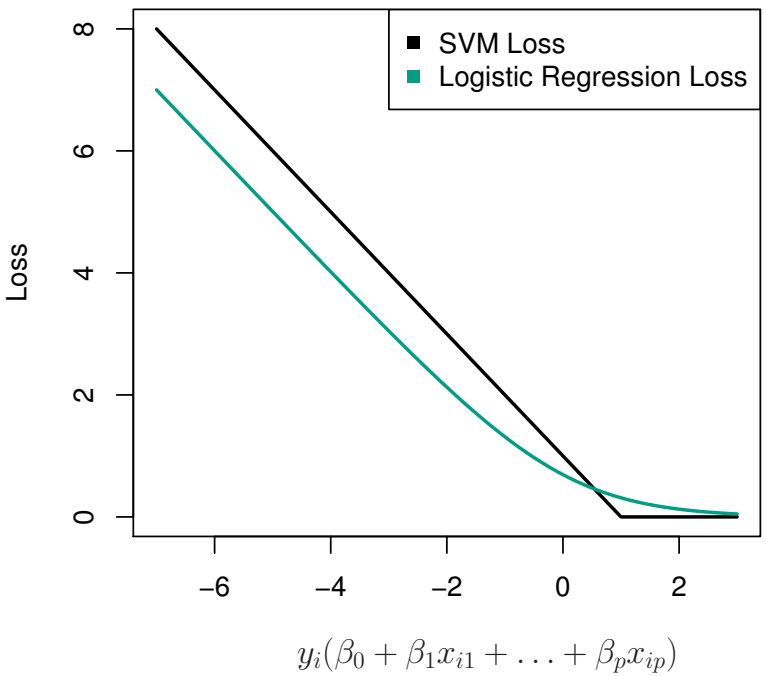
OVO One versus One. Fit all $\binom{K}{2}$ pairwise classifiers $\hat{f}_{k\ell}(x)$. Classify x to the class that wins the most pairwise competitions.

Which to choose? If K is not too large, use OVO.

Support Vector versus Logistic Regression?

With $f(X) = \beta_0 + \beta_1 X_1 + \cdots + \beta_p X_p$ can rephrase support-vector classifier optimization as

$$\underbrace{minimize}_{\beta_0,\beta_1,\ldots,\beta_p} \left\{ \sum_{i=1}^{n} max[0, 1 - y_i f(x_i)] + \lambda \sum_{j=1}^{p} \beta_j^2 \right\}$$



- This has the form loss plus penalty.
- The loss is known as the hinge loss.
- Very similar to "loss" in logistic regression (negative log-likelihood).

Which to use: SVM or Logistic Regression

- When classes are (nearly) separable, SVM does better than LR. So does LDA.
- When not, LR (with ridge penalty) and SVM very similar.
- If you wish to estimate probabilities, LR is the choice.
- For nonlinear boundaries, kernel SVMs are popular. Can use kernels with LR and LDA as well, but computations are more expensive.

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 $X_1, X_2, \ldots, X_p$.
- Here we instead focus on unsupervised learning, where we 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?

Dimensionality reduction

- As we've seen, data can be viewed as a matrix. Very often this matrix is very large. Occasionally it may be too large to fit in your computer memory.
- In many applications, the matrix can be summarized by finding smaller ("narrower") matrices that in some sense are close to the original.
- These narrow matrices have only a small number of rows or a small number of columns, and therefore can be used much more efficiently than the original large matrix.
- The process of finding these narrow matrices is called dimensionality reduction.

Singular-Value Decomposition

- Singular-Value Decomposition (SVD) is a standard technique of matrix analysis that leads to a low-dimensional representation of a high-dimensional matrix.
- SVD allows an exact representation of any matrix. It also makes it easy to eliminate the less important parts of that representation to produce an approximate representation with any desired number of dimensions.

Definition of SVD

- Let M be an $m \times n$ matrix, and let the rank of M be r. Recall that the rank of a matrix is the largest number of rows (or equivalently columns) we can choose for which no nonzero linear combination of the rows is the all-zero vector $\mathbf{0}$ (we say a set of such rows or columns is *independent*).
- Then we can find matrices U, Σ, and V with the following properties:
  1. U is an $m \times r$ column-orthonormal matrix; that is, each of its columns is a unit vector and the dot product of any two columns is 0.
  2. V is an $n \times r$ column-orthonormal matrix. Note that we always use V in its transposed form, so it is the rows of V^T that are orthonormal.
  3. Σ is a diagonal matrix; that is, all elements not on the main diagonal are 0. The elements of Σ are called the singular values of M.

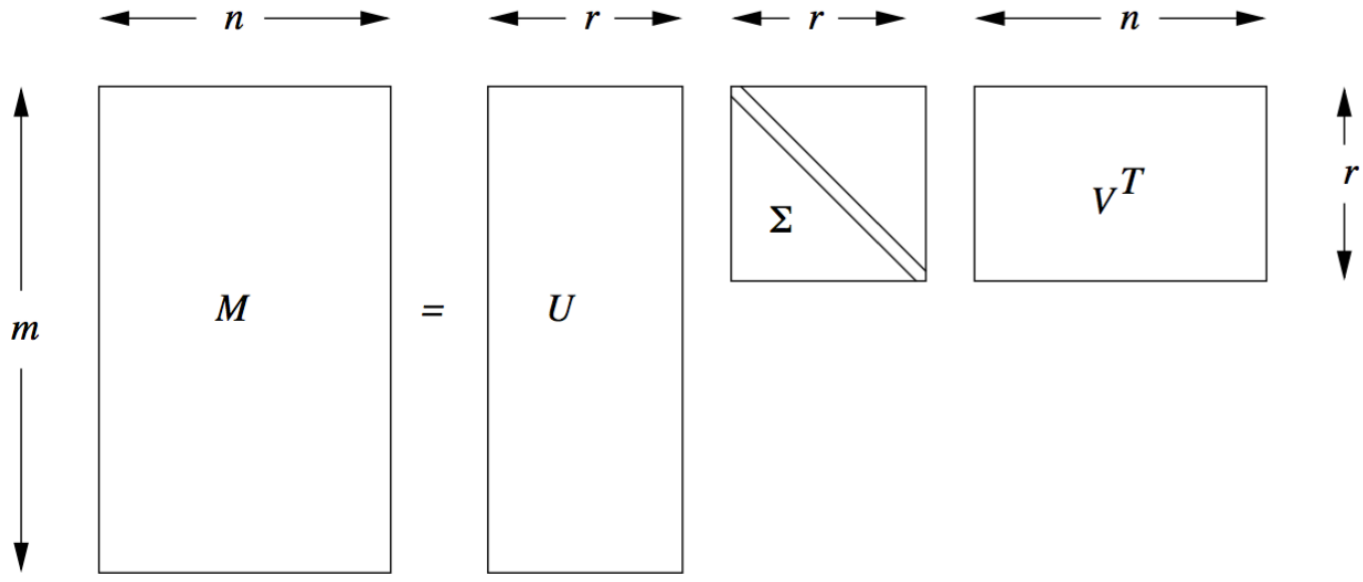


Figure 11.5: The form of a singular-value decomposition

	Matrix	Alien	Star Wars	Casablanca	Titanic
Joe	1	1	1	0	0
Jim	3	3	3	0	0
John	4	4	4	0	0
Jack	5	5	5	0	0
Jill	0	0	0	4	4
Jenny	0	0	0	5	5
Jane	0	0	0	2	2

