Unsupervised Learning

Szu-Chi Chung

Department of Applied Mathematics, National Sun Yat-sen University

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, 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?

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

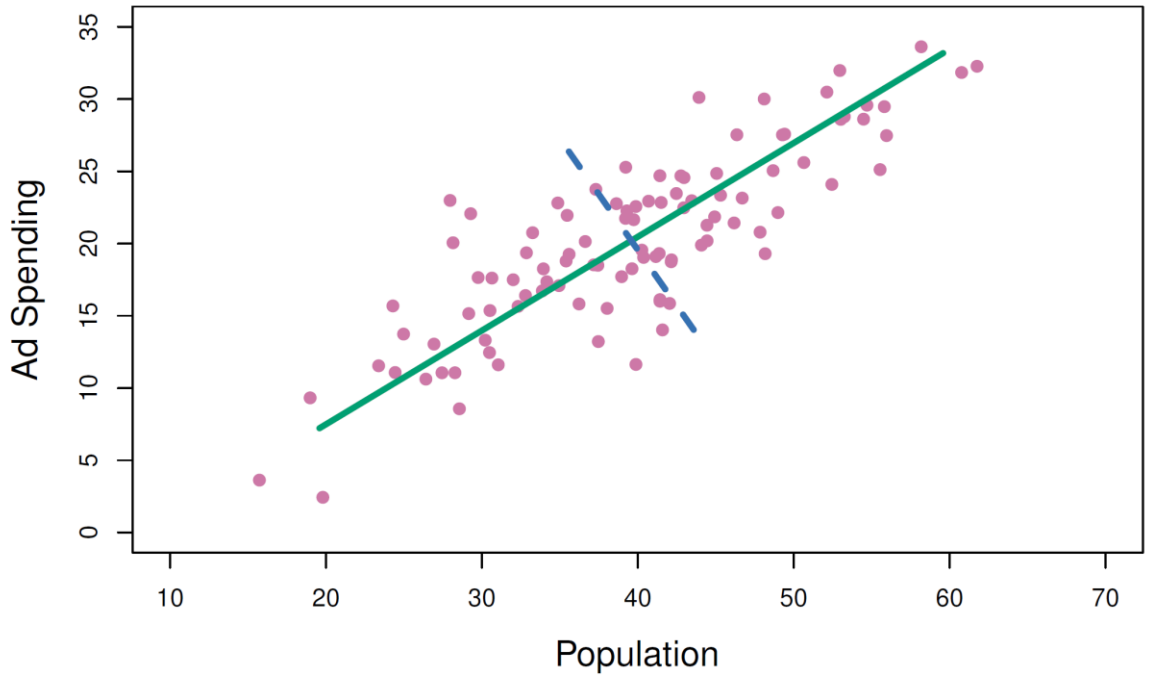
$$Z_1 = \Phi_{11}X_1 + \Phi_{21}X_2 + \cdots + \Phi_{p1}X_p$$

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

- 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, $\Phi_1 = (\Phi_{11}\Phi_{21} \ldots \Phi_{\mathrm{p1}})^T$
- 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 X. Since we are only interested in variance, we assume 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

$$z_{i1} = \Phi_{11} x_{i1} + \Phi_{21} x_{i2} + \cdots + \Phi_{p1} x_{ip}$$

  for $i = 1, \ldots, n$ that has largest sample variance, subject to the constraint that $\sum_{j=1}^{p} \Phi_{j1}^2$
- 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

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

- 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 score, with realized values $z_{11}, \ldots, z_{n1}$

Geometry of PCA

- The loading vector Φ_1 with elements $\Phi_{11}\Phi_{21}$ ... Φ_{p1} denfies a direction in feature space along which the data vary the most
- If we project the n data points $x_1, x_2, \ldots, x_n$ 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 $X_1, X_2, \ldots, X_p$ that has maximal variance among all linear combinations that are uncorrelated with Z_1
- The second principal component scores $z_{12}, \ldots, z_{n2}$ 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 $\Phi_{12} \Phi_{22} \ldots \Phi_{p2}$

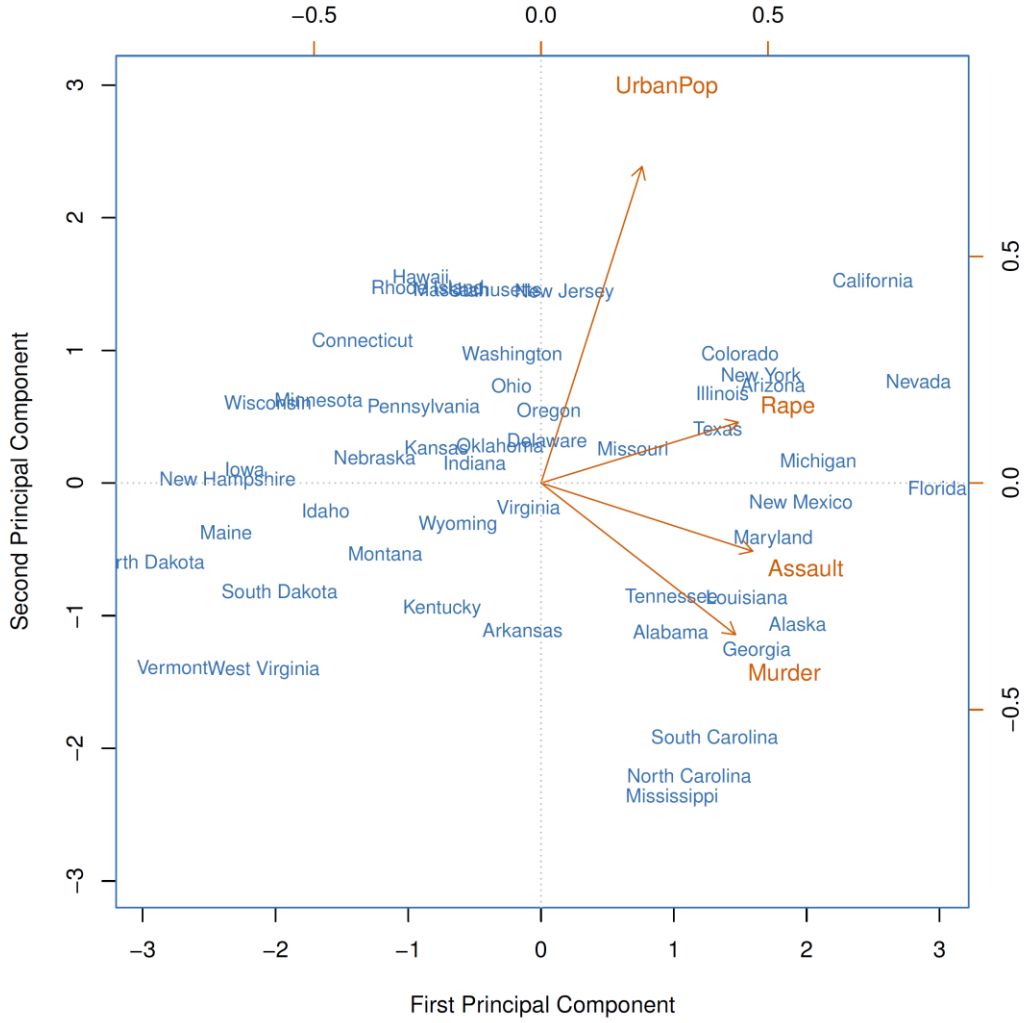
Further principal components: continued

- 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 Φ_1, Φ_2, Φ_3, ... are the ordered sequence of right singular vectors of the matrix 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

