Statistical Learning– MATH 6333
Set 9 (Unsupervised Learning)

Tamer Oraby

UTRGV

tamer.oraby@utrgv.edu

* Last updated December 1, 2021

Unsupervised Learning

Training data: $\mathcal{T} = \{(x_{i1}, x_{i2}, \ldots, x_{ip}) : i = 1, 2, \ldots, n\}$

Exploratory data analysis: No output and no model function f.

Goal of Unsup.L.: To describe the relationship between the $x_{i1}, x_{i2}, \ldots, x_{ip}$ of training data $\{(x_{i1}, x_{i2}, \ldots, x_{ip}) : i = 1, 2, \ldots, n\}$

Testing: To describe the relationship between the $x_{j1}, x_{j2}, \ldots, x_{jp}$ of testing data $\{(x_{j1}, x_{j2}, \ldots, x_{jp}) : j = 1, 2, \ldots, m\}$ of a known relationship.

Ultimate Goal (Generalization): To describe a new $x_* = (x_{*1}, x_{*2}, \ldots, x_{*p})$ based on the learned relationships.

Unsupervised Learning

Training data: $\mathcal{T} = \{(x_{i1}, x_{i2}, \ldots, x_{ip}) : i = 1, 2, \ldots, n\}$

Exploratory data analysis: No output and no model function f.

Goal of Unsup.L.: To describe the relationship between the $x_{i1}, x_{i2}, \ldots, x_{ip}$ of training data $\{(x_{i1}, x_{i2}, \ldots, x_{ip}) : i = 1, 2, \ldots, n\}$

Testing: To describe the relationship between the $x_{j1}, x_{j2}, \ldots, x_{jp}$ of testing data $\{(x_{j1}, x_{j2}, \ldots, x_{jp}) : j = 1, 2, \ldots, m\}$ of a known relationship.

Ultimate Goal (Generalization): To describe a new $x_* = (x_{*1}, x_{*2}, \ldots, x_{*p})$ based on the learned relationships.

Unsupervised Learning

Training data: $\mathcal{T} = \{(x_{i1}, x_{i2}, \ldots, x_{ip}) : i = 1, 2, \ldots, n\}$

Exploratory data analysis: No output and no model function f.

Goal of Unsup.L.: To describe the relationship between the $x_{i1}, x_{i2}, \ldots, x_{ip}$ of training data $\{(x_{i1}, x_{i2}, \ldots, x_{ip}) : i = 1, 2, \ldots, n\}$

Testing: To describe the relationship between the $x_{j1}, x_{j2}, \ldots, x_{jp}$ of testing data $\{(x_{j1}, x_{j2}, \ldots, x_{jp}) : j = 1, 2, \ldots, m\}$ of a known relationship.

Ultimate Goal (Generalization): To describe a new $x_* = (x_{*1}, x_{*2}, \ldots, x_{*p})$ based on the learned relationships.

Unsupervised Learning

Training data: $\mathcal{T} = \{(x_{i1}, x_{i2}, \ldots, x_{ip}) : i = 1, 2, \ldots, n\}$

Exploratory data analysis: No output and no model function f.

Goal of Unsup.L.: To describe the relationship between the $x_{i1}, x_{i2}, \ldots, x_{ip}$ of training data $\{(x_{i1}, x_{i2}, \ldots, x_{ip}) : i = 1, 2, \ldots, n\}$

Testing: To describe the relationship between the $x_{j1}, x_{j2}, \ldots, x_{jp}$ of testing data $\{(x_{j1}, x_{j2}, \ldots, x_{jp}) : j = 1, 2, \ldots, m\}$ of a known relationship.

Ultimate Goal (Generalization): To describe a new $x_* = (x_{*1}, x_{*2}, \ldots, x_{*p})$ based on the learned relationships.

Unsupervised Learning

Training data: $\mathcal{T} = \{(x_{i1}, x_{i2}, \ldots, x_{ip}) : i = 1, 2, \ldots, n\}$

Exploratory data analysis: No output and no model function f.

Goal of Unsup.L.: To describe the relationship between the $x_{i1}, x_{i2}, \ldots, x_{ip}$ of training data $\{(x_{i1}, x_{i2}, \ldots, x_{ip}) : i = 1, 2, \ldots, n\}$

Testing: To describe the relationship between the $x_{j1}, x_{j2}, \ldots, x_{jp}$ of testing data $\{(x_{j1}, x_{j2}, \ldots, x_{jp}) : j = 1, 2, \ldots, m\}$ of a known relationship.

Ultimate Goal (Generalization): To describe a new $x_* = (x_{*1}, x_{*2}, \ldots, x_{*p})$ based on the learned relationships.

Unsupervised Learning

- Cluster Analysis
- Hierarchical Clustering
- Principal Component Analysis (PCA)

Cluster Analysis

Cluster Analysis

- Goal: To partition the data space into a pre-determined number of K clusters $\mathcal{C} = \{C_1, \ldots, C_K\}$
- The idea is that some data points are clustered with each others more than other points.
- Those points can be described to be cluttered around a cluster centroid c_k for $k = 1, 2, \ldots, K$. They could be $c_k = \mu_k$ the mean in the K-means OR $c_k = X_{i(k)}$ one of the data points in the K-medoid.

Q How to measure the closeness to the centroid?

A using the sum of squares of errors

$$SSE(\mathcal{C}) = \sum_{k=1}^{K} \sum_{X_i \in C_k} \underbrace{||X_i - \mu_k||^2}_{\text{dissimilarity function}}$$

to be minimized.