Figure 11.6: Ratings of movies by users

$$\underbrace{\begin{bmatrix} 1 & 1 & 1 & 0 & 0 \\ 3 & 3 & 3 & 0 & 0 \\ 4 & 4 & 4 & 0 & 0 \\ 5 & 5 & 5 & 0 & 0 \\ 0 & 0 & 0 & 4 & 4 \\ 0 & 0 & 0 & 5 & 5 \\ 0 & 0 & 0 & 2 & 2 \end{bmatrix}}_{M} = \underbrace{\begin{bmatrix} .14 & 0 \\ .42 & 0 \\ .56 & 0 \\ .70 & 0 \\ 0 & .60 \\ 0 & .75 \\ 0 & .30 \end{bmatrix}}_{U} \underbrace{\begin{bmatrix} 12.4 & 0 \\ 0 & 9.5 \end{bmatrix}}_{\Sigma} \underbrace{\begin{bmatrix} .58 & .58 & .58 & 0 & 0 \\ 0 & 0 & 0 & .71 & .71 \end{bmatrix}}_{V^{\mathrm{T}}}$$

Figure 11.7: SVD for the matrix M of Fig. 11.6

Interpreting SVD

- We view r of U, Σ, and V as representing concepts that are hidden in the original matrix M.
- In the example above, these concepts can be viewed as "science fiction" and "romance".
- If we view the rows in M as people and the columns in M as films, then U connects people to concepts.
- V related films to concepts.
- Σ gives the strength of each of the concepts. Concepts may be stronger when the data provides more information about the films of that genre and the people who like them.
- In general, the concepts will not be clearly delineated. In practice, we won't be able to get the exact decomposition.

$$\begin{bmatrix} 1 & 1 & 1 & 0 & 0 \\ 3 & 3 & 3 & 0 & 0 \\ 4 & 4 & 4 & 0 & 0 \\ 5 & 5 & 5 & 0 & 0 \\ 0 & 2 & 0 & 4 & 4 \\ 0 & 0 & 0 & 5 & 5 \\ 0 & 1 & 0 & 2 & 2 \end{bmatrix} =$$

M'

$$\begin{bmatrix} .13 & .02 & -.01 \\ .41 & .07 & -.03 \\ .55 & .09 & -.04 \\ .68 & .11 & -.05 \\ .15 & -.59 & .65 \\ .07 & -.73 & -.67 \\ .07 & -.29 & .32 \end{bmatrix} \begin{bmatrix} 12.4 & 0 & 0 \\ 0 & 9.5 & 0 \\ 0 & 0 & 1.3 \end{bmatrix} \begin{bmatrix} .56 & .59 & .56 & .09 & .09 \\ .12 & -.02 & .12 & -.69 & -.69 \\ .40 & -.80 & .40 & .09 & .09 \end{bmatrix}$$

U Σ V^{T}

Figure 11.9: SVD for the matrix M' of Fig. 11.8

Dimensionality reduction using SVD

- Suppose we want to represent a very large matrix M by its SVD components U, Σ, and V. But these matrices are also too large to store conveniently.
- The best way to reduce the dimensionality of the three matrices is to set the smallest of the singular values to zero.
- If we set the s smallest singular values to 0, then we can also eliminate the corresponding s rows of U and V.

$$\begin{bmatrix} .13 & .02 \\ .41 & .07 \\ .55 & .09 \\ .68 & .11 \\ .15 & -.59 \\ .07 & -.73 \\ .07 & -.29 \end{bmatrix} \begin{bmatrix} 12.4 & 0 \\ 0 & 9.5 \end{bmatrix} \begin{bmatrix} .56 & .59 & .56 & .09 & .09 \\ .12 & -.02 & .12 & -.69 & -.69 \end{bmatrix}$$

$$= \begin{bmatrix} 0.93 & 0.95 & 0.93 & .014 & .014 \\ 2.93 & 2.99 & 2.93 & .000 & .000 \\ 3.92 & 4.01 & 3.92 & .026 & .026 \\ 4.84 & 4.96 & 4.84 & .040 & .040 \\ 0.37 & 1.21 & 0.37 & 4.04 & 4.04 \\ 0.35 & 0.65 & 0.35 & 4.87 & 4.87 \\ 0.16 & 0.57 & 0.16 & 1.98 & 1.98 \end{bmatrix}$$

Figure 11.10: Dropping the lowest singular value from the decomposition of Fig. 11.7

The resulting matrix is very close to the matrix M' above.

Decomposition of sparse matrices

- It is normal that the matrix M is sparse – most entries are 0. For example, when we have documents as rows and words as columns most words are not present in most documents. Similarly, a matrix of customers and products – most people do not buy most products.
- We cannot deal with dense matrices that have millions or billions of rows and/or columns.
- However, with SVD, even if M is sparse, U and V will be dense.
- Since Σ is diagonal, it will be sparse, but Σ is usually much smaller than U and V, so its sparseness does not help.

CUR Decomposition

- In CUR-decomposition, if M is sparse, then the two large matrices (C and R for columns and rows) analogous to U and V in SVD are also sparse.
- Only the matrix in the middle (analogous to Σ in SVD) is dense, but this matrix is small so the density does not hurt too much.
- Unlike SVD, which gives an exact decomposition as long as the parameter r is taken to be at least as great as the rank of the matrix M, CUR-decomposition is an approximation no matter how large we make r.
- However, a decomposition with a relatively small value of r has a good probability of being a useful and accurate decomposition.

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

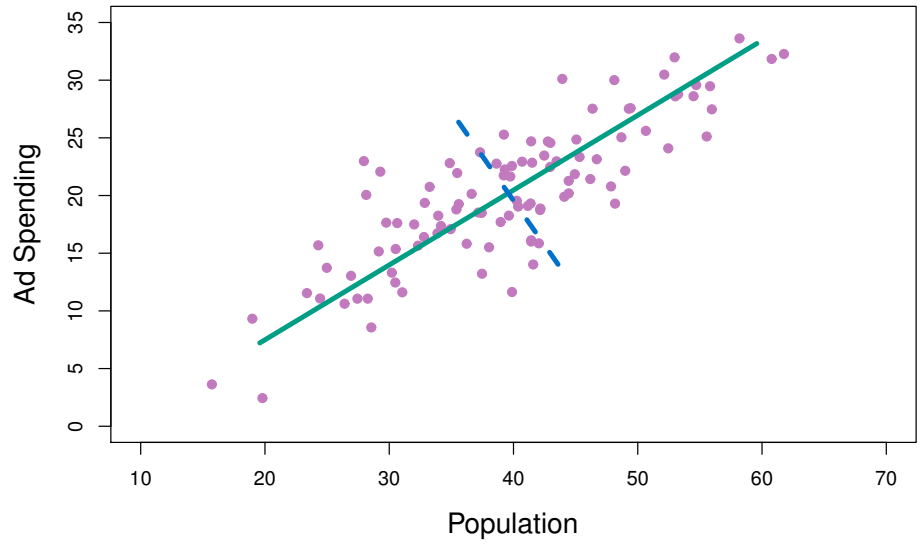
- The first principal component of a set of features $X_1, X_2, \ldots, X_p$ is the normalized linear combination of the features

$$Z_1 = \phi_{11} X_1 + \phi_{21} X_2 + \cdots + \phi_{p1} X_p$$

that has the largest variance. By normalized, we mean that $\sum_{j=1}^{p} \phi_{j1}^2 = 1$.