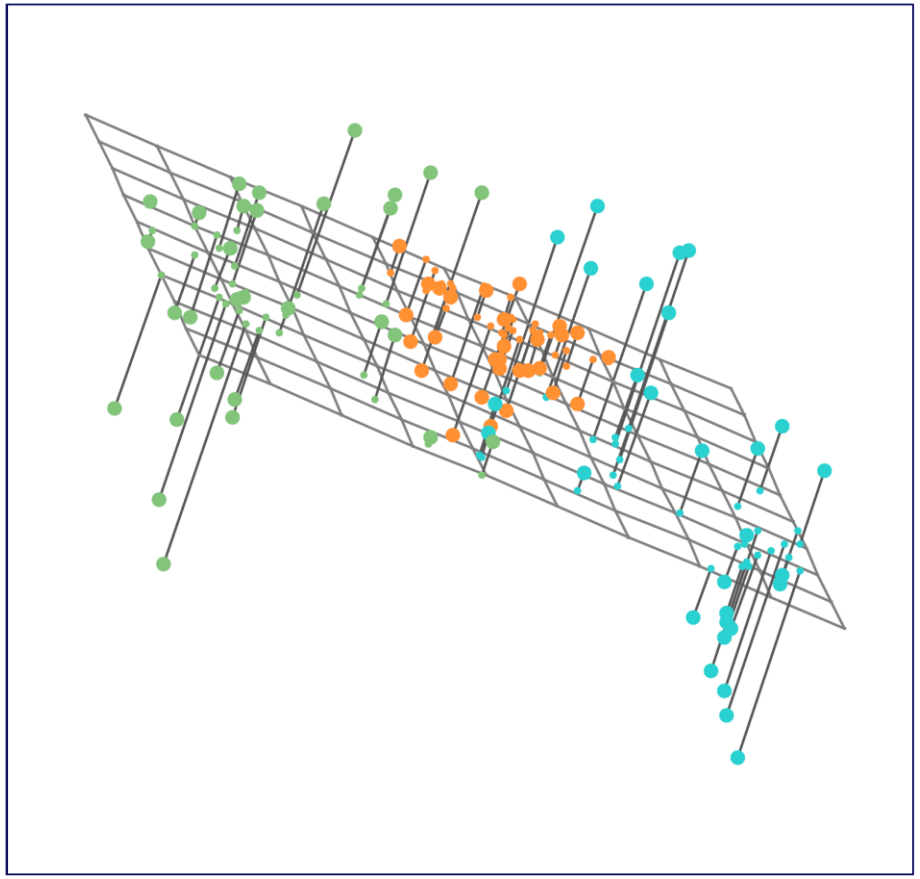


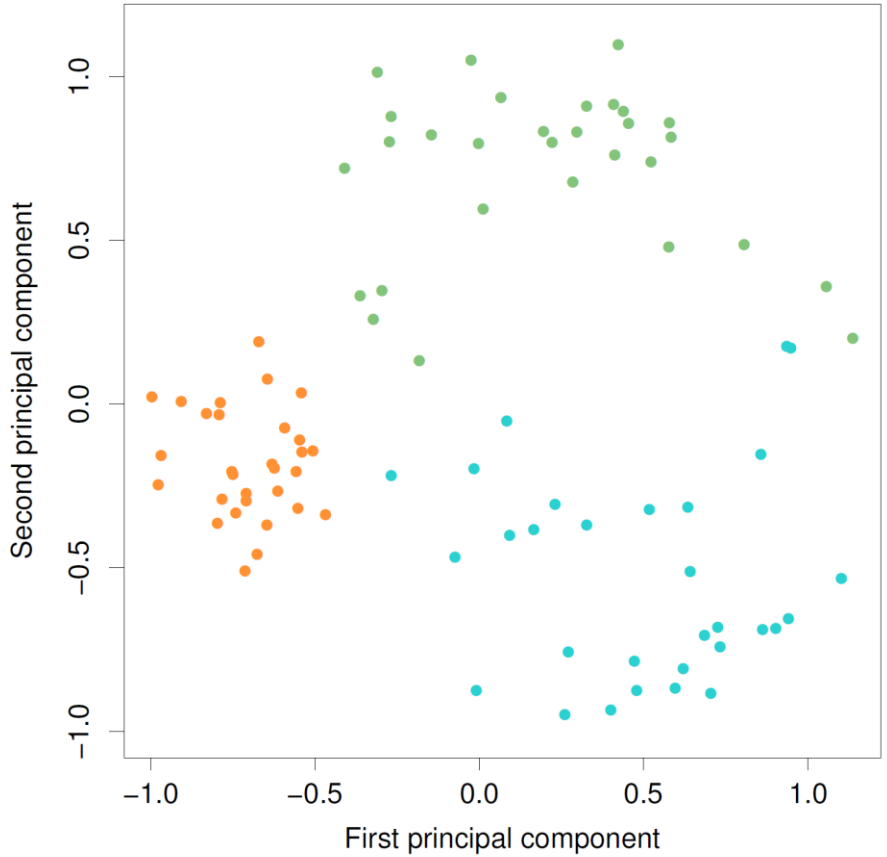
	PC1	PC2
Murder	0.5358995	-0.4181809
Assault	0.5831836	-0.1879856
UrbanPop	0.2781909	0.8728062
Rape	0.5434321	0.1673186

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

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 denes 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 dataset span the plane that is closest to the n observations, in terms of average squared Euclidean distance

Another Interpretation of Principal Components

- The first M principal component score vectors and the first M principal component loading vectors provide the best M-dimensional approximation (in terms of Euclidean distance) to the ith observation x_{ij}

$$x_{ij} \approx \sum_{m=1}^{M} z_{im}\Phi_{jm}$$

- Suppose the data matrix X is column-centered. Out of all approximations of the form $x_{ij} \approx \sum_{m=1}^{M} a_{im}b_{jm}$. We have

$$\min_{A\in R^{n\times M},\, B\in R^{p\times M}} \{\sum_{j=1}^{p}\sum_{i=1}^{n}(x_{ij} - \sum_{m=1}^{M} a_{im}b_{jm})^2\}$$

- It can be shown that for any value of M, the columns of the matrices $\hat{A}$ and $\hat{B}$ are in fact the first M principal components score and loading vectors

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

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

and the variance explained by the mth principal component is

$$Var(Z_m) = \frac{1}{n} \sum_{i=1}^{n} z_{im}^2 = \frac{1}{n} \sum_{i=1}^{n} (\sum_{j=1}^{p} \Phi_{jm} x_{ij})^2$$

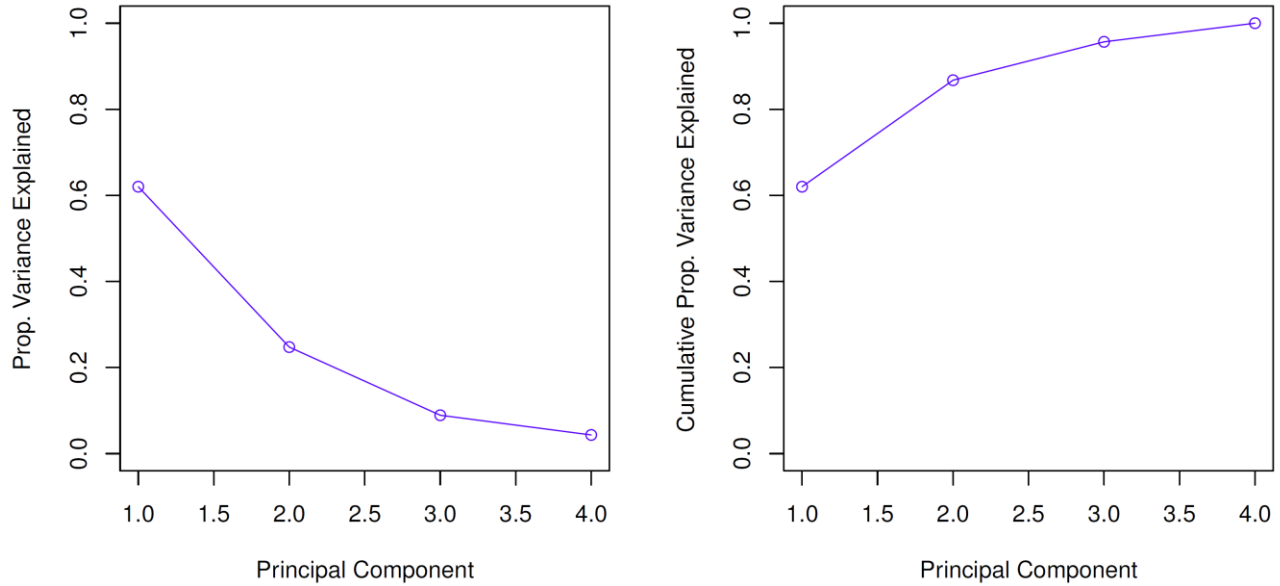
- It can be shown that $\sum_{j=1}^{p} Var(X_j) = \sum_{m=1}^{M} Var(Z_m)$, with $M = \min(n-1, p)$

Proportion Variance Explained: continued

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

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

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



More on PCA – Connection with SVD

- Let $x_1, \ldots, x_n$ be length-p observation vectors

$$x_i = \begin{pmatrix} x_{i1} \\ x_{i2} \\ \vdots \\ x_{ip} \end{pmatrix}$$

- Without Loss Of Generality (WLOG), let their mean be length-p 0-vector
- Let the data matrix $X = (x_1, x_2, \ldots, x_n)$ be a p by n matrix
- The sample covariance matrix

$$S = XX^T/(n-1) = \sum_{i=1}^{n} x_i x_i^T/(n-1) = \sum_{i=1}^{n} (x_i - \bar{x})(x_i - \bar{x})^T/(n-1)$$

More on PCA – Connection with SVD

- Find a direction vector $u_1 \in R^p$ and $u_1^T u_1 = 1$ such that the variance of the projected data is maximized

$$\frac{1}{n}\sum_{i=1}^{n}(u_1^T x_i - u_1^T \bar{x})^2 = u_1^T S u_1$$

  - To enforce the constraint, we introduce a Lagrange multiplier denoted by λ_1 and get the unconstrained maximization of

$$u_1^T S u_1 + \lambda_1(1 - u_1^T u_1) \text{ or maximize } \frac{u^T S u}{u^T u}$$

  - By setting the derivative with respect to u_1 equal to zero, we see that this quantity will have a stationary point when

$$S u_1 = \lambda_1 u_1$$

A is not a function of **x** **A** is symmetric	$\frac{\partial \mathbf{x}^\top \mathbf{A}\mathbf{x}}{\partial \mathbf{x}} =$	$2\mathbf{x}^\top \mathbf{A}$	$2\mathbf{A}\mathbf{x}$
u = **u**(**x**), **v** = **v**(**x**)	$\frac{\partial(\mathbf{u}\cdot\mathbf{v})}{\partial \mathbf{x}} = \frac{\partial \mathbf{u}^\top \mathbf{v}}{\partial \mathbf{x}} =$	$\mathbf{u}^\top \frac{\partial \mathbf{v}}{\partial \mathbf{x}} + \mathbf{v}^\top \frac{\partial \mathbf{u}}{\partial \mathbf{x}}$ $\frac{\partial \mathbf{u}}{\partial \mathbf{x}}, \frac{\partial \mathbf{v}}{\partial \mathbf{x}}$ in numerator layout	$\frac{\partial \mathbf{u}}{\partial \mathbf{x}}\mathbf{v} + \frac{\partial \mathbf{v}}{\partial \mathbf{x}}\mathbf{u}$ $\frac{\partial \mathbf{u}}{\partial \mathbf{x}}, \frac{\partial \mathbf{v}}{\partial \mathbf{x}}$ in denominator layout

More on PCA – Connection with SVD

- u_1 must be an eigenvector of S, if we left-multiply by u_1^T we get

$$u_1^T S u_1 = \lambda_1$$

  - and so the variance will be a maximum when we set u_1 equal to the eigenvector having the largest eigenvalue λ_1. This eigenvector is known as the first principal component.