Cluster Analysis

- Goal: To partition the data space into a pre-determined number of K clusters $\mathcal{C} = \{C_1, \ldots, C_K\}$
- The idea is that some data points are clustered with each others more than other points.
- Those points can be described to be cluttered around a cluster centroid c_k for $k = 1, 2, \ldots, K$. They could be $c_k = \mu_k$ the mean in the K-means OR $c_k = X_{i(k)}$ one of the data points in the K-medoid.

Q How to measure the closeness to the centroid?

A using the sum of squares of errors

$$SSE(\mathcal{C}) = \sum_{k=1}^{K} \sum_{X_i \in C_k} \underbrace{||X_i - \mu_k||^2}_{\text{dissimilarity function}}$$

to be minimized.

Cluster Analysis

- Goal: To partition the data space into a pre-determined number of K clusters $\mathcal{C} = \{C_1, \ldots, C_K\}$
- The idea is that some data points are clustered with each others more than other points.
- Those points can be described to be cluttered around a cluster centroid c_k for $k = 1, 2, \ldots, K$. They could be $c_k = \mu_k$ the mean in the K-means OR $c_k = X_{i(k)}$ one of the data points in the K-medoid.

Q How to measure the closeness to the centroid?

A using the sum of squares of errors

$$SSE(\mathcal{C}) = \sum_{k=1}^{K} \sum_{X_i \in C_k} \underbrace{||X_i - \mu_k||^2}_{\text{dissimilarity function}}$$

to be minimized.

Cluster Analysis

- Goal: To partition the data space into a pre-determined number of K clusters $\mathcal{C} = \{C_1, \ldots, C_K\}$
- The idea is that some data points are clustered with each others more than other points.
- Those points can be described to be cluttered around a cluster centroid c_k for $k = 1, 2, \ldots, K$. They could be $c_k = \mu_k$ the mean in the K-means OR $c_k = X_{i(k)}$ one of the data points in the K-medoid.

Q How to measure the closeness to the centroid?

A using the sum of squares of errors

$$SSE(\mathcal{C}) = \sum_{k=1}^{K} \sum_{X_i \in C_k} \underbrace{||X_i - \mu_k||^2}_{\text{dissimilarity function}}$$

to be minimized.

Cluster Analysis

- Goal: To partition the data space into a pre-determined number of K clusters $\mathcal{C} = \{C_1, \ldots, C_K\}$
- The idea is that some data points are clustered with each others more than other points.
- Those points can be described to be cluttered around a cluster centroid c_k for $k = 1, 2, \ldots, K$. They could be $c_k = \mu_k$ the mean in the K-means OR $c_k = X_{i(k)}$ one of the data points in the K-medoid.

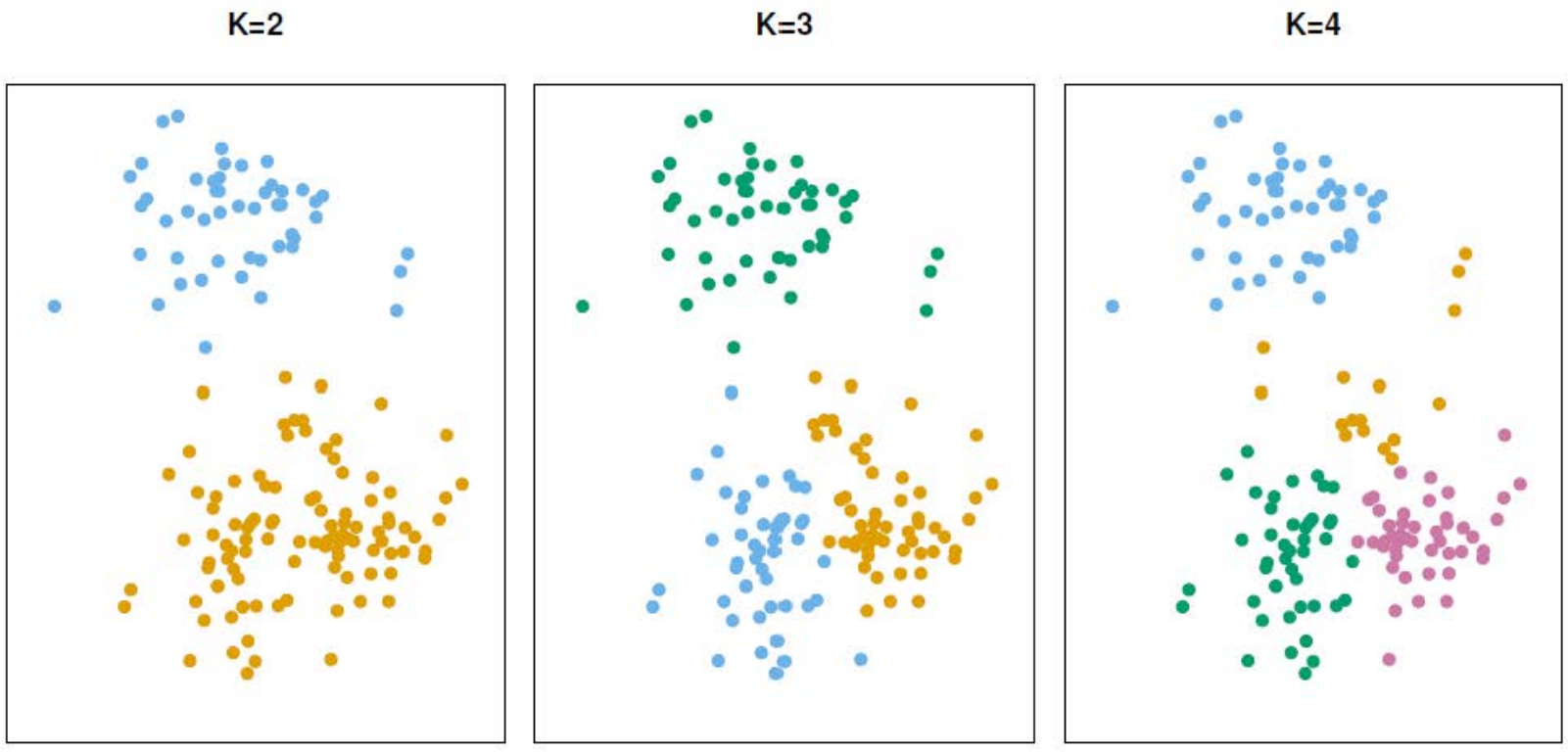
Q How to measure the closeness to the centroid?