- We refer to the elements $\phi_{11}, \ldots, \phi_{p1}$ as the loadings of the first principal component; together, the loadings make up the principal component loading vector, $\phi_1 = (\phi_{11}\ \phi_{21} \ldots \phi_{p1})^T$.
- 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, and the blue dashed line indicates the second principal component.

Computation of Principal Components

- Suppose we have a $n \times p$ data set $\mathbf{X}$. Since we are only interested in variance, we assume that each of the variables in $\mathbf{X}$ has been centered to have mean zero (that is, the column means of $\mathbf{X}$ are zero).
- We then look for the linear combination of the sample feature values of the form

$$z_i 1 = \phi_{11} x_{i1} + \phi_{21} x_{i2} + \cdots + \phi_{p1} x_{ip} \tag{1}$$

  for $i = 1, \ldots, n$ that has largest sample variance, subject to the constraint that $\sum_{j=1}^{p} \phi_{j1}^2 = 1$.
- Since each of the x_{ij} has mean zero, then so does z_{i1} (for any values of ϕ_{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

- Plugging in (1) the first principal component loading vector solves the optimization problem

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

- This problem can be solved via a singular-value decomposition of the matrix $\mathbf{X}$, 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 ϕ_1 with elements $\phi_{11}, \phi_{21}, \ldots, \phi_{p1}$ 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

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

where ϕ_2 is the second principal component loading vector, with elements $\phi_{12}, \phi_{22}, \ldots, \phi_{p2}$.

Further principal components: continued

- It turns out that constraining Z_2 to be uncorrelated with Z_1 is equivalent to constraining the direction ϕ_2 to be orthogonal (perpendicular) to the direction ϕ_1. And so on.
- The principal component directions $\phi_1, \phi_2, \phi_3, \ldots$ are the ordered sequence of right singular vectors of the matrix $\mathbf{X}$, and the variances of the components are $\frac{1}{n}$ 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).
- 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