- We can define additional principal components in an incremental fashion by choosing each new direction to be that which maximizes the projected variance amongst all possible directions orthogonal to those already considered.
  - In a r-dimensional projection space, we now consider the optimal linear projection for which the variance of the projected data is maximized is defined by the r eigenvectors $u_1, \ldots, u_r$ of the data covariance matrix S corresponding to the r largest eigenvalues $\lambda_1, \ldots, \lambda_r$.

More on PCA – Connection with SVD

- If we collect eigenvectors and eigenvalues into matrix

$$S_{p\times p}U_{p\times p} = U_{p\times p}\Lambda_{p\times p}$$
$$S_{p\times p} = U_{p\times p}\Lambda_{p\times p}U^T_{p\times p}$$

- Note $X = USV^T$
  - Scores are $U^T X = SV^T$
- It is equivalent to Minimum error formulation

$$argmin_{U\epsilon\, O_{p,r}} \sum_{i=1}^{n} |(X_i - \bar{X}) - UU^T(X_i - \bar{X}))|_F^2$$

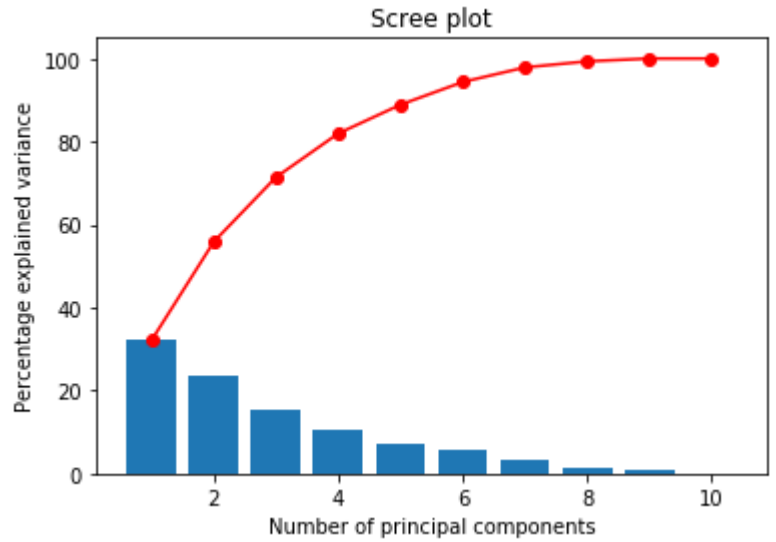
	Convention 1	Convention 2
U	Principal component Principal direction Loading	Principal axis Principal direction
$U^T X$	Principal component scores	Principal component

More on PCA – Connection with SVD

- Connection with SVD

$$S = \frac{XX^T}{n-1} = \frac{UDV^T VDU^T}{n-1} = U\frac{D^2}{n-1}U^T = U\Lambda U^T$$

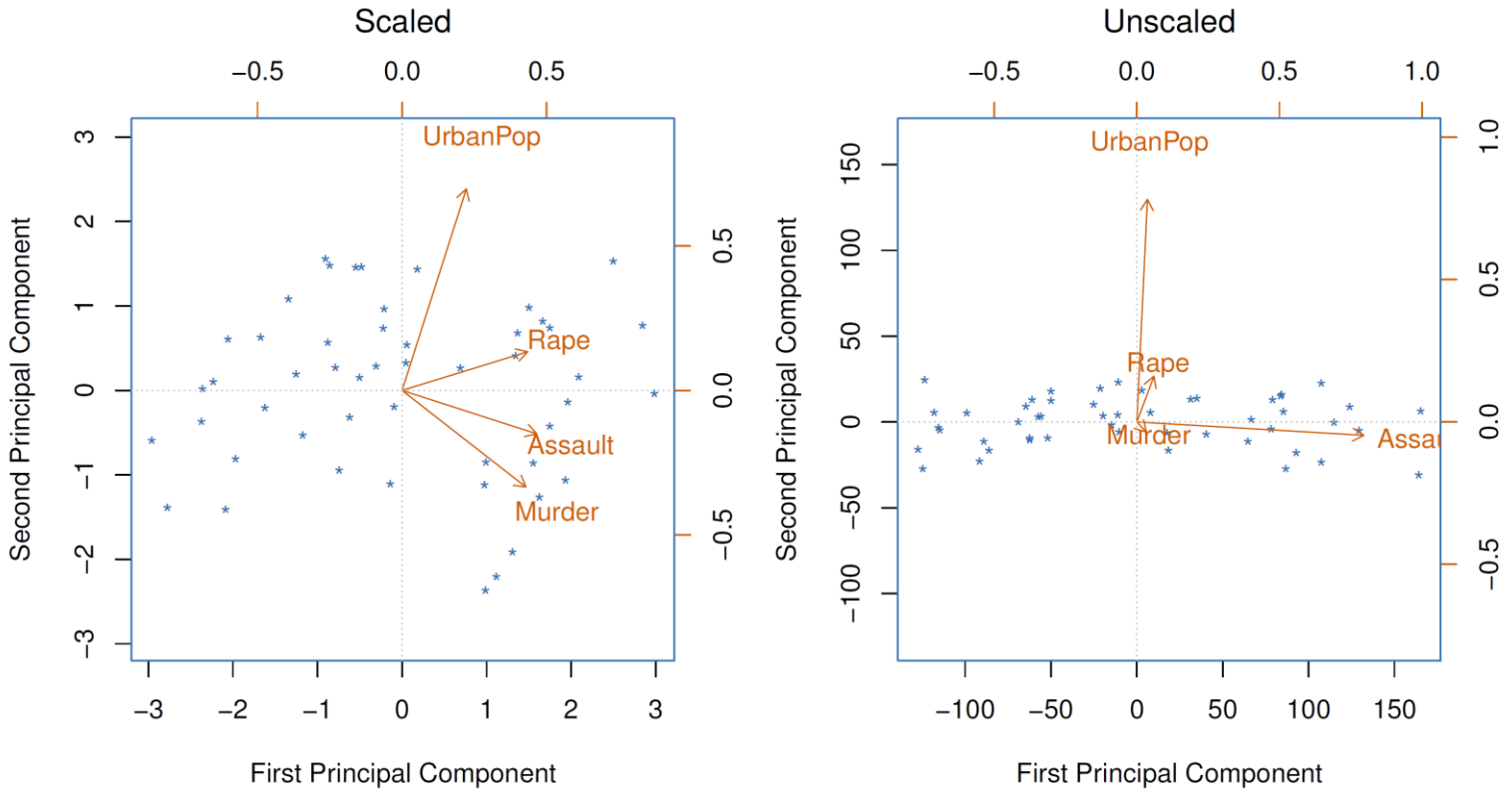
- In practice, we will often scale data before PCA



- Whiten data matrix (identity covariance matrix)
  - $\Lambda^{-1/2}U^T X$
- ZCA (Close to original data (often not reduce dimension))
  - $U\Lambda^{-1/2}U^T X$

More on 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



More on PCA - 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?
  - 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”

Missing Values and Matrix Completion

- Often datasets have missing values, which can be a nuisance. How should we proceed?
- We could remove the rows that contain missing observations and perform our data analysis on the complete rows
  - But this seems wasteful, and depending on the fraction missing
- Alternatively, if x_{ij} is missing, then we could replace it by the mean of the jth column (using the non-missing entries to compute the mean)
  - Although this is a common and convenient strategy, often we can do better by exploiting the correlation between the variables

Missing Values and Matrix Completion

- We show how principal components can be used to impute the missing values, through a process known as matrix completion
  - The complete matrix can then be used in a statistical learning method, such as linear regression or LDA
- This approach for imputing missing data is appropriate if the missingness is random
- Sometimes data is missing by necessity
  - For example, if we form a matrix of the ratings (on a scale from 1 to 5) that n customers have given to the entire Netflix catalog of p movies, then most of the matrix will be missing, since no customer will have seen and rated more than a tiny fraction of the catalog
  - If we can impute the missing values well, then we will have an idea of what each customer will think of movies they have not yet seen. Hence matrix completion can be used to power recommender systems

Principal Components with Missing Values

- The first M principal component score and loading vectors provide the "best" approximation to the data matrix X
- Now, some of the observations x_{ij} are missing. One can both impute the missing values and solve the principal component problem at the same time

$$\min_{A \in R^{n \times M},\, B \in R^{p \times M}} \{ \sum_{(i,j) \in O} (x_{ij} - \sum_{m=1}^{M} a_{im} b_{jm})^2 \}$$

  where O is the set of all observed pairs of indices (i, j), a subset of the possible $n \times p$ pairs

- We can estimate a missing observation x_{ij} using $x_{ij} = \sum_{m=1}^{M} \hat{a}_{im} \hat{b}_{jm}$
- We can (approximately) recover the M principal component scores and loadings, as we did when the data were complete

Algorithm 12.1 *Iterative Algorithm for Matrix Completion*

1. Create a complete data matrix $\tilde{\mathbf{X}}$ of dimension $n \times p$ of which the (i, j) element equals

$$\tilde{x}_{ij} = \begin{cases} x_{ij} & \text{if } (i,j) \in \mathcal{O} \\ \bar{x}_j & \text{if } (i,j) \notin \mathcal{O}, \end{cases}$$

   where $\bar{x}_j$ is the average of the observed values for the jth variable in the incomplete data matrix $\mathbf{X}$. Here, $\mathcal{O}$ indexes the observations that are observed in $\mathbf{X}$.

2. Repeat steps (a)–(c) until the objective (12.14) fails to decrease:

   (a) Solve

$$\underset{\mathbf{A}\in\mathbb{R}^{n\times M},\mathbf{B}\in\mathbb{R}^{p\times M}}{\text{minimize}} \left\{ \sum_{j=1}^{p}\sum_{i=1}^{n}\left(\tilde{x}_{ij} - \sum_{m=1}^{M} a_{im}b_{jm}\right)^2 \right\} \tag{12.13}$$

   by computing the principal components of $\tilde{\mathbf{X}}$.

