MY474: Applied Machine Learning for Social Science

Lecture 9: Unsupervised Learning

21 March 2022

Weeks 9 - 11

1. Bagging, random forests, and boosting
2. **Unsupervised learning**
3. Neural networks

Today

1. Introduction
2. Dimension Reduction
3. Clustering
4. Guided coding

Introduction

Machine Learning

1. Supervised learning
2. Unsupervised learning
3. Reinforcement learning

Unsupervised Learning

Unsupervised vs Supervised Learning:

- Most of this course focused on **supervised learning** methods such as regression and classification.
- In that setting we observe both a set of features $X_1, X_2, \dots, X_p$ for each object, as well as a response or outcome variable Y. The goal is then to predict Y using $X_1, X_2, \dots, X_p$.
- Here we instead focus on **unsupervised learning**, where we observe only the features $X_1, X_2, \dots, X_p$. We do not have an associated response variable Y that can *supervise* the learning.

The Goals of Unsupervised Learning

- Find pattern and structure in the data:
  - Can we discover subgroups among the variables or among the observations?
  - Can we represent the data in a more concise or useful way?
- We discuss two approaches:
  - **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 Particularities 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.
- Yet, there are much larger amounts of unlabeled data available.
- Furthermore, techniques for unsupervised learning are of importance for use cases in a number of fields, e.g.:
    - subgroups of 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,
    - recommendation systems.

Dimension Reduction

Problems with High Dimensional Data

- Correlated and noisy features
  - Ideally we would like our features to each convey independent information.
  - In reality we often have redundant or superfluous features.
  - In some supervised learning applications the goal is to extract the main components out a range of noisy features and then use them as regressors.
- Computational complexity
  - We are often limited by the computational complexity of classifiers.
  - With large p data, we may have very long training times.
- Cannot be easily visualised

Principal Components Analysis

- Principal Component Analysis (PCA) yields a low-dimensional representation of a dataset. It finds derived variables which are **linear combinations** of the old variables, have **maximal variance**, and are **mutually uncorrelated**.
- These principal components are then e.g. used as inputs in supervised learning models, or to visualise data in 2D or 3D.
- Sidenote: For **visualisation** of high dimensional data, also see methods such as t-SNE (van der Maaten and Hinton, 2008) which are non-linear mappings from the high dimensional space to lower dimensions, and are built to better preserve cluster structures for visualisation. Yet, the output of t-SNE is for visualisation only

Details and Notation

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 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.
- 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,
  $$\phi_1 = (\phi_{11} \phi_{21} \ldots \phi_{p1})^T.$$

Details and Notation: Continued

- The realized values of a principal component, e.g. Z_1, are referred to as *scores* $z_{11}, \dots, z_{n1}$.
- As an example, let us assume we have a $n \times 3$ data set $\mathbf{X}$.
- Since principal components are linear combinations, the first two prinicipal component scores after computation would have the following form:

$$\mathbf{X} = \begin{pmatrix} x_{11} & x_{12} & x_{13} \\ x_{21} & x_{22} & x_{23} \\ \dots & \dots & \dots \\ x_{n1} & x_{n2} & x_{n3} \end{pmatrix}$$

$$\mathbf{Z_2} = \begin{pmatrix} z_{11} & z_{12} \\ z_{21} & z_{22} \\ \dots & \dots \\ z_{n1} & z_{n2} \end{pmatrix} = \begin{pmatrix} \phi_{11}x_{11} + \phi_{21}x_{12} + \phi_{31}x_{13} & \phi_{12}x_{11} + \phi_{22}x_{12} + \phi_{32}x_{13} \\ \phi_{11}x_{21} + \phi_{21}x_{22} + \phi_{31}x_{23} & \phi_{12}x_{21} + \phi_{22}x_{22} + \phi_{32}x_{23} \\ \dots & \dots \\ \phi_{11}x_{n1} + \phi_{21}x_{n2} + \phi_{31}x_{n3} & \phi_{12}x_{n1} + \phi_{22}x_{n2} + \phi_{32}x_{n3} \end{pmatrix}$$

- Next, let us look at how the loading vectors ϕ_j, which characterise the principal components, are actually obtained.