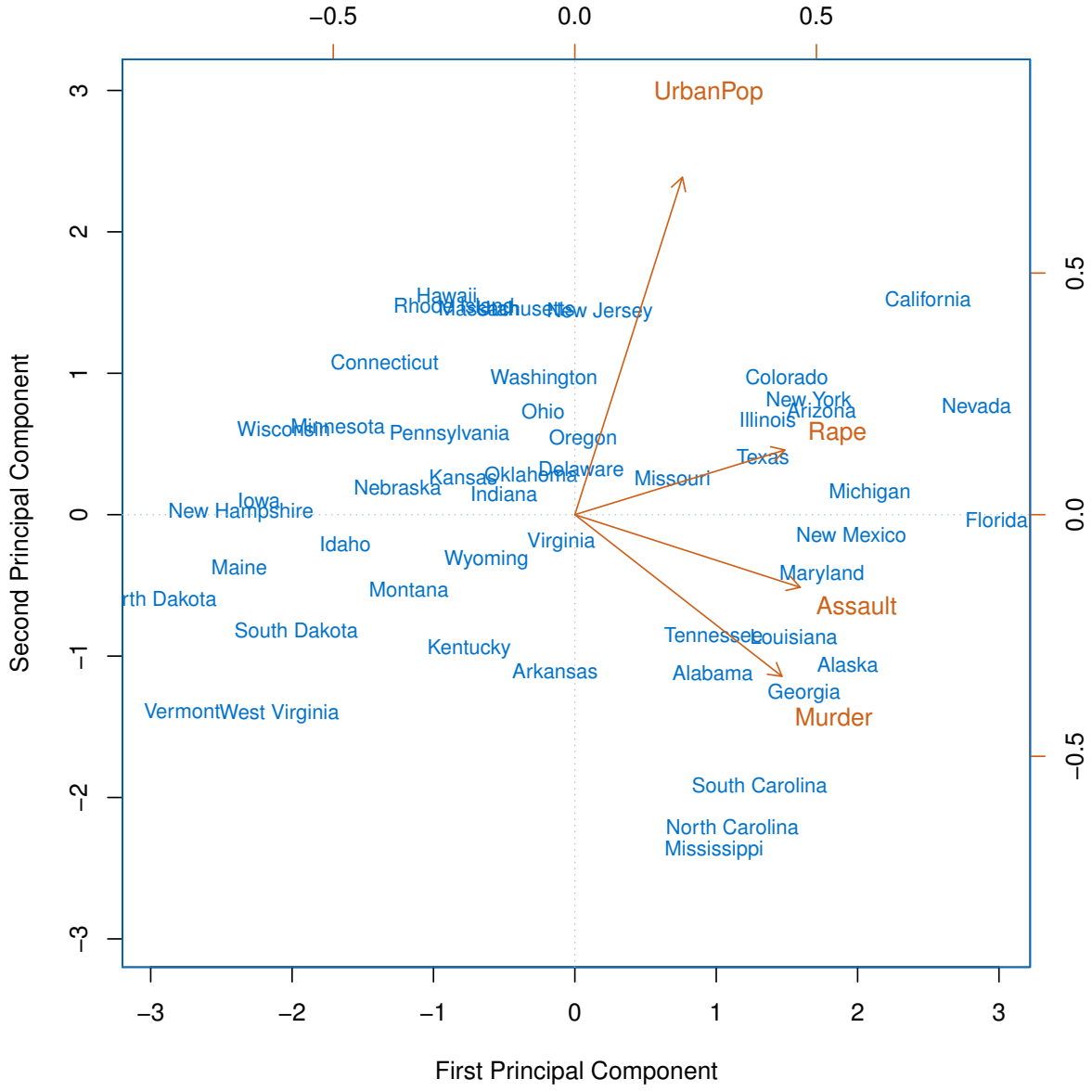


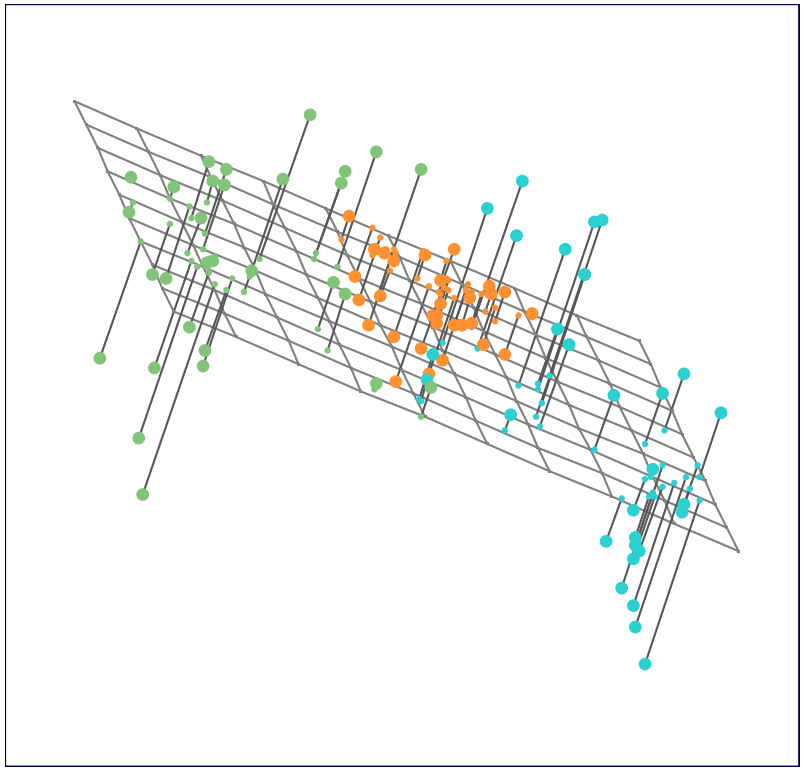
Figure details

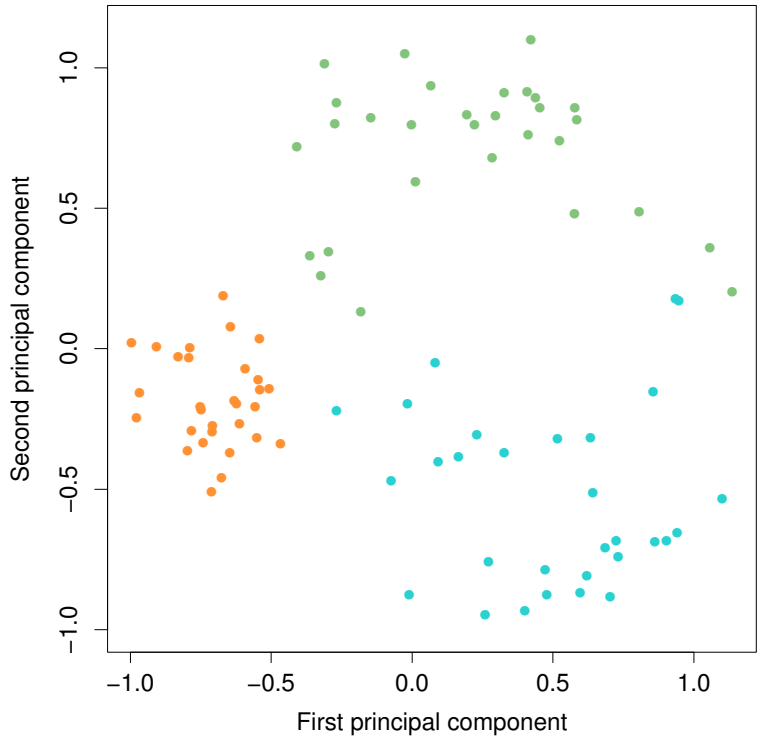
The first two principal components for the USArrests data.

- The blue state names represent the scores for the first two principal components.
- 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)].
- This figure is known as a biplot, because it displays both the principal component scores and the principal component loadings.

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

Another Interpretation of Principal Components



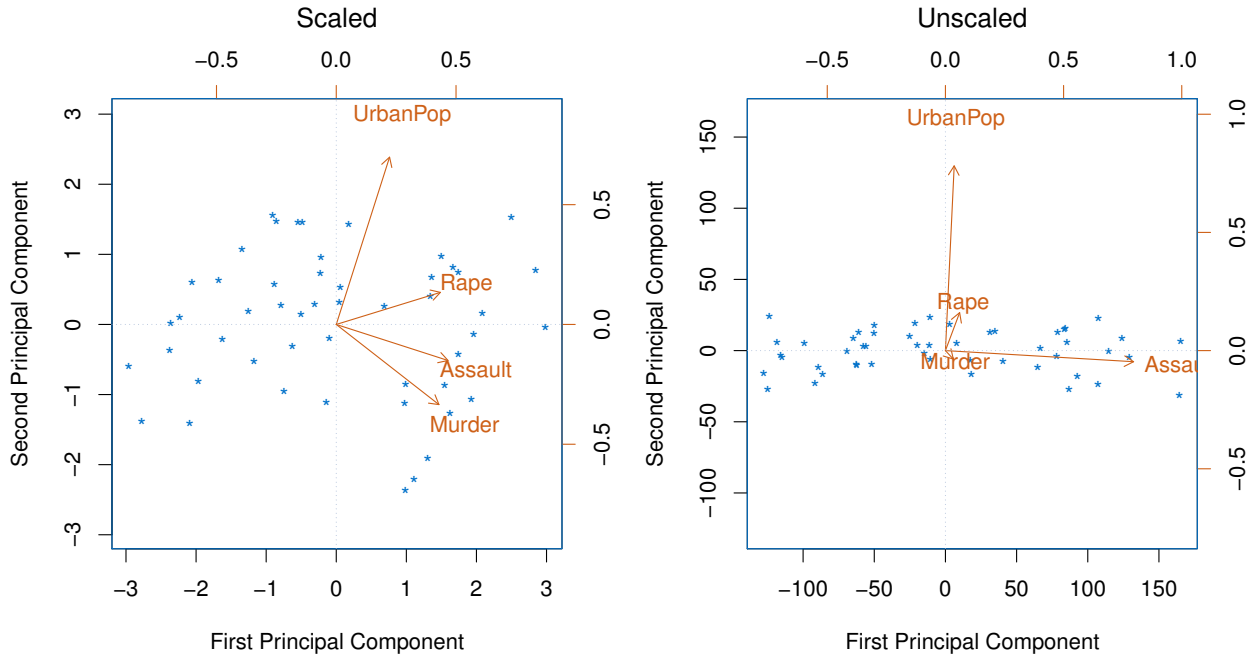


PCA find the hyperplane closest to the observations

- 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).
- 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

- 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

- 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

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

and the variance explained by the mth principal component is

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

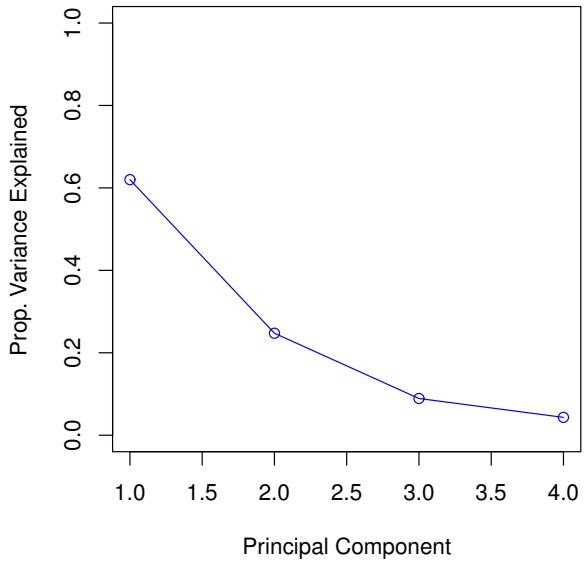
- It can be shown that $\sum_{j=1}^{p} \mathrm{Var}(X_j) = \sum_{m=1}^{M} \mathrm{Var}(Zm)$, with $M = \min(n-1, p)$.