   (b) For each element $(i, j) \notin \mathcal{O}$, set $\tilde{x}_{ij} \leftarrow \sum_{m=1}^{M} \hat{a}_{im}\hat{b}_{jm}$.

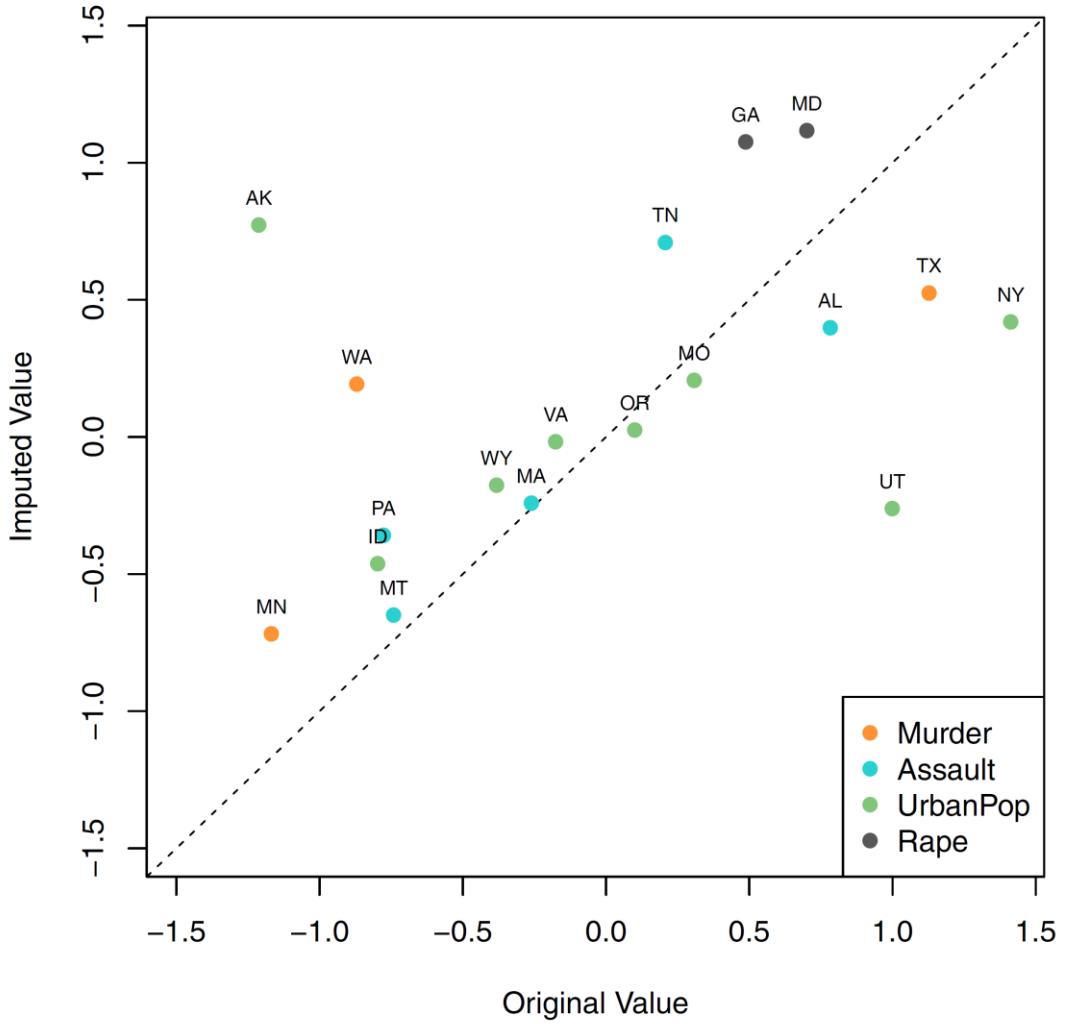
   (c) Compute the objective

$$\sum_{(i,j)\in\mathcal{O}}\left(x_{ij} - \sum_{m=1}^{M}\hat{a}_{im}\hat{b}_{jm}\right)^2. \tag{12.14}$$

3. Return the estimated missing entries $\tilde{x}_{ij}$, $(i, j) \notin \mathcal{O}$.

Example on USArrests data

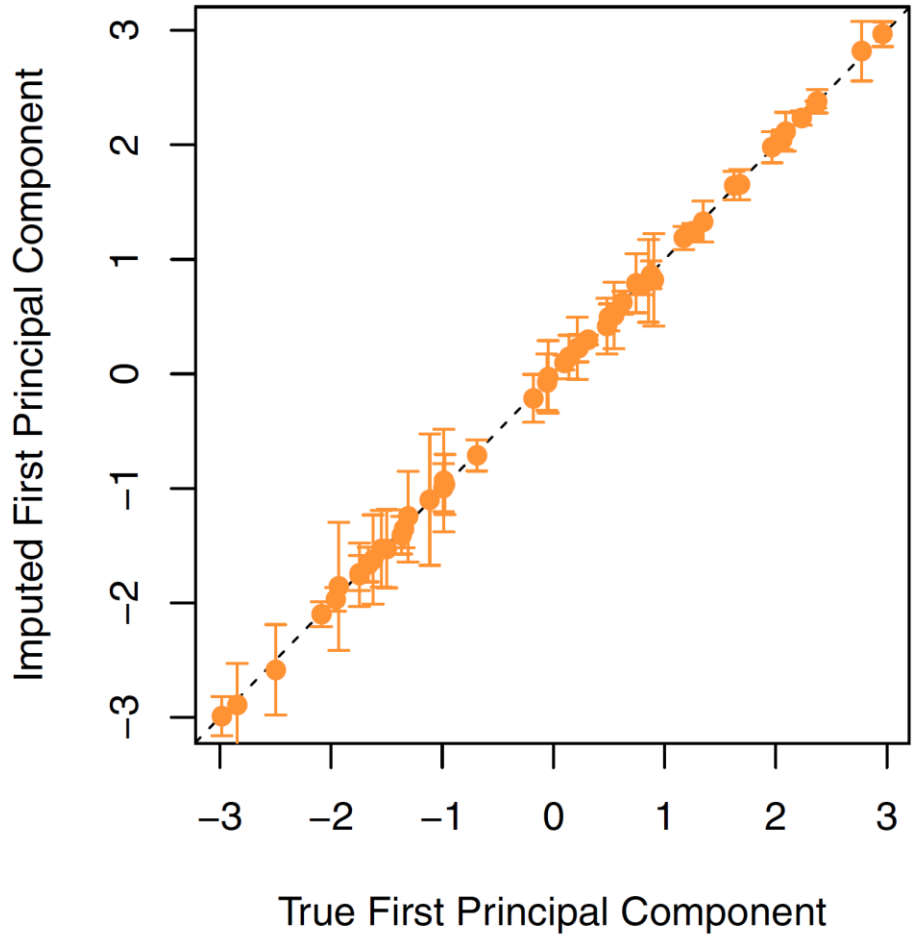
- $p = 4$ and $n = 50$ observations (states). We first standardized the data
- We then randomly selected 20 of the 50 states, and then for each of these we randomly set one of the four variables to be missing. Thus, 10% of the elements of the data matrix were missing.
- We applied Algorithm 12.1 with $M = 1$ principal component

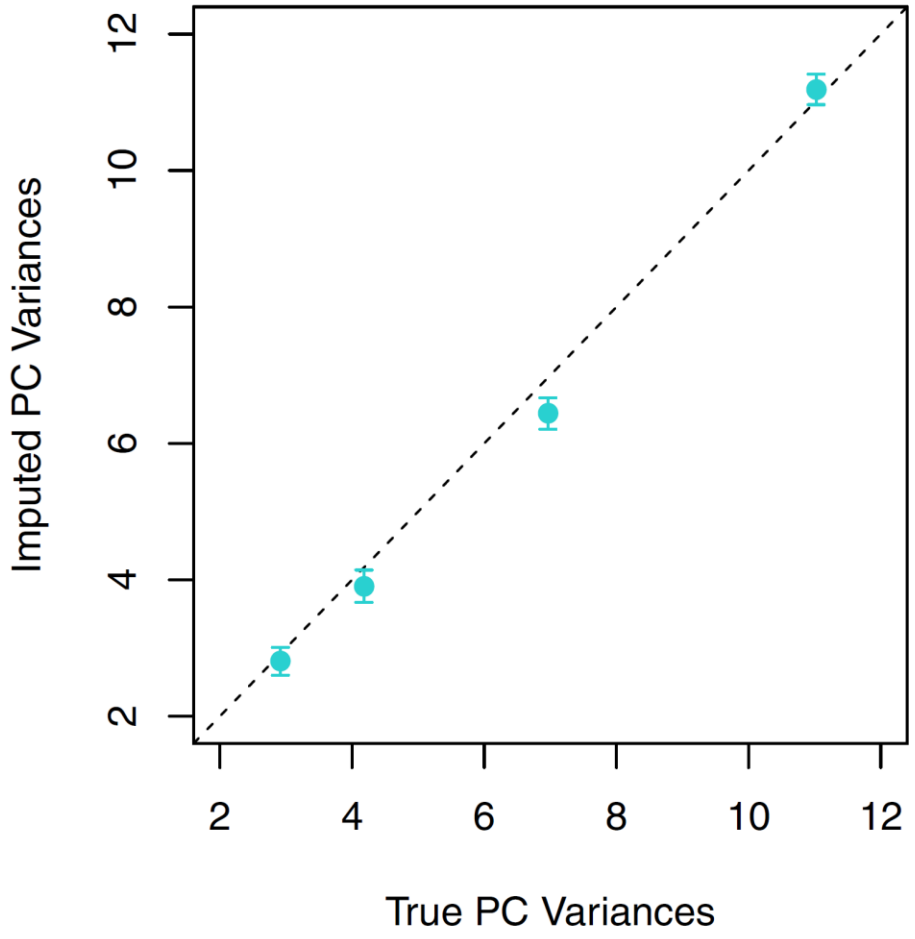


Example on USArrests data

- Over 100 random runs of this experiment, the average correlation between the true and imputed values of the missing elements is 0.63, with a standard deviation of 0.11
- If we had simply computed $\hat{x}_{ij} = z_{i1}\Phi_{j1}$, where z_{i1} and Φ_{j1} are elements of the first principal component score and loading vectors of the complete data. Using the complete data in this way results in an average correlation of 0.79 between the true and estimated values for these 20 elements, with a standard deviation of 0.08
- Thus, our imputation method does worse than the method that uses all of the data (0.63 ± 0.11 versus 0.79 ± 0.08), but its performance is still pretty good