Computation of First Principal Components

- Suppose we have a generic $n \times p$ 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

  $$z_{i1} = \phi_{11}x_{i1} + \phi_{21}x_{i2} + \cdots + \phi_{p1}x_{ip} \text{ for } i = 1, \ldots, n$$

  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 of First Principal Components: Maximisation Problem

- Plugging in z_{i1} from the previous slide, the first principal component loading vector solves the optimization problem

$$\underset{\phi_{11},\ldots,\phi_{p1}}{\text{maximize}} \frac{1}{n}\sum_{i=1}^{n}\left(\underbrace{\sum_{j=1}^{p}\phi_{j1}x_{ij}}_{z_{i1}}\right)^2 \text{ subject to } \sum_{j=1}^{p}\phi_{j1}^2 = 1$$

- The problem can be solved via a **singular-value decomposition** of the de-meaned matrix $\mathbf{X}$, a technique in linear algebra.

Computation of Further Principal Components

- The second principal component is the linear combination of $X_1, \ldots, X_p$ that has maximal variance subject to $\sum_{j=1}^{p} \phi_{j2}^2 = 1$ and subject to being orthogonal (perpendicular) to the first loading vector, i.e. $\phi_1 \cdot \phi_2 = 0$.
- 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 $\phi_{12}, \phi_{22}, \ldots, \phi_{p2}$.
- The third principal component is the linear combination of $X_1, \ldots, X_p$ that has maximal variance among all linear combinations subject to $\sum_{j=1}^{p} \phi_{j3}^2 = 1$ and under the constraint of being orthogonal to ϕ_1 and ϕ_2. And so on.

Linear Algebra Terminology

- 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.

Geometry of PCA: One Interpretation

- The loading vector ϕ_1 defines a direction in feature space along which the data vary the most. Subsequent loading vectors are orthogonal to this vector.
- 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.

Geometry of PCA: Example

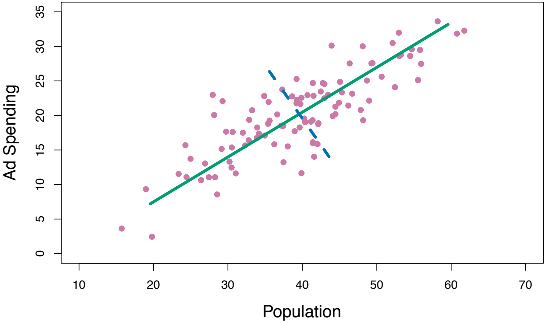


Figure 1: The population size (pop) and ad spending (ad) for 100 different cities are shown as purple circles. The green solid line indicates the first principal component direction, and the blue dashed line indicates the second principal component direction.

Geometry of PCA: Another Interpretation

- PCA finds a lower dimensional surface that is 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) - see again the previous figure
- 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.

Geometry of PCA: Example

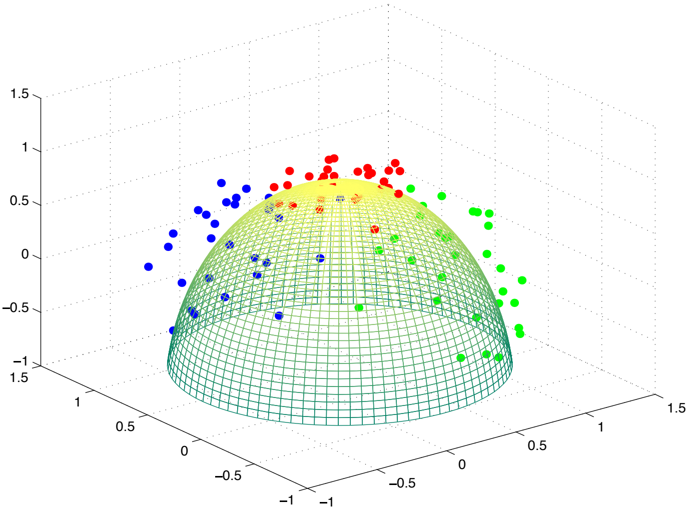


Figure 2: Simulated data in three classes, near the surface of a half–sphere.