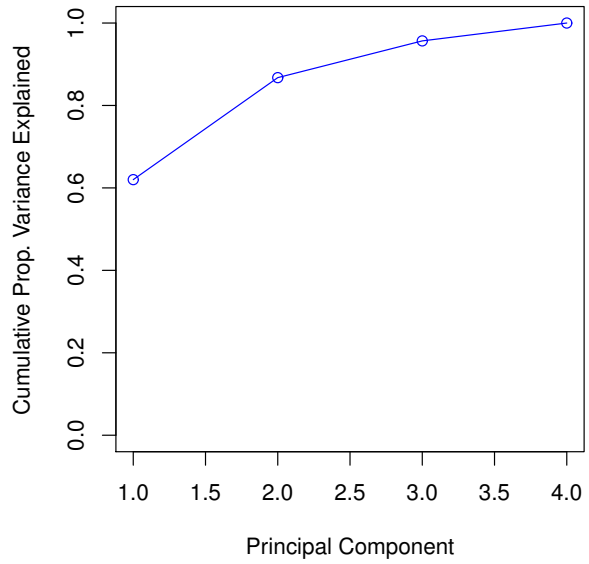
Proportion Variance Explained: continued

- Therefore, the PVE of the mth principal component is given by the positive quantity between 0 and 1

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

- 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.
- the "scree plot" on the previous slide can be used as a guide: we look for an "elbow."

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.
- To make this concrete, 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.

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.

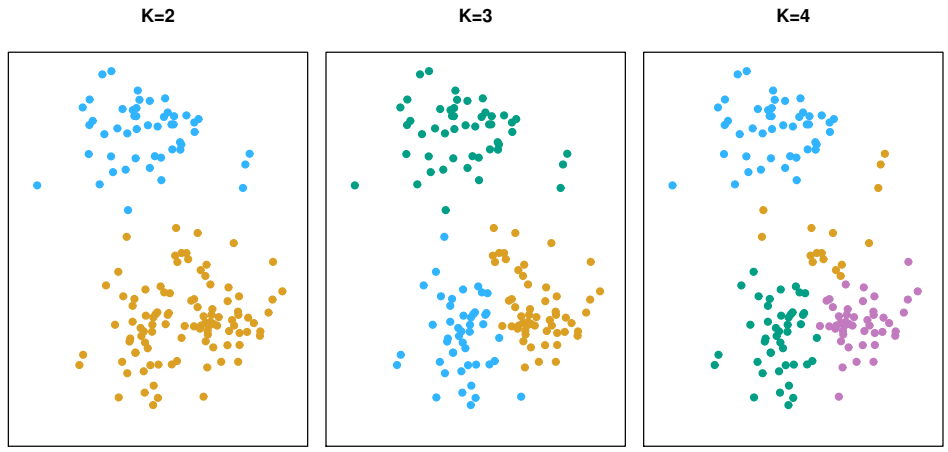
Clustering for Market Segmentation

- 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.
- 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.
- 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.
- 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



- 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.

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:

  1. $C_1 \cup C_2 \cup \cdots \cup C_K = \{1, \ldots, n\}$. In other words, each observation belongs to at least one of the K clusters.
  2. $C_k \cap C_{k'} = \emptyset$ for all $k \neq k'$. In other words, the clusters are non-overlapping: no observation belongs to more than one cluster.

- For instance, if the ith observation is in the kth 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.
- The within-cluster variation for cluster C_k is a measure $\mathrm{WCV}(C_k)$ of the amount by which the observations within a cluster differ from each other.
- Hence we want to solve the problem

$$\underset{C_1,\ldots,C_K}{\text{minimize}} \left\{ \sum_{k=1}^{K} \mathrm{WCV}(C_k) \right\}. \tag{2}$$

- 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

$$\mathrm{WCV}(C_k) = \frac{1}{|C_k|} \sum_{i,i' \in C_k} \sum_{j=1}^{p} (x_{ij} - x_{i'j})^2, \tag{3}$$

  where $|C_k|$ denotes the number of observations in the kth cluster.

- Combining (2) and (3) gives the optimization problem that defines K-means clustering,

$$\underset{C_1,\ldots,C_K}{\text{minimize}} \left\{ \sum_{k=1}^{K} \frac{1}{|C_k|} \sum_{i,i' \in C_k} \sum_{j=1}^{p} (x_{ij} - x_{i'j})^2 \right\}. \tag{4}$$

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:
   - 2.1 For each of the K clusters, compute the cluster centroid. The kth cluster centroid is the vector of the p feature means for the observations in the kth cluster.
   - 2.2 Assign each observation to the cluster whose centroid is closest (where closest is defined using Euclidean distance).