Example on USArrests data





Recommender Systems

- Netflix and Amazon use data about the content that a customer has viewed in the past to suggest other content for the customer
- Some years back, Netflix had customers rate each movie that they had seen with a score from 1–5. This resulted in a very big $n \times p$ matrix for which the (i, j) element is the rating given by the ith customer to the jth movie
- Netflix needed a way to impute the missing values of this data matrix
  - The key idea is as follows: the set of movies that the ith customer has seen will overlap with those that other customers have seen. Furthermore, some of those other customers will have similar movie preferences to the ith customer
  - By applying Algorithm 12.1, we can predict the ith customer's rating for the jth movie
  - We can interpret the M components in terms of "cliques" and "genres"

Recommender Systems

	Jerry Maguire	Oceans	Road to Perdition	A Fortunate Man	Catch Me If You Can	Driving Miss Daisy	The Two Popes	The Laundromat	Code 8	The Social Network	...
Customer 1	•	•	•	•	4	•	•	•	•	•	...
Customer 2	•	•	3	•	•	•	3	•	•	3	...
Customer 3	•	2	•	4	•	•	•	•	2	•	...
Customer 4	3	•	•	•	•	•	•	•	•	•	...
Customer 5	5	1	•	•	4	•	•	•	•	•	...
Customer 6	•	•	•	•	•	2	4	•	•	•	...
Customer 7	•	•	5	•	•	•	•	3	•	•	...
Customer 8	•	•	•	•	•	•	•	•	•	•	...
Customer 9	3	•	•	•	5	•	•	1	•	•	...
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋱

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,
- It make this concrete, we must dene 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

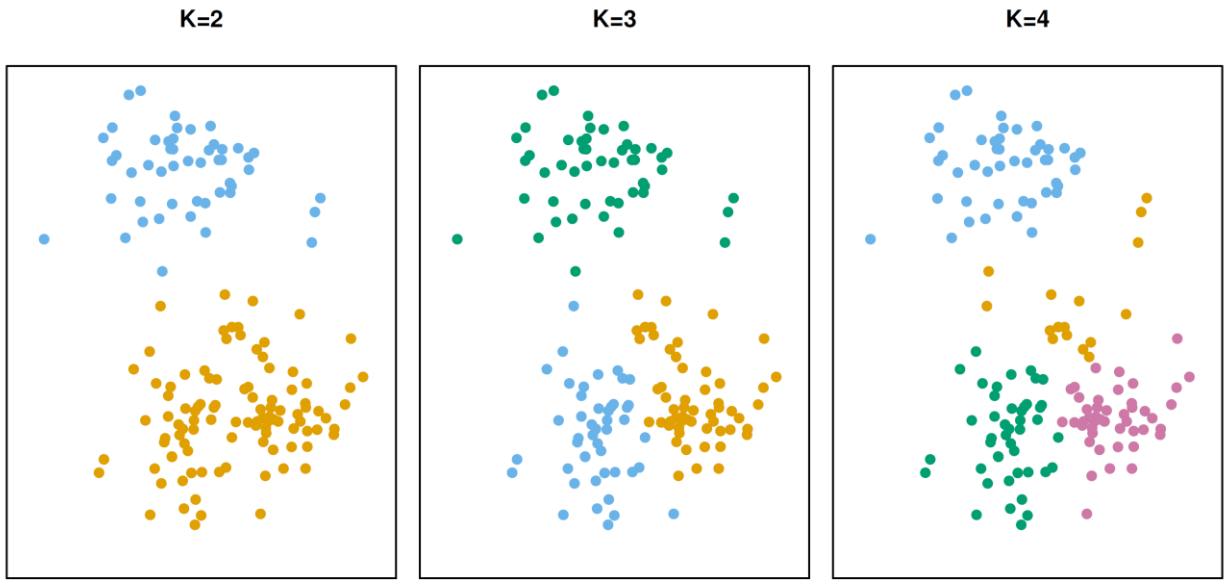
Clustering for Market Segmentation

- 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

Details of 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. $C_k \cap C_{k'} \neq 0$ 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

$$\min_{C_1,\ldots,C_k} \{\sum_{k=1}^{K} \mathrm{WCV}(C_k)\}$$

- 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

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

Where $|C_k|$ denotes the number of observations in the kth cluster

- Combining previous two equation gives the optimization problem which minimize the following objective function that defines K-means clustering,

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

Details of K-means clustering

Algorithm 12.2 *K-Means Clustering*

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:

   (a) 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.

   (b) 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 function at each step. Why? Note that

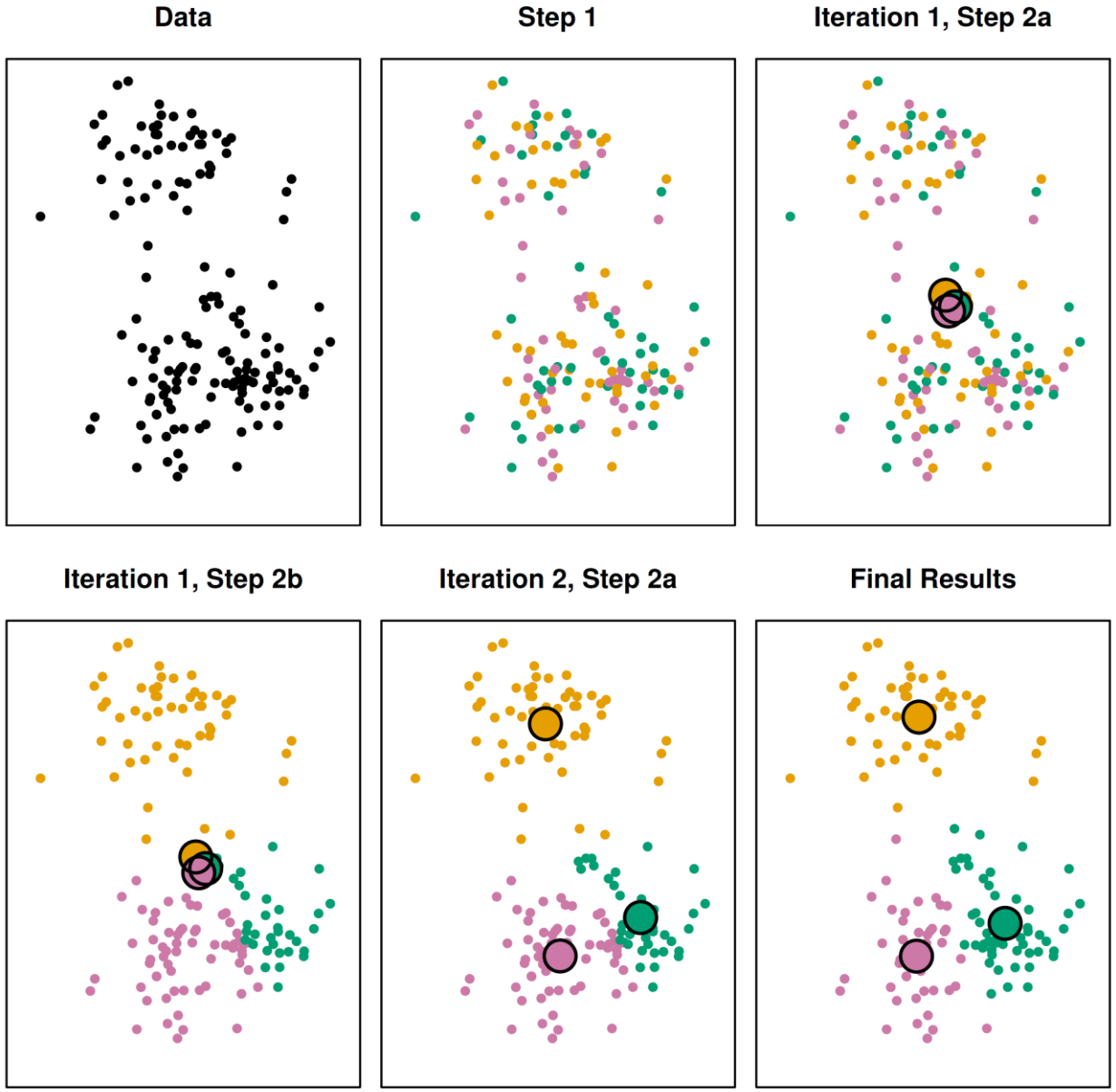
$$\frac{1}{|C_k|}\sum_{i,i' \in C_k}\sum_{j=1}^{p}(x_{ij}-x_{i'j})^2 = 2\sum_{i \in C_k}\sum_{j=1}^{p}(x_{ij}-\bar{x}_{kj})^2$$

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 and

$$\frac{1}{n}\sum_{i,j=1}^{n}(x_i - x_j)^2 = \frac{1}{n}\sum_{i,j=1}^{n}[(x_i-\bar{x})-(x_j-\bar{x})]^2 = \frac{2}{n}\sum_{i}^{n}(x_i-\bar{x})^2$$