Geometry of PCA: Example

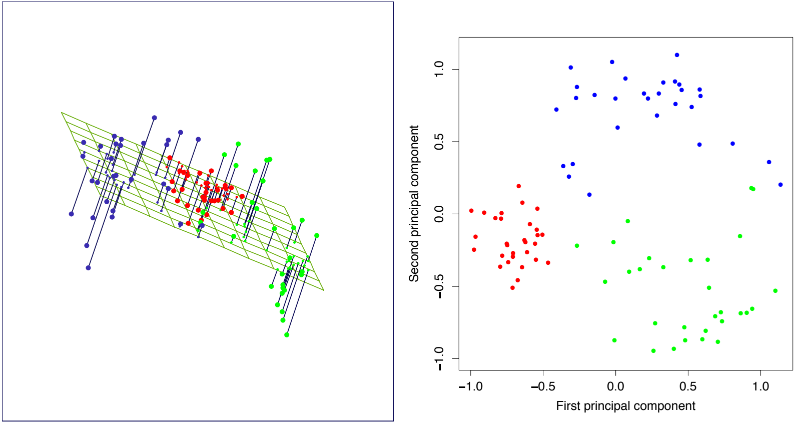


Figure 3: The best rank-two linear approximation to the half-sphere data. The right panel shows the projected points with coordinates given by the first two principal components of the data (source: ESL p. 536)

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.

PCA Loadings

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

USAarrests Data: PCA Plot

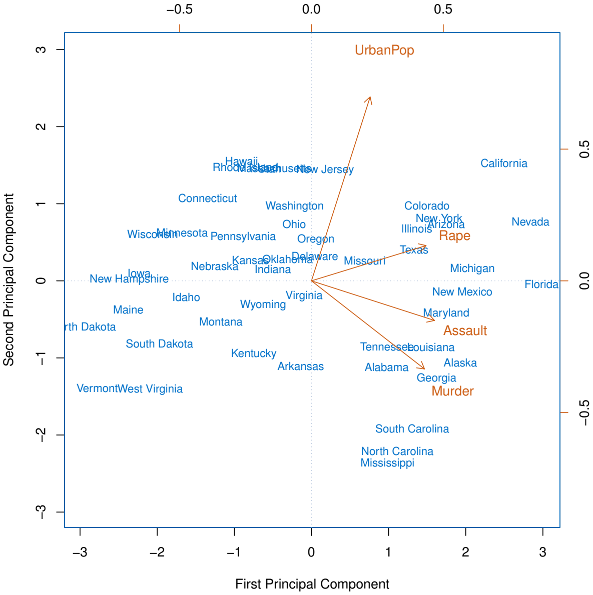


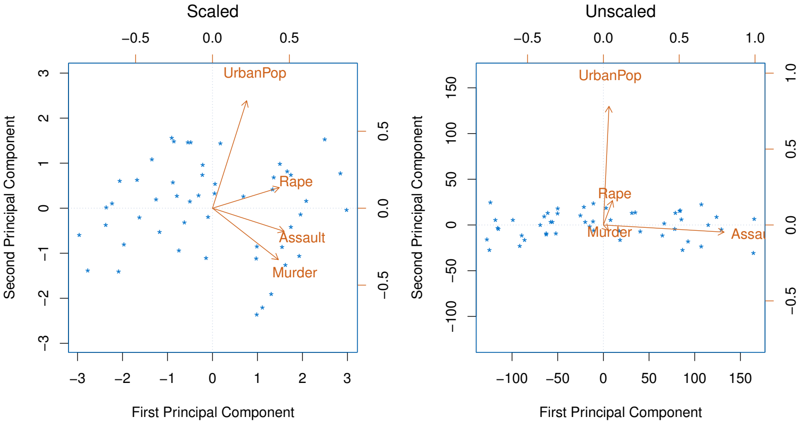
Figure Details

The figure plots the first two principal component scores for each observation in 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 loadings (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.