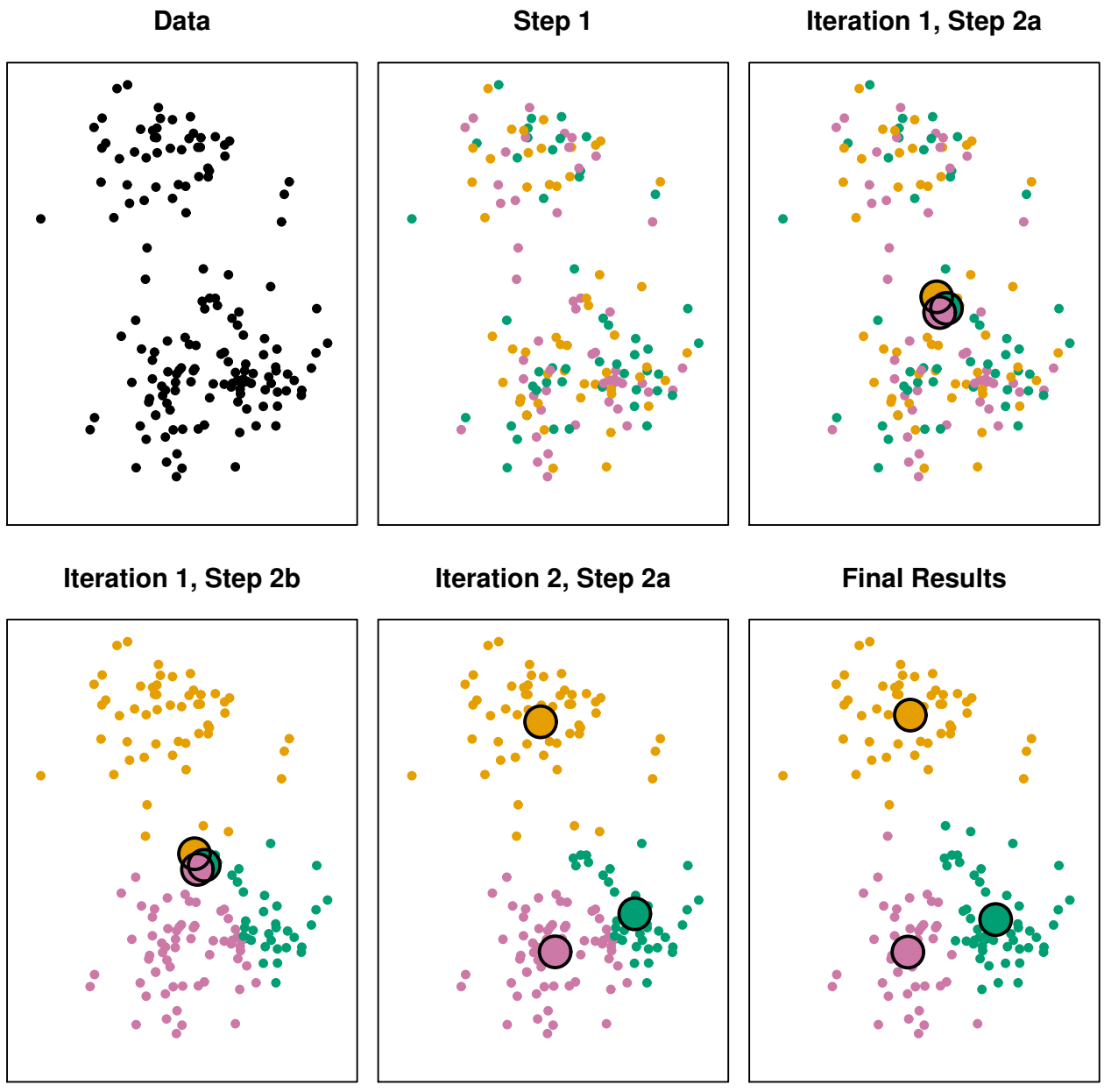
Properties of the Algorithm

- This algorithm is guaranteed to decrease the value of the objective (4) at each step. Note that

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

  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.

K-means clustering

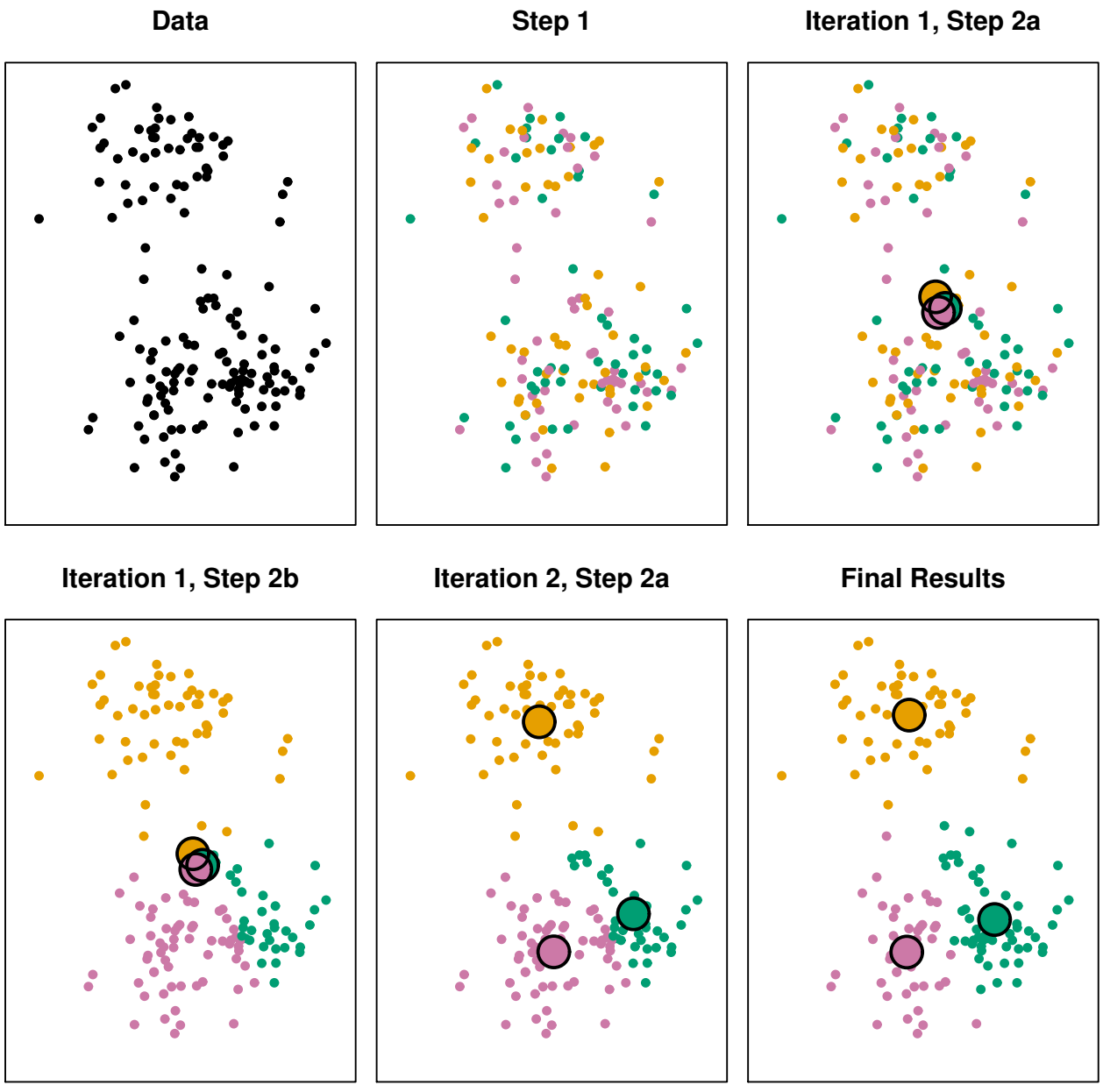


Details of Previous Figure

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.
- 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.
- Above each plot is the value of the objective (4).
- 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

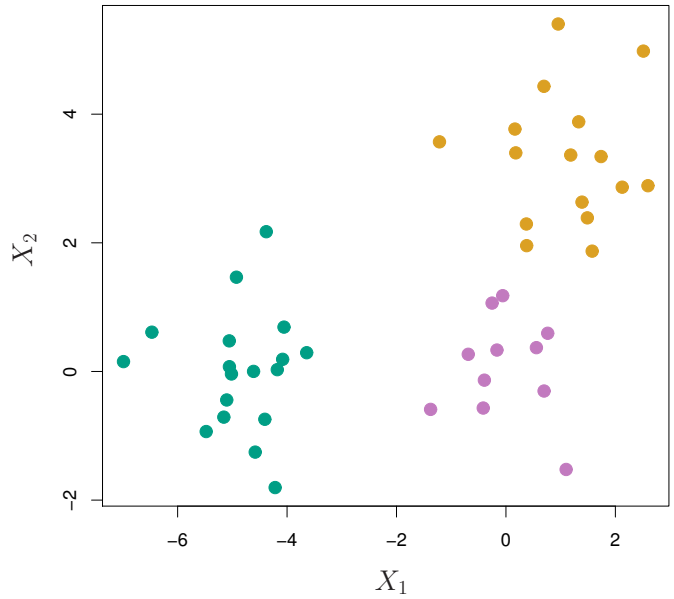
Hierarchical Clustering

- 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.
- Here, 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 Algorithm