- However it is not guaranteed to give the global minimum

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
- 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

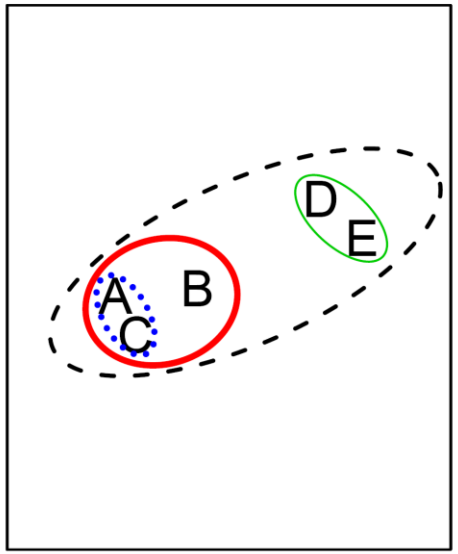
- K-means clustering performed six times on the data from previous figure with $K = 3$, each time with a different random assignment of the observations in Step 1 of the K-means algorithm.
- Above each plot is the value of the objective function
- Three different local optima were obtained, one of which 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

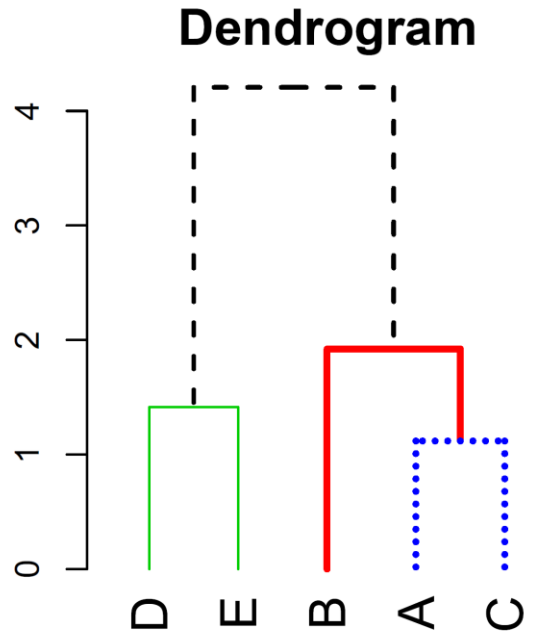
Hierarchical Clustering

- 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
- 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

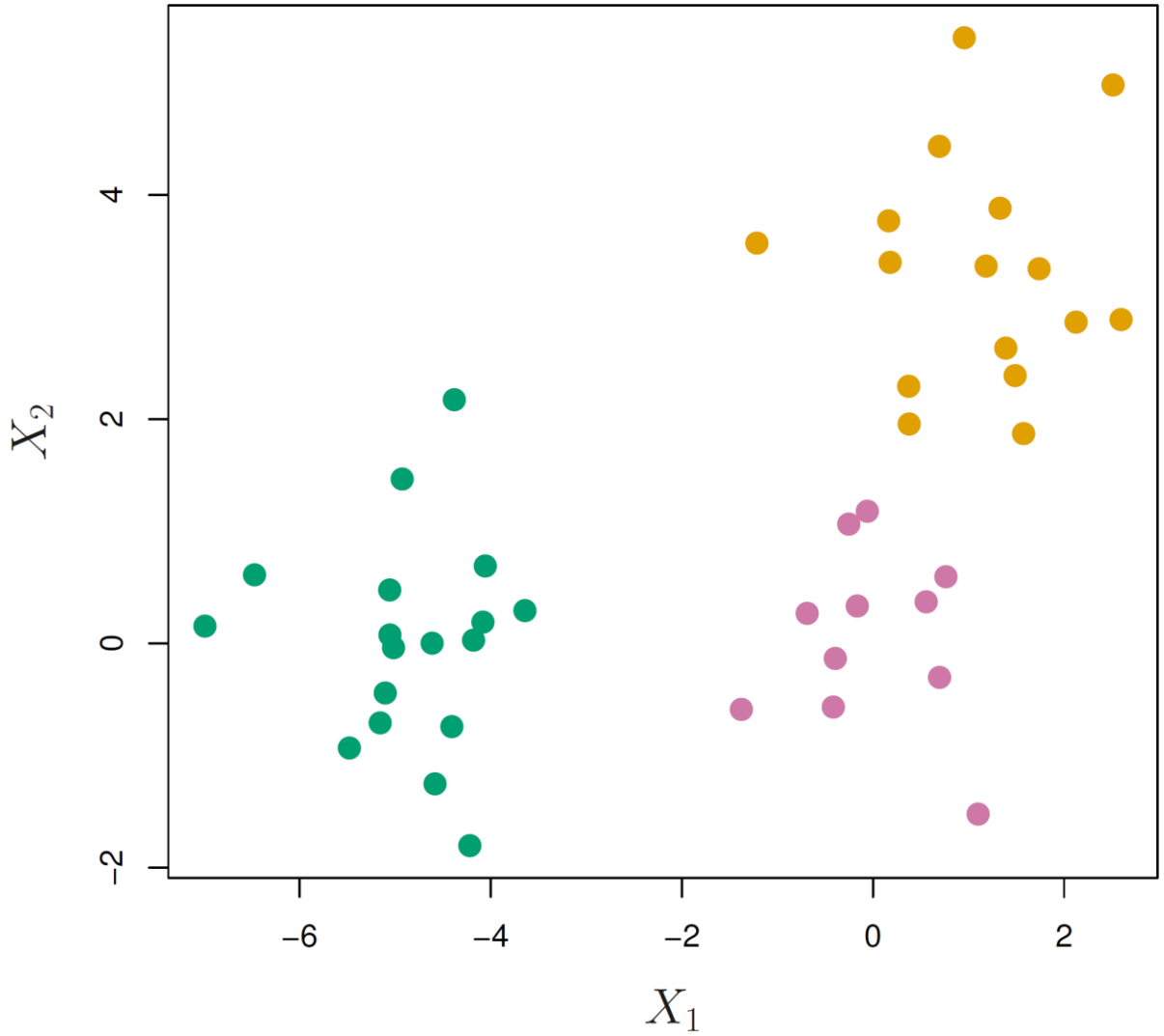
- The approach in words:
  - Start with each point in its own cluster
  - Identify the closest two clusters and merge them
  - Repeat
  - Ends when all points are in a single cluster