Scaling of the Variables Matters

- If the variables are in different units, scaling each to have standard deviation equal to one is recommended (i.e. divide each variable by its standard deviation).
- 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

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

and the variance explained by the mth principal component is

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

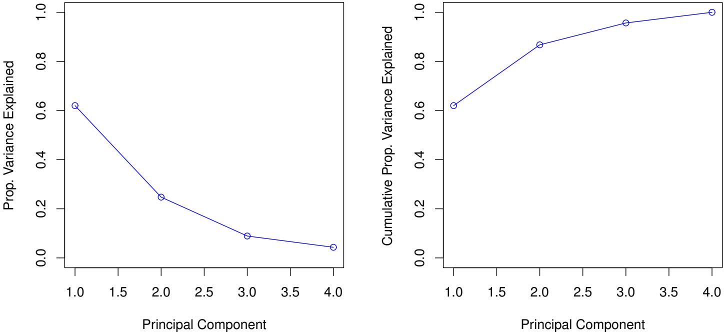
- It can be shown that $\sum_{j=1}^{p} Var(X_j) = \sum_{m=1}^{M} Var(Z_m)$, with $M = min(n-1, p)$.

Proportion Variance Explained: continued

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

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

- 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 PCA in itself.
- The **“scree plot”** on the previous slide can be used as a guide: we look for an **“elbow”**.
- If principal components are used as inputs to supervised learning, however, we could use cross-validation to select the number of components relative to minimising some loss for the prediction model.

Clustering

Clustering

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

Dimension Reduction vs Clustering

- **PCA** typically reduces amount of columns in **X** in a way that new combined features explain a large fraction of the variance
- **Clustering** usually finds groups of observations, i.e. groups of rows in **X** (rarely groups of features)

Applications of Clustering

- **Market segmentation:** identify demographic or behavioral groups for targeted advertising or marketing campaigns.
- **Fraud detection:** identify election fraud, anomolous network activity, etc.
- **Discovering heterogeneity in treatment effects:** are certain groups more likely to respond to a drug?
- **Finding structures in texts** Clustering of documents.

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

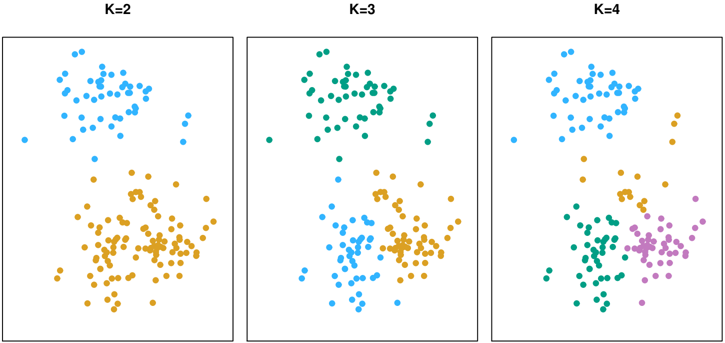


Figure 4: A simulated data set with 150 observations in 2-dimensional space. Panels show the results of applying K-means clustering with different values of K, the number of clusters. The color of each observation indicates the cluster to which it was assigned using the K-means clustering algorithm. 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. $C_1 \cup C_2 \cup \cdots \cup C_K = \{1, \ldots, n\}$. Inotherwords,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 $WCV(C_k)$ of the amount by which the observations within a cluster differ from each other.

- Hence we want to solve the problem

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

- 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 the squared Euclidean distance between all pairwise combination of observations in a cluster:

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

where $|C_k|$ denotes the number of observations in the kth cluster. - The optimization problem that defines K-means clustering is