A using the sum of squares of errors

$$SSE(\mathcal{C}) = \sum_{k=1}^{K} \sum_{X_i \in C_k} \underbrace{||X_i - \mu_k||^2}_{\text{dissimilarity function}}$$

to be minimized.

Cluster Analysis



Cluster Analysis

- Other distance functions or (dis)similarity functions are also available to be used in the error function.
  - Euclidean distance:

$$d_2(x, y) = ||x - y||_2$$

  - Manhattan distance:

$$d_1(x, y) = ||x - y||_1$$

  - Correlation (Pearson or Spearman)-based distance:

$$d_{corr}(x, y) = 1 - Corr(x, y)$$

    If x and y are standardized then it is called Eisen-cosine correlation distance.

  They are almost the same if the data is standardized which is advised to do at the beginning of the analyses.

Cluster Analysis

- Other distance functions or (dis)similarity functions are also available to be used in the error function.
  - Euclidean distance:

$$d_2(x, y) = ||x - y||_2$$

  - Manhattan distance:

$$d_1(x, y) = ||x - y||_1$$

  - Correlation (Pearson or Spearman)-based distance:

$$d_{corr}(x, y) = 1 - Corr(x, y)$$

    If x and y are standardized then it is called Eisen-cosine correlation distance.

They are almost the same if the data is standardized which is advised to do at the beginning of the analyses.

Cluster Analysis

- Other distance functions or (dis)similarity functions are also available to be used in the error function.
  - Euclidean distance:

$$d_2(x, y) = ||x - y||_2$$

  - Manhattan distance:

$$d_1(x, y) = ||x - y||_1$$

  - Correlation (Pearson or Spearman)-based distance:

$$d_{corr}(x, y) = 1 - Corr(x, y)$$

    If x and y are standardized then it is called Eisen-cosine correlation distance.

They are almost the same if the data is standardized which is advised to do at the beginning of the analyses.

Cluster Analysis

- Other distance functions or (dis)similarity functions are also available to be used in the error function.
  - Euclidean distance:

$$d_2(x, y) = ||x - y||_2$$

  - Manhattan distance:

$$d_1(x, y) = ||x - y||_1$$

  - Correlation (Pearson or Spearman)-based distance:

$$d_{corr}(x, y) = 1 - Corr(x, y)$$

    If x and y are standardized then it is called Eisen-cosine correlation distance.

They are almost the same if the data is standardized which is advised to do at the beginning of the analyses.

Cluster Analysis

- Other distance functions or (dis)similarity functions are also available to be used in the error function.
  - Euclidean distance:

$$d_2(x, y) = ||x - y||_2$$

  - Manhattan distance:

$$d_1(x, y) = ||x - y||_1$$

  - Correlation (Pearson or Spearman)-based distance:

$$d_{corr}(x, y) = 1 - Corr(x, y)$$

    If x and y are standardized then it is called Eisen-cosine correlation distance.

They are almost the same if the data is standardized which is advised to do at the beginning of the analyses.

Cluster Analysis

- Other distance functions or (dis)similarity functions are also available to be used in the error function.
  - Euclidean distance:

$$d_2(x, y) = ||x - y||_2$$

  - Manhattan distance:

$$d_1(x, y) = ||x - y||_1$$

  - Correlation (Pearson or Spearman)-based distance:

$$d_{corr}(x, y) = 1 - Corr(x, y)$$

    If x and y are standardized then it is called Eisen-cosine correlation distance.

They are almost the same if the data is standardized which is advised to do at the beginning of the analyses.

Cluster Analysis

- Jensen-Shannon divergence:

$$d_{JS}(x, y) = \frac{1}{2}(d_{KL}(x, y) + d_{KL}(y, x))$$

where

$$d_{KL}(x, y) = \sum_{i=1}^{n} x_i \log(\frac{x_i}{y_i})$$

is the Kullback-Leibler divergence (relative entropy) that measures the lost information when we use x to represent y.

Brute-Force (exhaustive) Algorithm

Cluster Analysis

Brute-Force (exhaustive) Algorithm

- Generate all of the ($\sim K^n/K!$) possible partitions $\mathcal{C}$.
- For each possible partition, calculate $\{\mu_k, k = 1, 2, \ldots, K\}$
- Calculate $SSE(\mathcal{C})$, for all $\mathcal{C}$'s
- The optimal cluster $\mathcal{C}^* = argmin_{\mathcal{C}} SSE(\mathcal{C})$

What do you think?