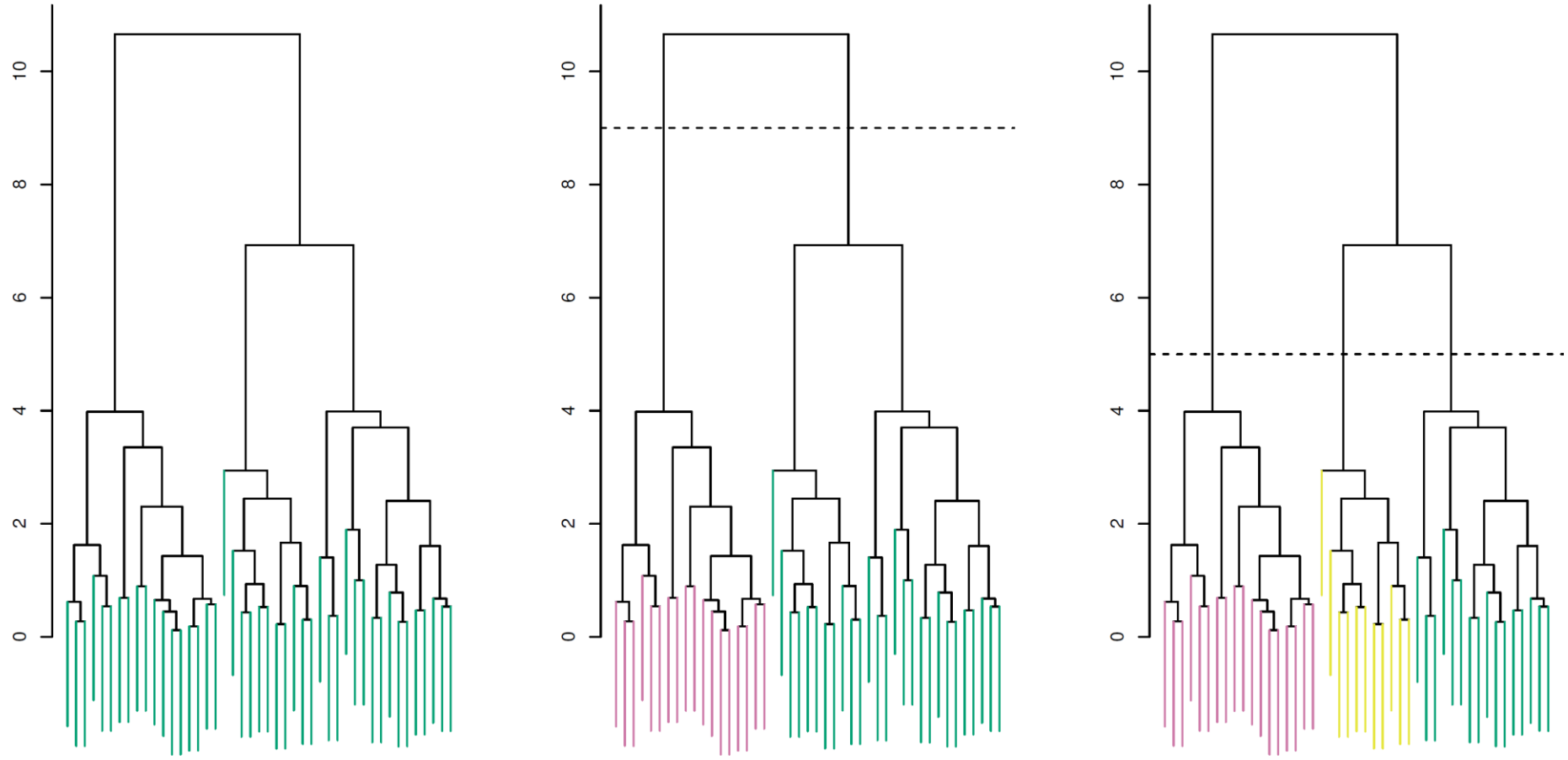


An Example

- 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

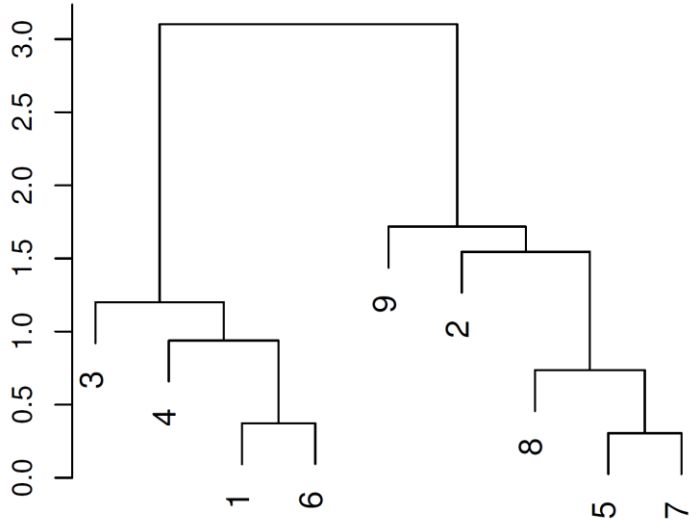


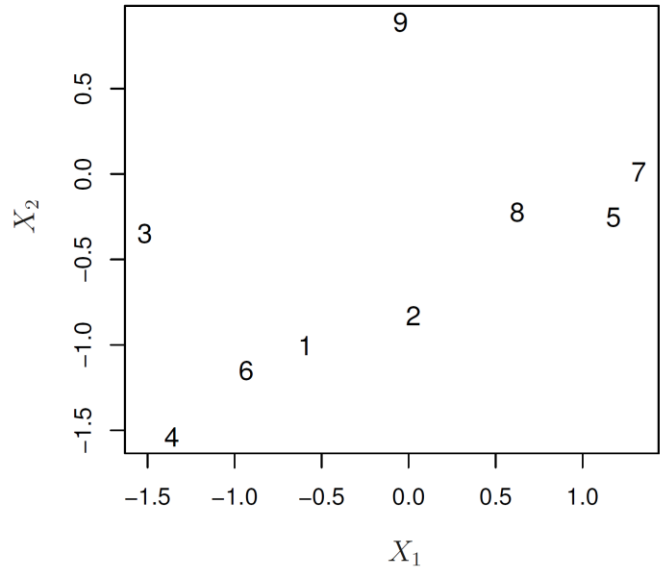
Details of previous figure

- 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

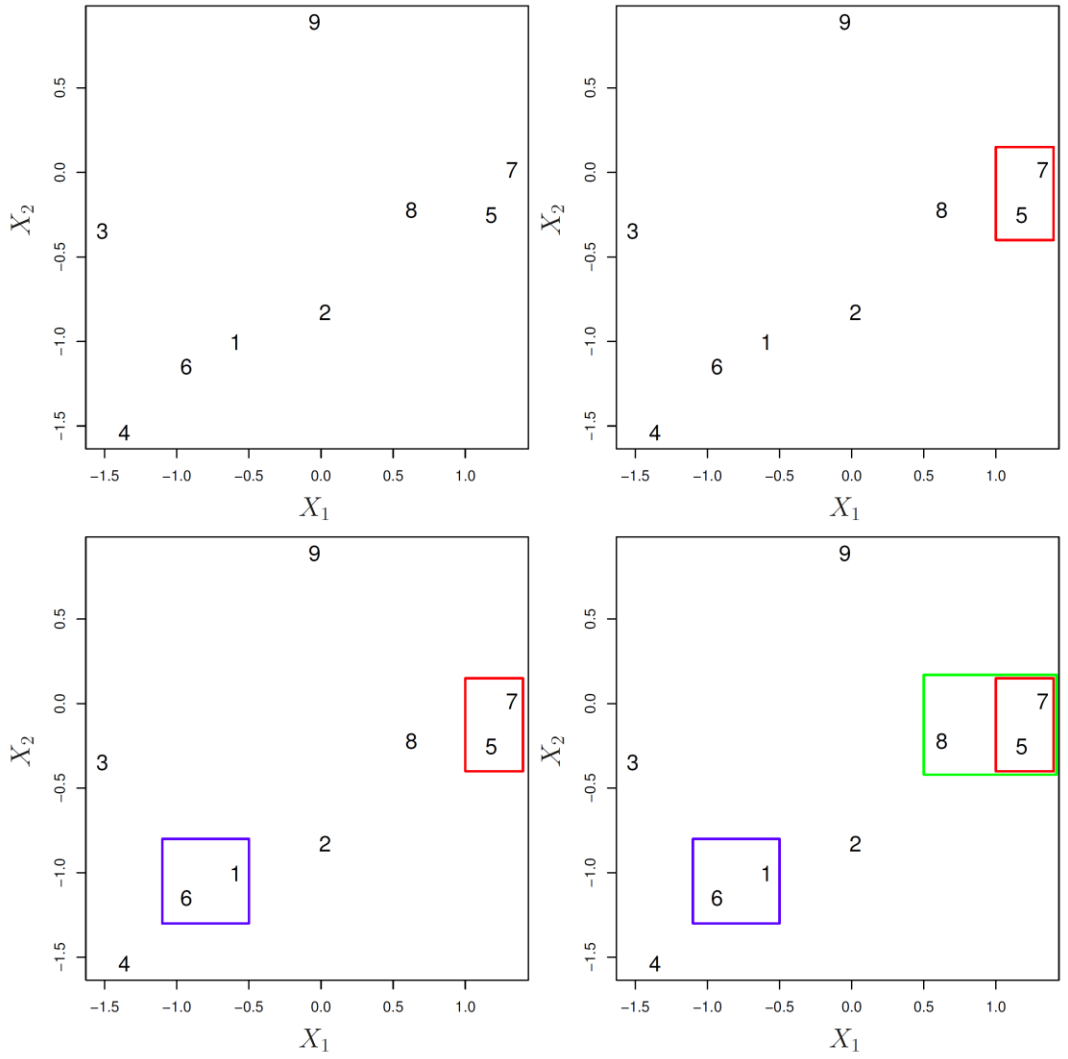
Another Example

- 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



Hierarchical Clustering

Algorithm 12.3 *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 $i = n, n-1, \ldots, 2$:

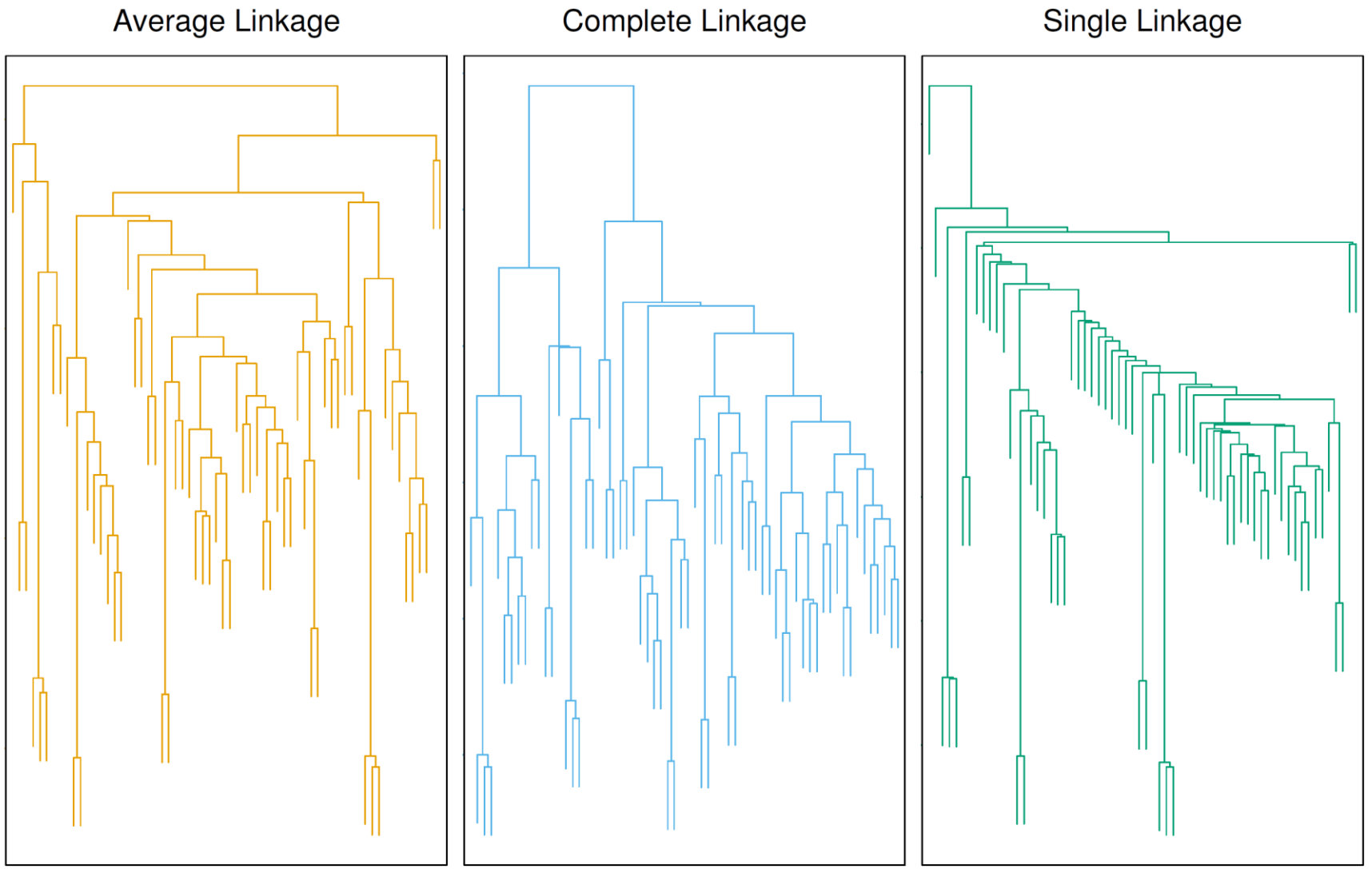
   (a) Examine all pairwise inter-cluster dissimilarities among the i clusters and identify the pair of clusters that are least dissimilar (that is, most similar). Fuse these two clusters. The dissimilarity between these two clusters indicates the height in the dendrogram at which the fusion should be placed.