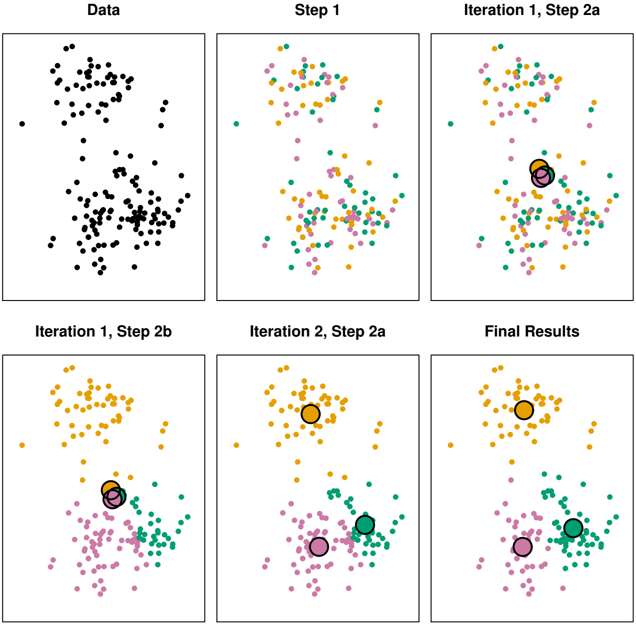
$$\underset{C_1,\ldots,C_K}{\text{minimize}} \left\{ \sum_{k=1}^{K} \frac{1}{|C_k|} \sum_{i,i' \in C_k} \sum_{j=1}^{p} (x_{ij} - x_{i'j})^2 \right\}$$

- This is a very complex problem, it is computationally infeasible to find the global optimum unless K and n are small. Instead: Use algorithm which can be shown to yield a local optimum.

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:
   - 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.
   - Assign each observation to the cluster whose centroid is closest (where closest is defined using Euclidean distance).

Algorithm Illustration



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.

Properties of the Algorithm

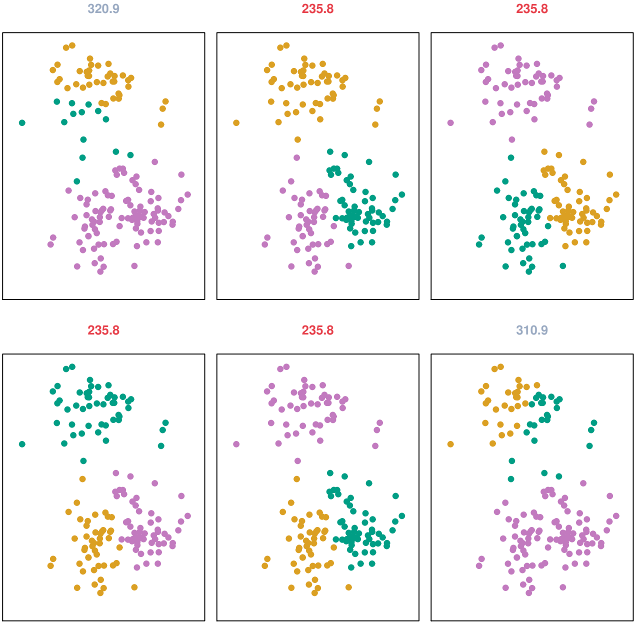
- This algorithm is guaranteed to decrease the value of the objective function at each step until convergence. To see this, note the following identity:

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

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

- However, it will yield a local optimum. This is why trying different pseudo random starting values is important.

Example: Different Starting Values



320.9

235.8

235.8

235.8

235.8

310.9

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

- Next, we are going to talk about 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