Cluster Analysis

Brute-Force (exhaustive) Algorithm

- Generate all of the ($\sim K^n/K!$) possible partitions $\mathcal{C}$.
- For each possible partition, calculate $\{\mu_k, k = 1, 2, \ldots, K\}$
- Calculate $SSE(\mathcal{C})$, for all $\mathcal{C}$'s
- The optimal cluster $\mathcal{C}^* = argmin_{\mathcal{C}} SSE(\mathcal{C})$

What do you think?

Cluster Analysis

Brute-Force (exhaustive) Algorithm

- Generate all of the ($\sim K^n/K!$) possible partitions $\mathcal{C}$.
- For each possible partition, calculate $\{\mu_k, k = 1, 2, \ldots, K\}$
- Calculate $SSE(\mathcal{C})$, for all $\mathcal{C}$'s
- The optimal cluster $\mathcal{C}^* = argmin_{\mathcal{C}} SSE(\mathcal{C})$

What do you think?

Cluster Analysis

Brute-Force (exhaustive) Algorithm

- Generate all of the ($\sim K^n/K!$) possible partitions $\mathcal{C}$.
- For each possible partition, calculate $\{\mu_k, k = 1, 2, \ldots, K\}$
- Calculate $SSE(\mathcal{C})$, for all $\mathcal{C}$'s
- The optimal cluster $\mathcal{C}^* = argmin_{\mathcal{C}} SSE(\mathcal{C})$

What do you think?

K-means Clustering Algorithm

Cluster Analysis

K-means Clustering Algorithm

It is a recursive greedy algorithm to minimize SSE.

Initialize Start with a randomly generated centroids $\{\mu_k, k = 1, 2, \ldots, K\}$

Iterative

- Assign X_i to cluster C_{k^*} for which

$$k^* = argmin_{1 \le k \le K} ||X_i - \mu_k||^2$$

- After assigning all of the $X_i's$, update the centroids $\{\mu_k, k = 1, 2, , K\}$ (averages)
- continue the iterations till $max_k ||\mu_k^{new} - \mu_k^{old}|| < TOL$

Repeat the whole algorithm several times for randomly generated initial centroids (Monte Carlo alg.) and find the smallest

$$SSE(\mathcal{C}) = \sum_{k=1}^{K} \sum_{X_i \in C_k} ||X_i - \mu_k||^2 = \sum_{k=1}^{K} \sum_{i=1}^{n} I(X_i \in C_k) ||X_i - \mu_k||^2$$

Cluster Analysis

K-means Clustering Algorithm

It is a recursive greedy algorithm to minimize *SSE*.

Initialize Start with a randomly generated centroids $\{\mu_k, k = 1, 2, \ldots, K\}$

Iterative

- Assign X_i to cluster C_{k^*} for which

$$k^* = argmin_{1 \leq k \leq K} ||X_i - \mu_k||^2$$

- After assigning all of the $X_i's$, update the centroids $\{\mu_k, k = 1, 2, , K\}$ (averages)
- continue the iterations till $max_k ||\mu_k^{new} - \mu_k^{old}|| < TOL$

Repeat the whole algorithm several times for randomly generated initial centroids (Monte Carlo alg.) and find the smallest

$$SSE(\mathcal{C}) = \sum_{k=1}^{K} \sum_{X_i \in C_k} ||X_i - \mu_k||^2 = \sum_{k=1}^{K} \sum_{i=1}^{n} I(X_i \in C_k) ||X_i - \mu_k||^2$$

Cluster Analysis

K-means Clustering Algorithm

It is a recursive greedy algorithm to minimize SSE.

Initialize Start with a randomly generated centroids $\{\mu_k, k = 1, 2, \ldots, K\}$

Iterative

- Assign X_i to cluster C_{k^*} for which

$$k^* = argmin_{1 \leq k \leq K} ||X_i - \mu_k||^2$$

- After assigning all of the $X_i's$, update the centroids $\{\mu_k, k = 1, 2, , K\}$ (averages)
- continue the iterations till $max_k ||\mu_k^{new} - \mu_k^{old}|| < TOL$

Repeat the whole algorithm several times for randomly generated initial centroids (Monte Carlo alg.) and find the smallest

$$SSE(\mathcal{C}) = \sum_{k=1}^{K} \sum_{X_i \in C_k} ||X_i - \mu_k||^2 = \sum_{k=1}^{K} \sum_{i=1}^{n} I(X_i \in C_k) ||X_i - \mu_k||^2$$

Cluster Analysis

K-means Clustering Algorithm

It is a recursive greedy algorithm to minimize SSE.

Initialize Start with a randomly generated centroids $\{\mu_k, k = 1, 2, \ldots, K\}$

Iterative

- Assign X_i to cluster C_{k^*} for which

$$k^* = argmin_{1 \leq k \leq K} ||X_i - \mu_k||^2$$

- After assigning all of the $X_i's$, update the centroids $\{\mu_k, k = 1, 2, , K\}$ (averages)
- continue the iterations till $max_k ||\mu_k^{new} - \mu_k^{old}|| < TOL$

Repeat the whole algorithm several times for randomly generated initial centroids (Monte Carlo alg.) and find the smallest

$$SSE(\mathcal{C}) = \sum_{k=1}^{K} \sum_{X_i \in C_k} ||X_i - \mu_k||^2 = \sum_{k=1}^{K} \sum_{i=1}^{n} I(X_i \in C_k) ||X_i - \mu_k||^2$$

Cluster Analysis

K-means Clustering Algorithm

It is a recursive greedy algorithm to minimize *SSE*.