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

Types of Linkage

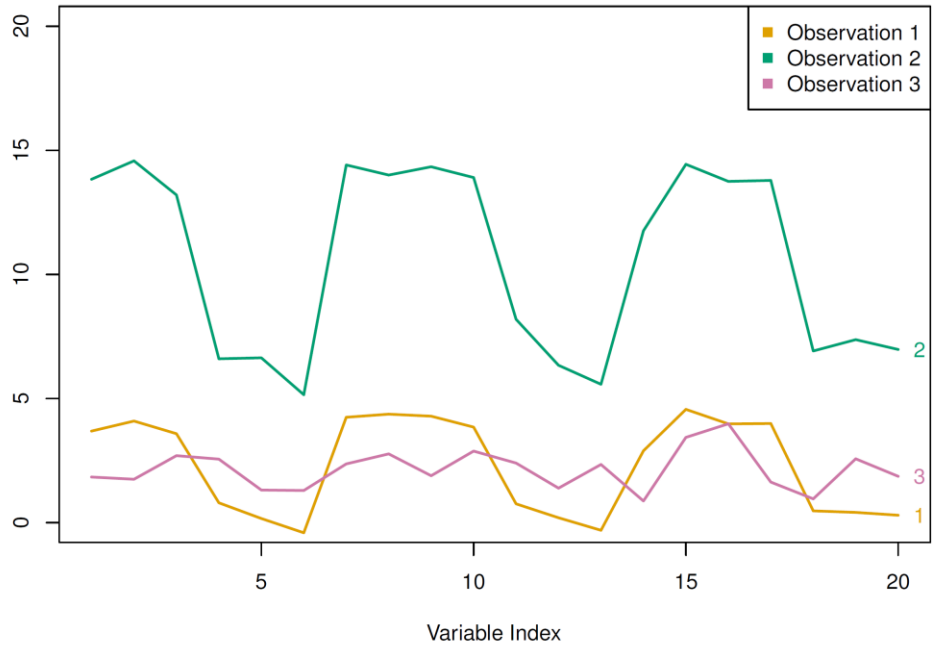
Linkage	Description
Complete	Maximal intercluster dissimilarity. Compute all pairwise dissimilarities between the observations in cluster A and the observations in cluster B, and record the *largest* of these dissimilarities
Single	Minimal intercluster dissimilarity. Compute all pairwise dissimilarities between the observations in cluster A and the observations in cluster B, and record the *smallest* of these dissimilarities. Single linkage can result in extended, trailing clusters in which single observations are fused one-at-a-time
Average	Mean intercluster 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

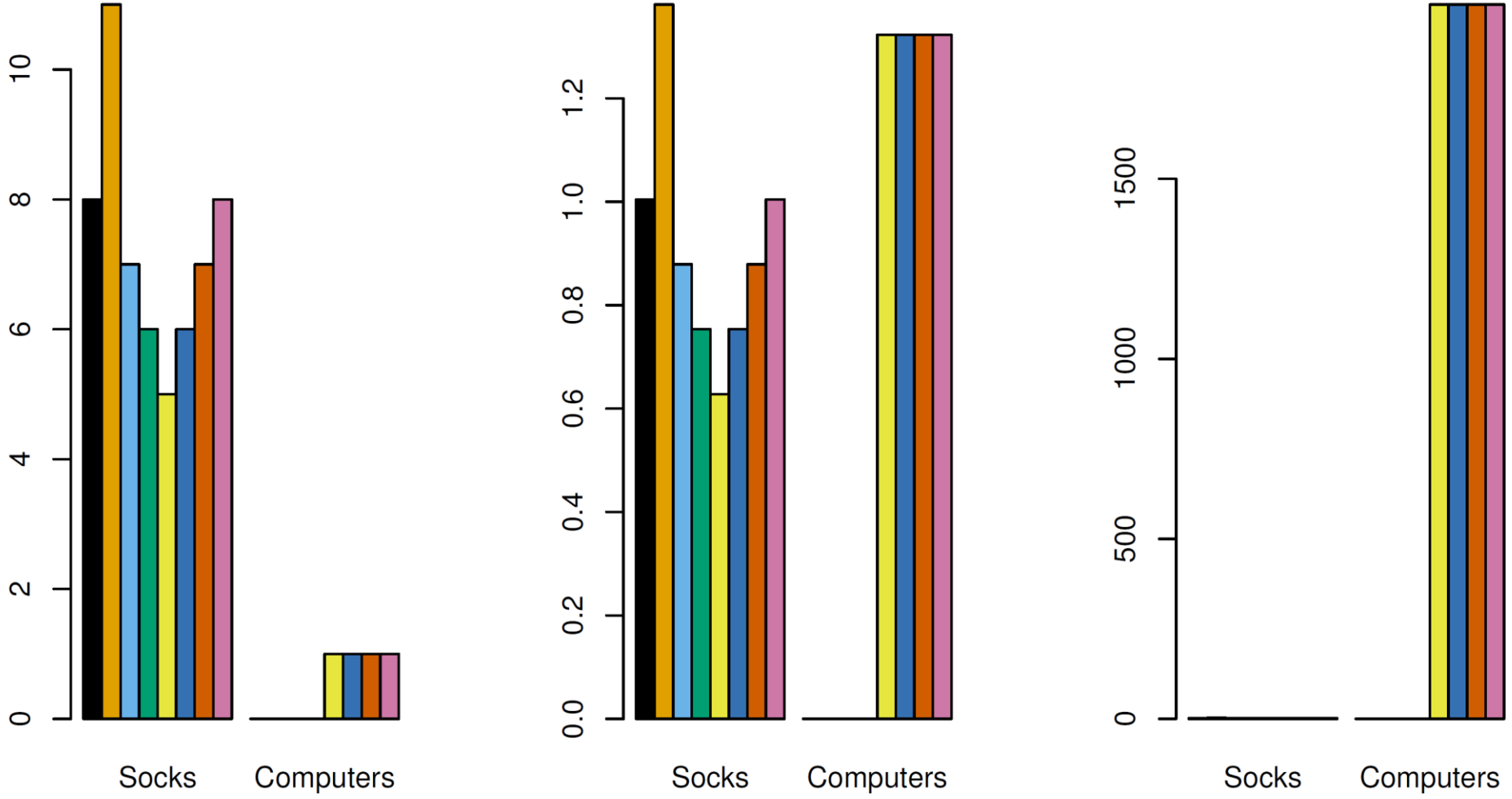


Choice of Dissimilarity Measure

- 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 proles 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

Practical issues - Validating the Clusters Obtained

- Any time clustering is performed on a data set we will find clusters. But we really want to know whether the clusters that have been found represent true subgroups in the data, or whether they are simply a result of clustering the noise
  - For instance, if we were to obtain an independent set of observations, then would those observations also display the same set of clusters?
- This is a hard question to answer. There exist a number of techniques for assigning a p-value to a cluster in order to assess whether there is more evidence for the cluster than one would expect due to chance. However, there has been no consensus on a single best approach.
  - More details can be found in ESL

Practical issues

- Both K-means and hierarchical clustering will assign each observation to a cluster. However, sometimes this might not be appropriate
  - Suppose a small subset of the observations are quite different from each other and from all other observations. The clusters found may be heavily distorted due to the presence of *outliers* that do not belong to any cluster
  - Mixture models are an attractive approach for accommodating the presence of such outliers. These amount to a soft version of K-means clustering, and are described in ESL
- Clustering methods generally are not very robust to perturbations to the data. For instance, suppose that we cluster n observations, and then cluster the observations again after removing a subset of the n observations at random. One would hope that the two sets of clusters obtained would be quite similar, but often this is not the case!

Practical issues

- Clustering can be a very useful and valid statistical tool if used properly
- We mentioned that small decisions in how clustering is performed
  - Such as how the data are standardized and what type of linkage is used, can have a large effect on the results
  - Therefore, we recommend performing clustering with different choices of these parameters, and looking at the full set of results in order to see what patterns consistently emerge
  - We must be careful about how the results of a clustering analysis are reported. These results should not be taken as the absolute truth about a data set. Rather, they should constitute a starting point for the development of a scientific hypothesis and further study, preferably on an independent data set

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

Conclusions

- Going further
  - Manifold learning
  - Self-supervised learning
  - Deep learning
  - Graph models
  - Bayesian data analysis
  - …
- Practical topics
  - Preprocessing
  - Transfer learning
  - High performance computing
  - Database
  - …

Appendix

Manifold learning

- https://scikit-learn.org/stable/modules/manifold.html

Spectral clustering

- https://jlmelville.github.io/smallvis/spectral.html

Mixture models

- https://cs229.stanford.edu/syllabus.html
- https://cs229.stanford.edu/notes2021fall/cs229-notes7b.pdf

Self-supervised learning

- https://speech.ee.ntu.edu.tw/~hylee/ml/ml2021-course-data/bert_v8.pdf