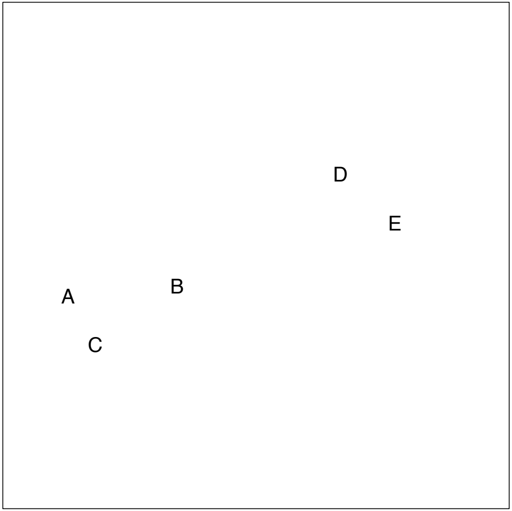


Figure 5: Builds a hierarchy in a "bottom-up" fashion

Hierarchical Clustering: The Idea

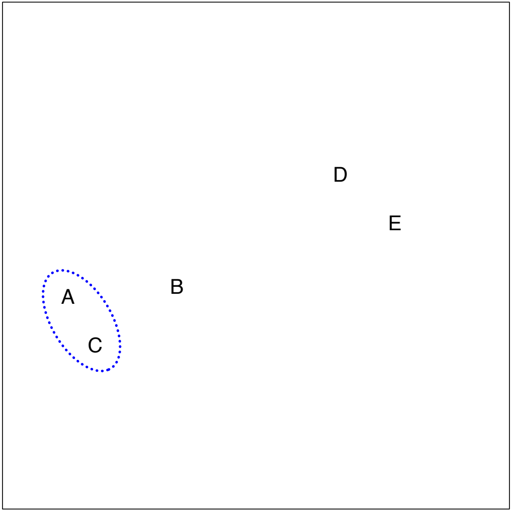


Figure 6: Builds a hierarchy in a “bottom-up” fashion

Hierarchical Clustering: The Idea

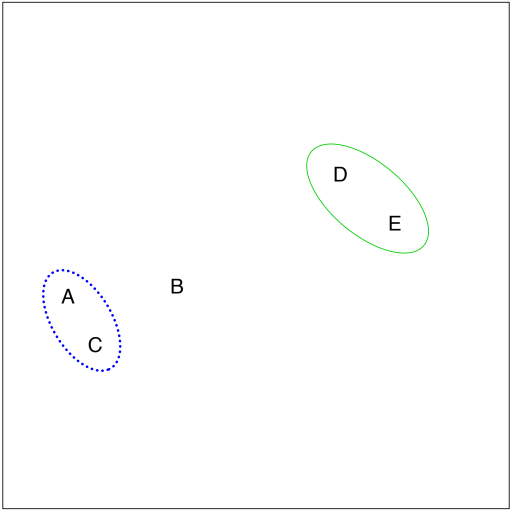


Figure 7: Builds a hierarchy in a "bottom-up" fashion

Hierarchical Clustering: The Idea

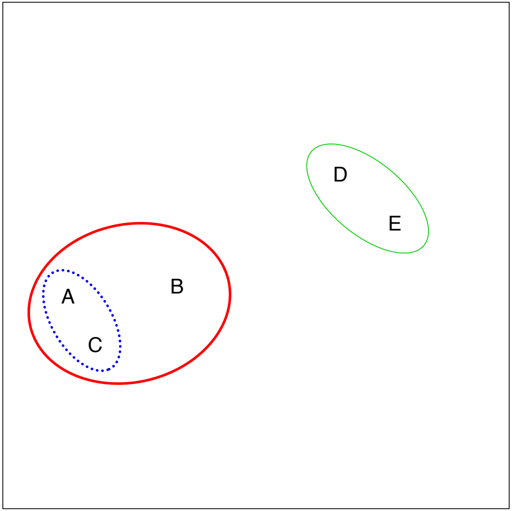


Figure 8: Builds a hierarchy in a “bottom-up” fashion

Hierarchical Clustering: The Idea

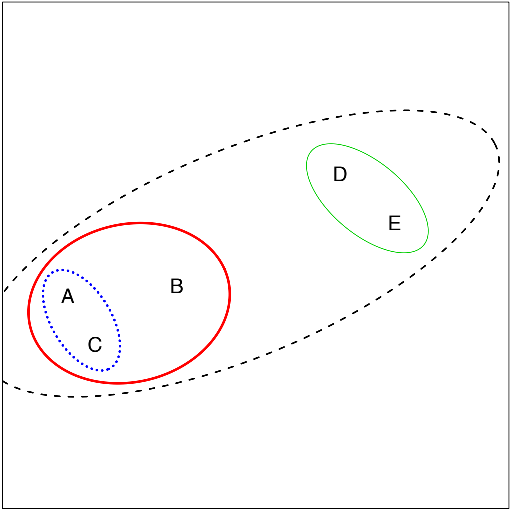
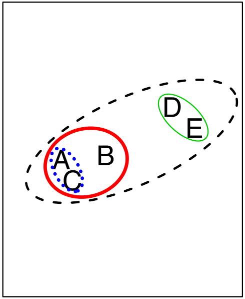


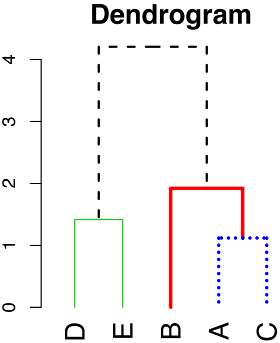
Figure 9: Builds a hierarchy in a “bottom-up” fashion

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.