Initialize Start with a randomly generated centroids $\{\mu_k, k = 1, 2, \ldots, K\}$

Iterative

- Assign X_i to cluster C_{k^*} for which

$$k^* = argmin_{1 \leq k \leq K} ||X_i - \mu_k||^2$$

- After assigning all of the $X_i's$, update the centroids $\{\mu_k, k = 1, 2, , K\}$ (averages)
- continue the iterations till $max_k ||\mu_k^{new} - \mu_k^{old}|| < TOL$

Repeat the whole algorithm several times for randomly generated initial centroids (Monte Carlo alg.) and find the smallest

$$SSE(\mathcal{C}) = \sum_{k=1}^{K} \sum_{X_i \in C_k} ||X_i - \mu_k||^2 = \sum_{k=1}^{K} \sum_{i=1}^{n} I(X_i \in C_k) ||X_i - \mu_k||^2$$

Cluster Analysis

K-means Clustering Algorithm

It is a recursive greedy algorithm to minimize *SSE*.

Initialize Start with a randomly generated centroids $\{\mu_k, k = 1, 2, \ldots, K\}$

Iterative

- Assign X_i to cluster C_{k^*} for which

$$k^* = argmin_{1 \leq k \leq K} ||X_i - \mu_k||^2$$

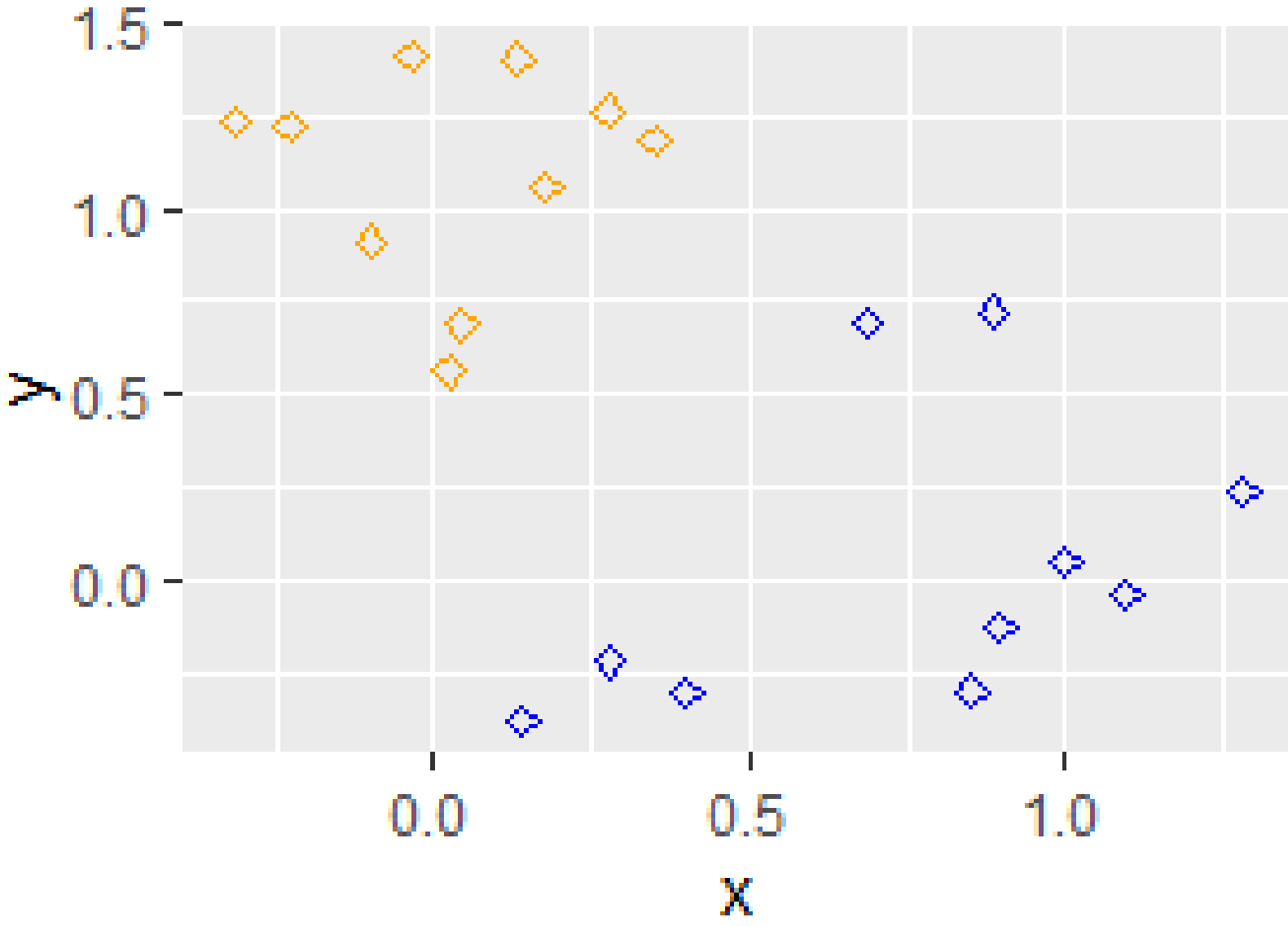
- After assigning all of the $X_i's$, update the centroids $\{\mu_k, k = 1, 2, , K\}$ (averages)
- continue the iterations till $max_k ||\mu_k^{new} - \mu_k^{old}|| < TOL$