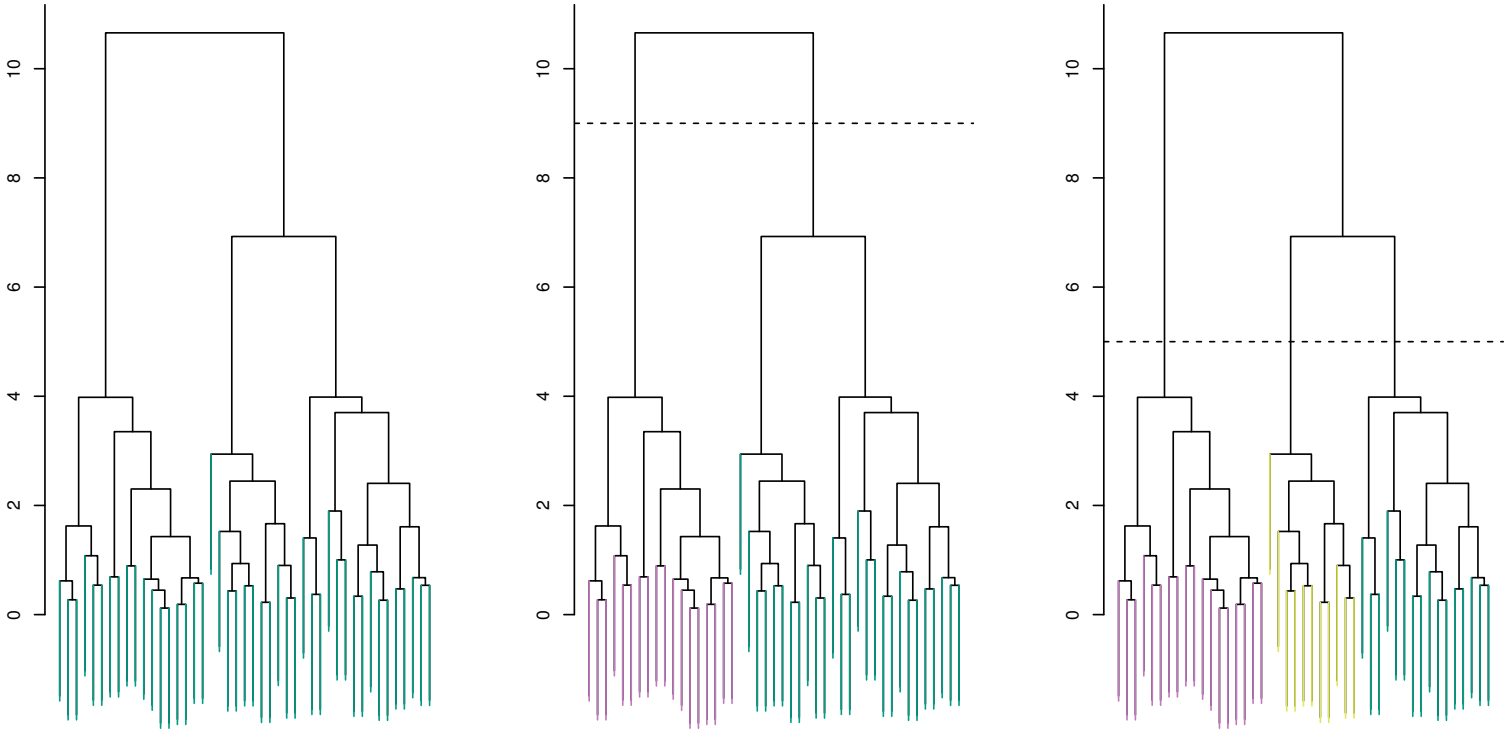
- Start with each point in its own cluster.
- Identify the closest two clusters and merge them.
- Repeat.
- Ends when all points are in a single cluster.

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.

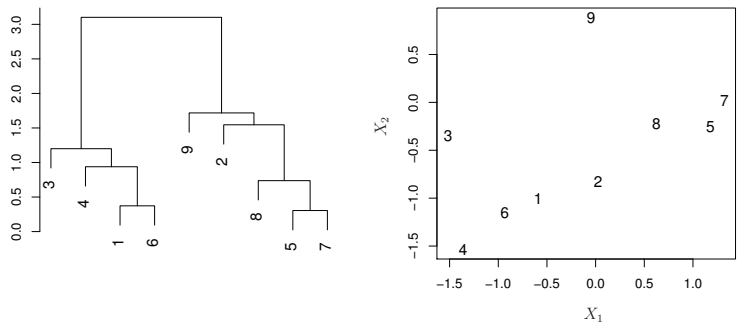
Application of hierarchical clustering



Details of previous figure

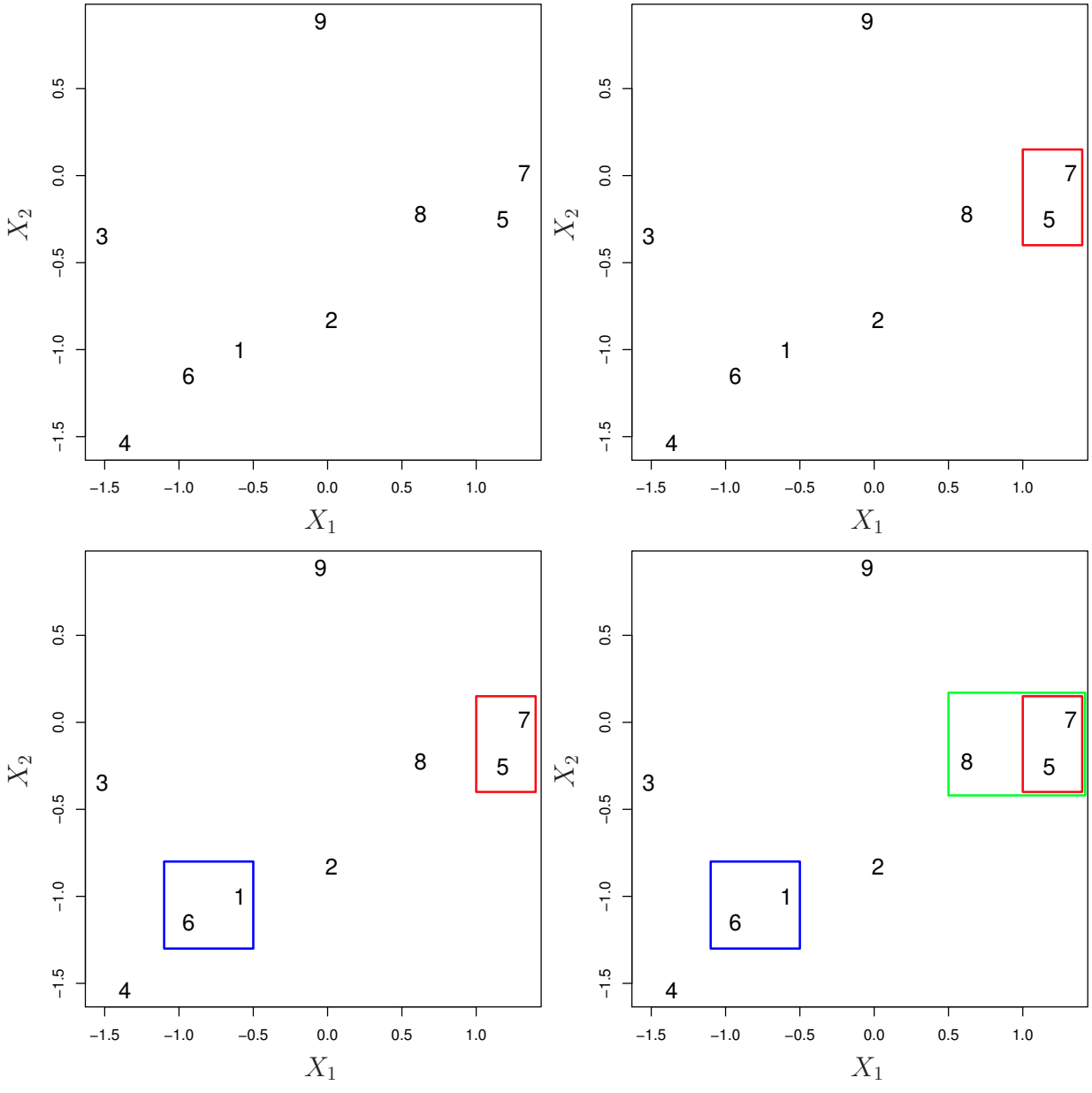
- Left: Dendrogram obtained from hierarchically clustering the data from previous slide, with complete linkage and Euclidean distance.
- Center: The dendrogram from the left-hand panel, cut at a height of 9 (indicated by the dashed line). This cut results in two distinct clusters, shown in different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