An Example

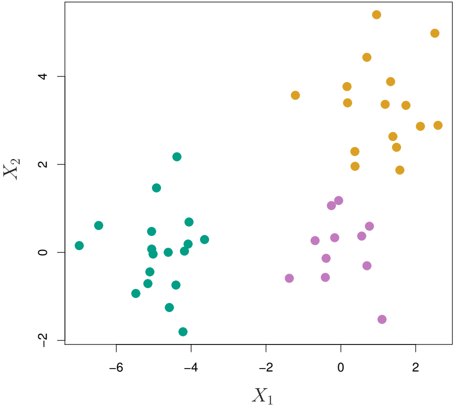


Figure 10: 45 observations generated in 2-dimensional space. In reality there are three distinct classes, shown in separate colors. However, we will treat these class labels as unknown and will seek to cluster the observations in order to discover the classes from the data.

Application of hierarchical clustering

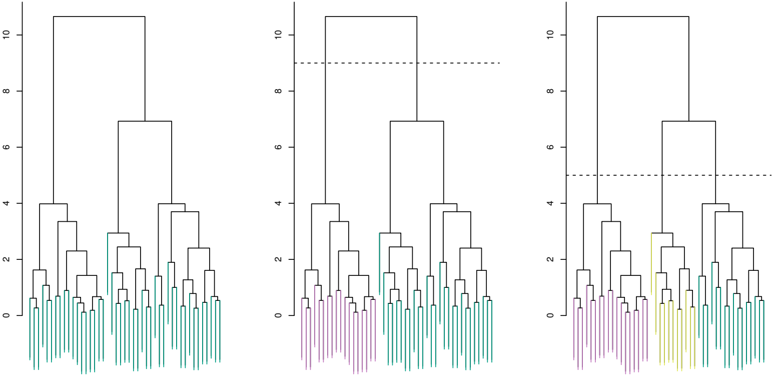
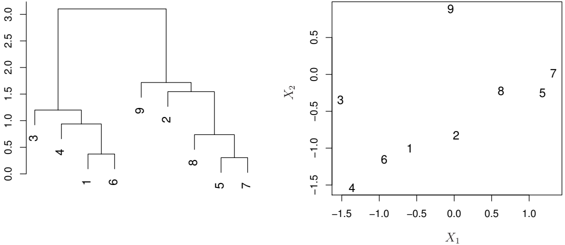


Figure 11: (see details in next slide)

Details of previous figure

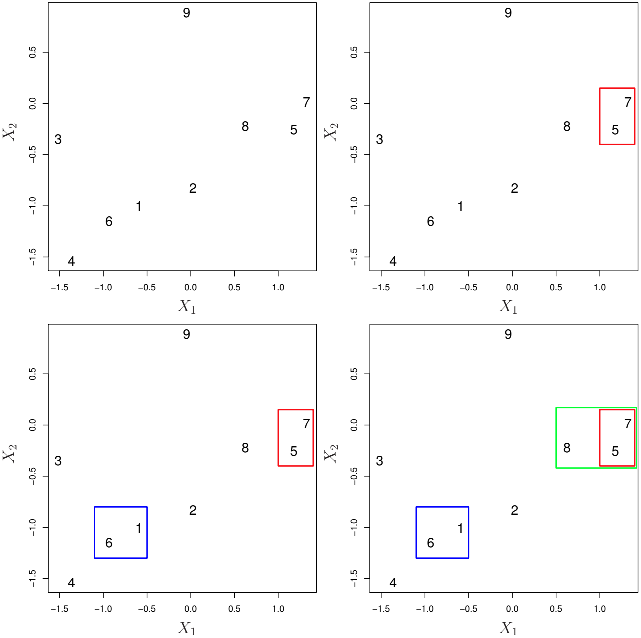
- Left: Dendrogram obtained from hierarchically clustering the data from previous slide, with complete linkage and Euclidean distance.
- Center: The dendrogram from the left-hand panel, cut at a height of 9 (indicated by the dashed line). This cut results in two distinct clusters, shown in different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

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.