Repeat the whole algorithm several times for randomly generated initial centroids (Monte Carlo alg.) and find the smallest

$$SSE(\mathcal{C}) = \sum_{k=1}^{K} \sum_{X_i \in C_k} ||X_i - \mu_k||^2 = \sum_{k=1}^{K} \sum_{i=1}^{n} I(X_i \in C_k) ||X_i - \mu_k||^2$$

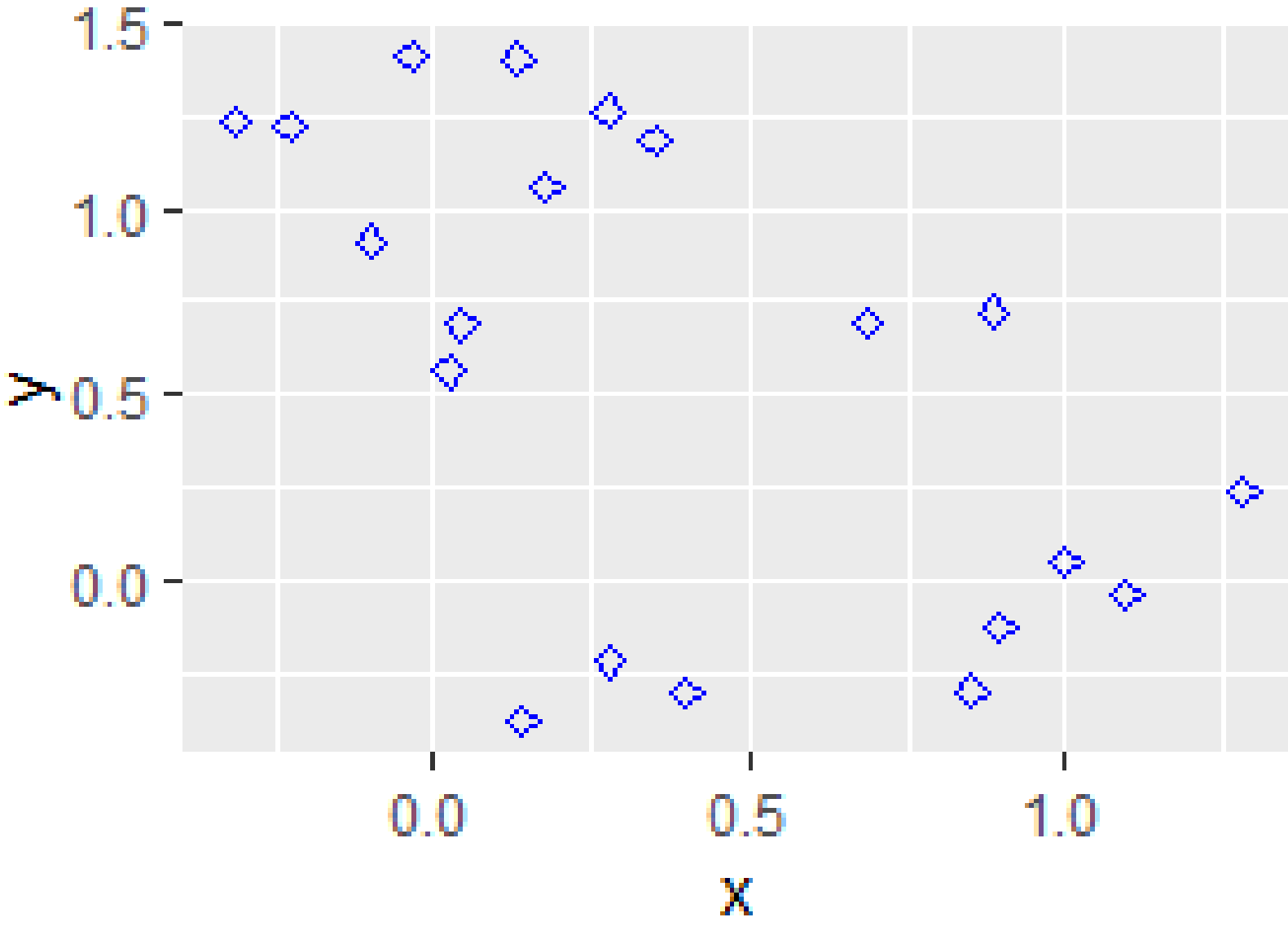
Cluster Analysis

Simulation Example: See code cluster.R



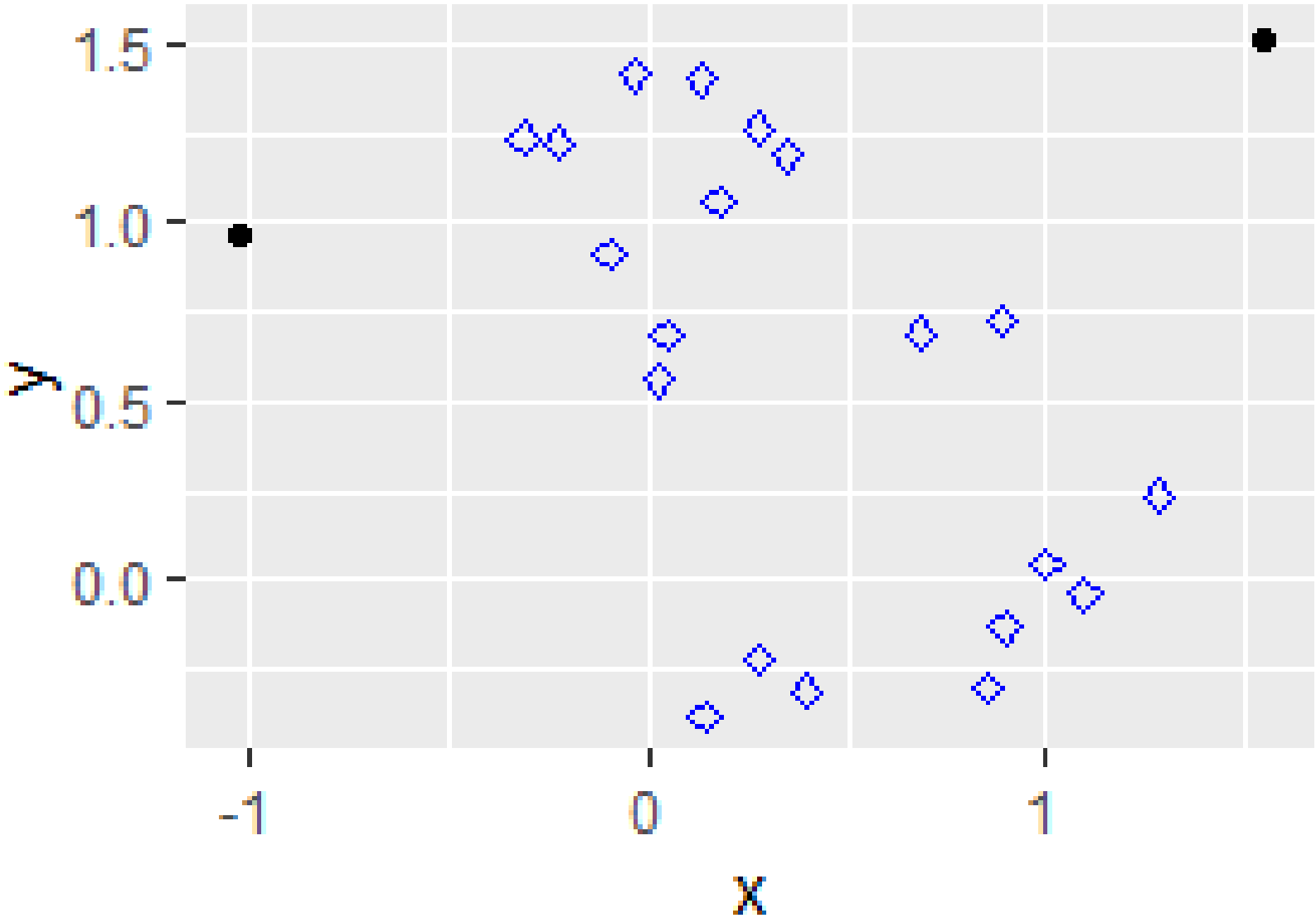
Cluster Analysis