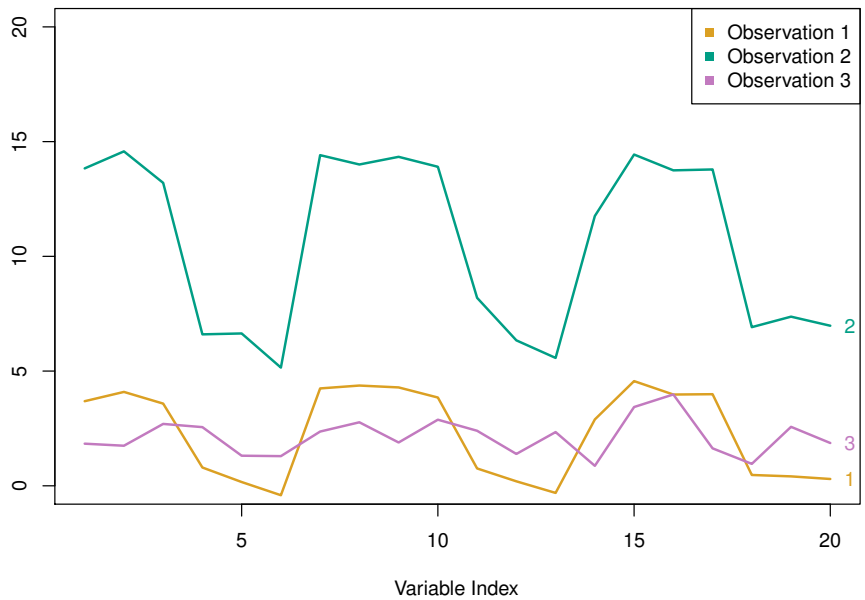


Types of Linkage

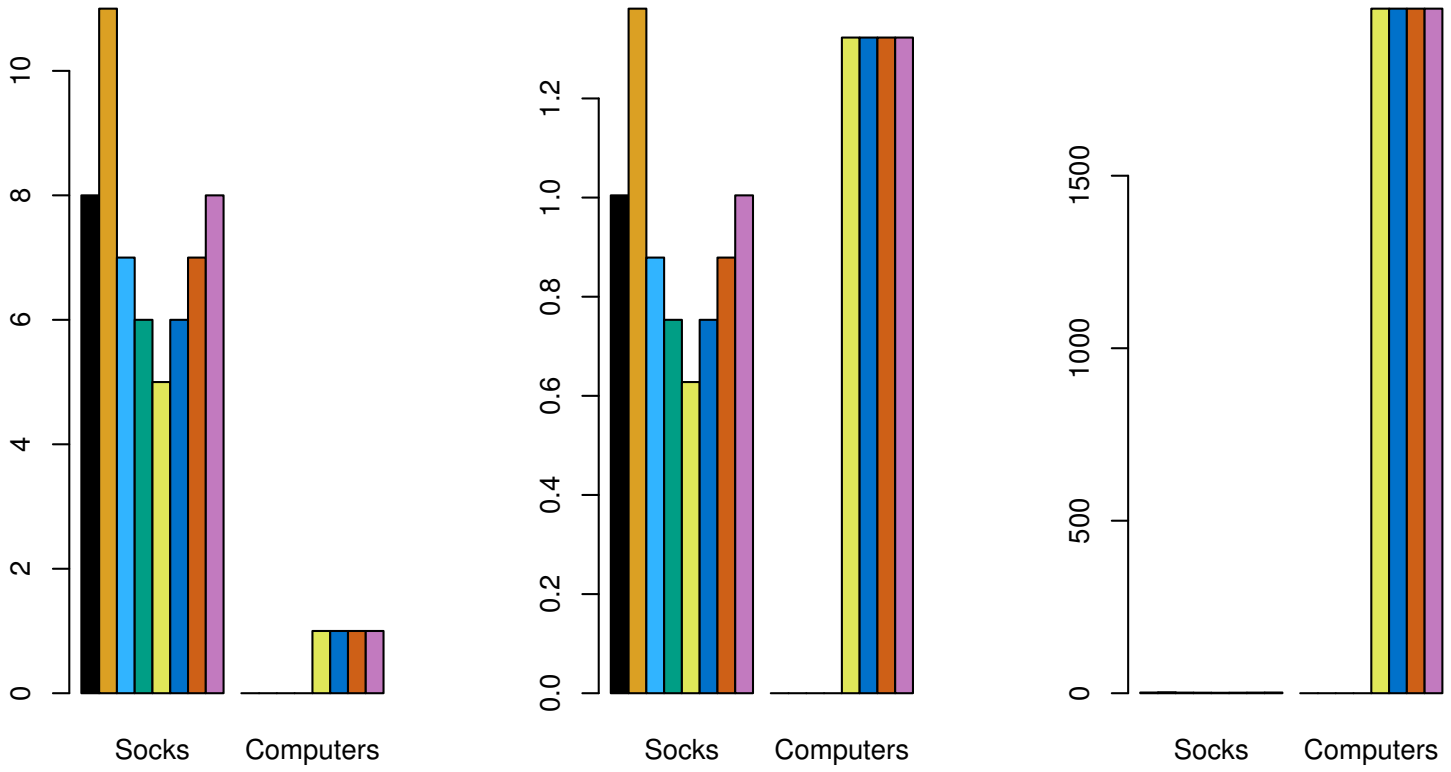
Linkage	Description
Complete	Maximal inter-cluster dissimilarity. Compute all pairwise dissimilarities between the observations in cluster A and the observations in cluster B, and record the largest of these dissimilarities.
Single	Minimal inter-cluster dissimilarity. Compute all pairwise dissimilarities 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.

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.



Scaling of the variables matters



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.

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).
- 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.