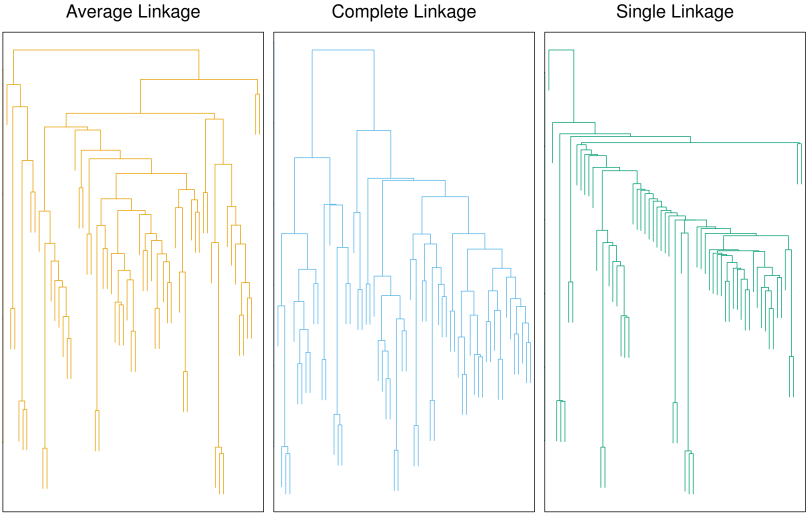
Merges in Previous Example



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 Linkage

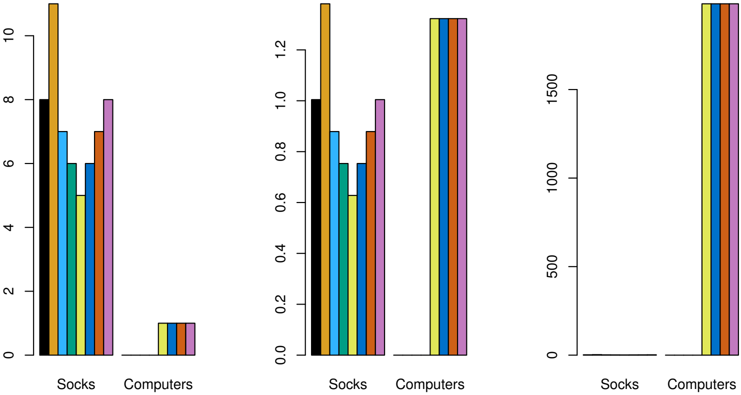


Average Linkage

Complete Linkage

Single Linkage

Scaling of The Variables Matters



Practical Issues

- **Scaling of the variables matters.** 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? We have used Euclidean distance here, but there are also others. What type of linkage should be used?
- How many cluster should be chosen? This is a difficult problem with no agreed-upon method. See Elements of Statistical Learning, chapter 13 for more details.
- Which features should we base the clustering on in a specific problem?

Choosing *k*

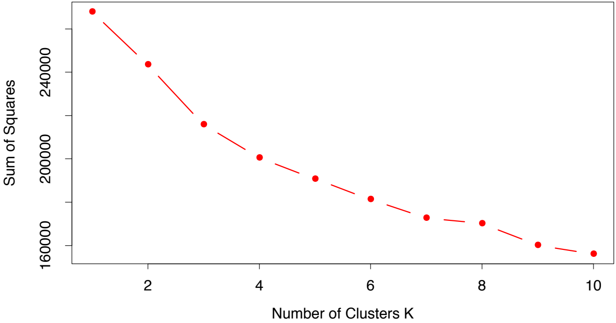


Figure 12: Total within-cluster sum of squares for K-means clustering applied to the human tumor microarray data. Typically we look for an elbow in the sum of squares curve, but there isn't a clear one here. (Source ESL p. 513)

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

Guided Coding

- 01-voting.Rmd