Simulation Example: See code cluster.R



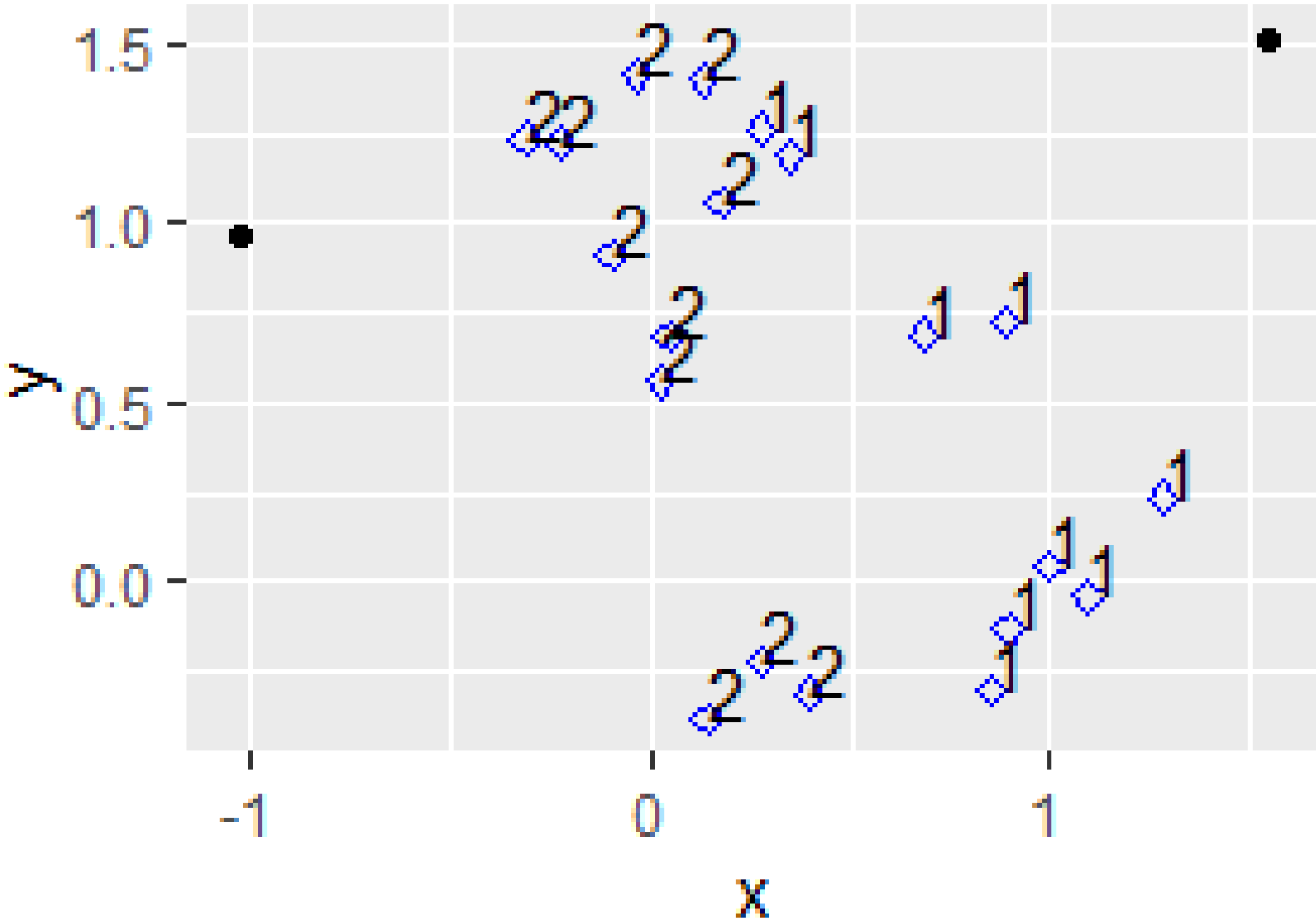
Cluster Analysis

Simulation Example: See code cluster.R



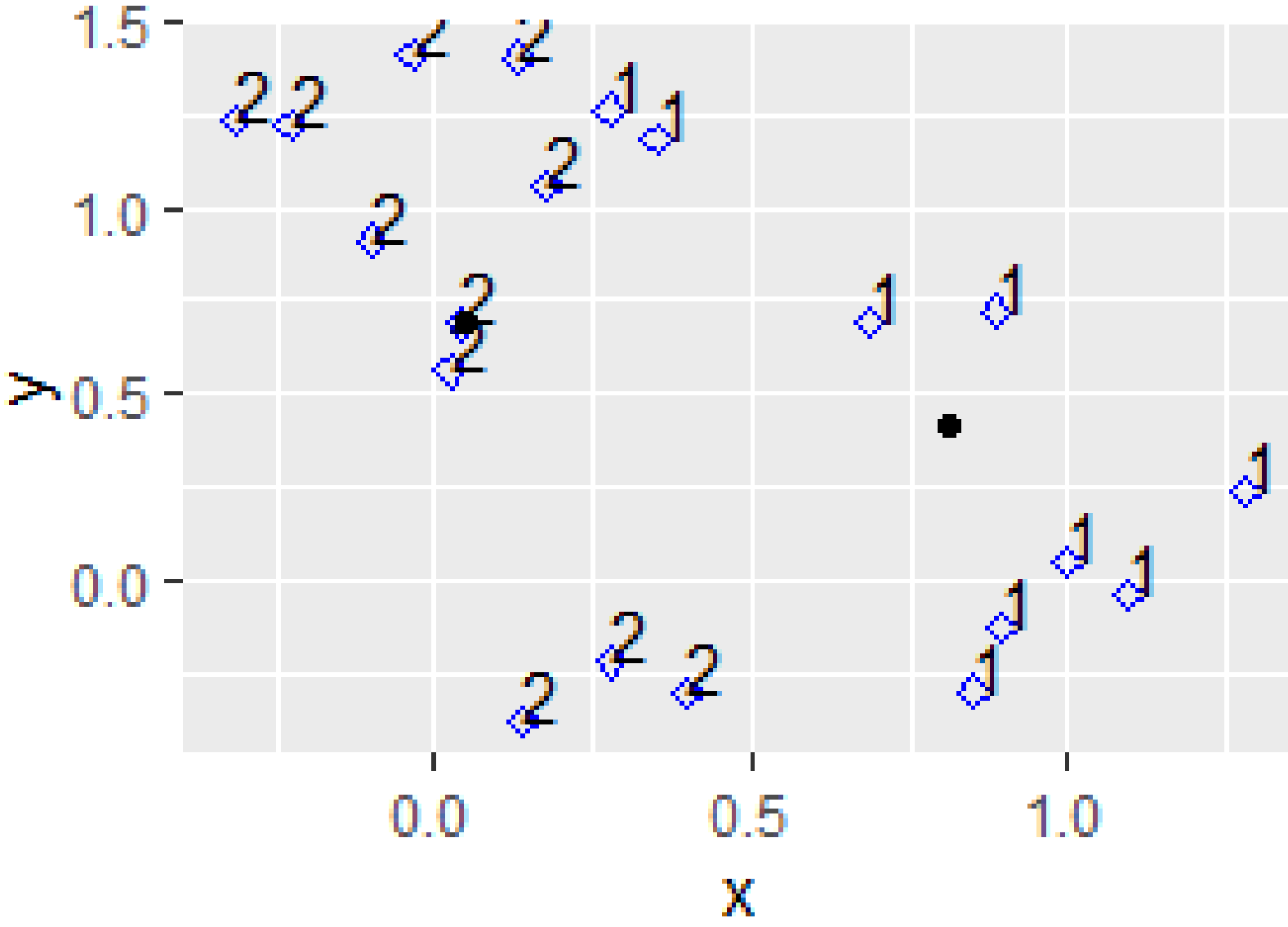
Cluster Analysis

Simulation Example: See code cluster.R



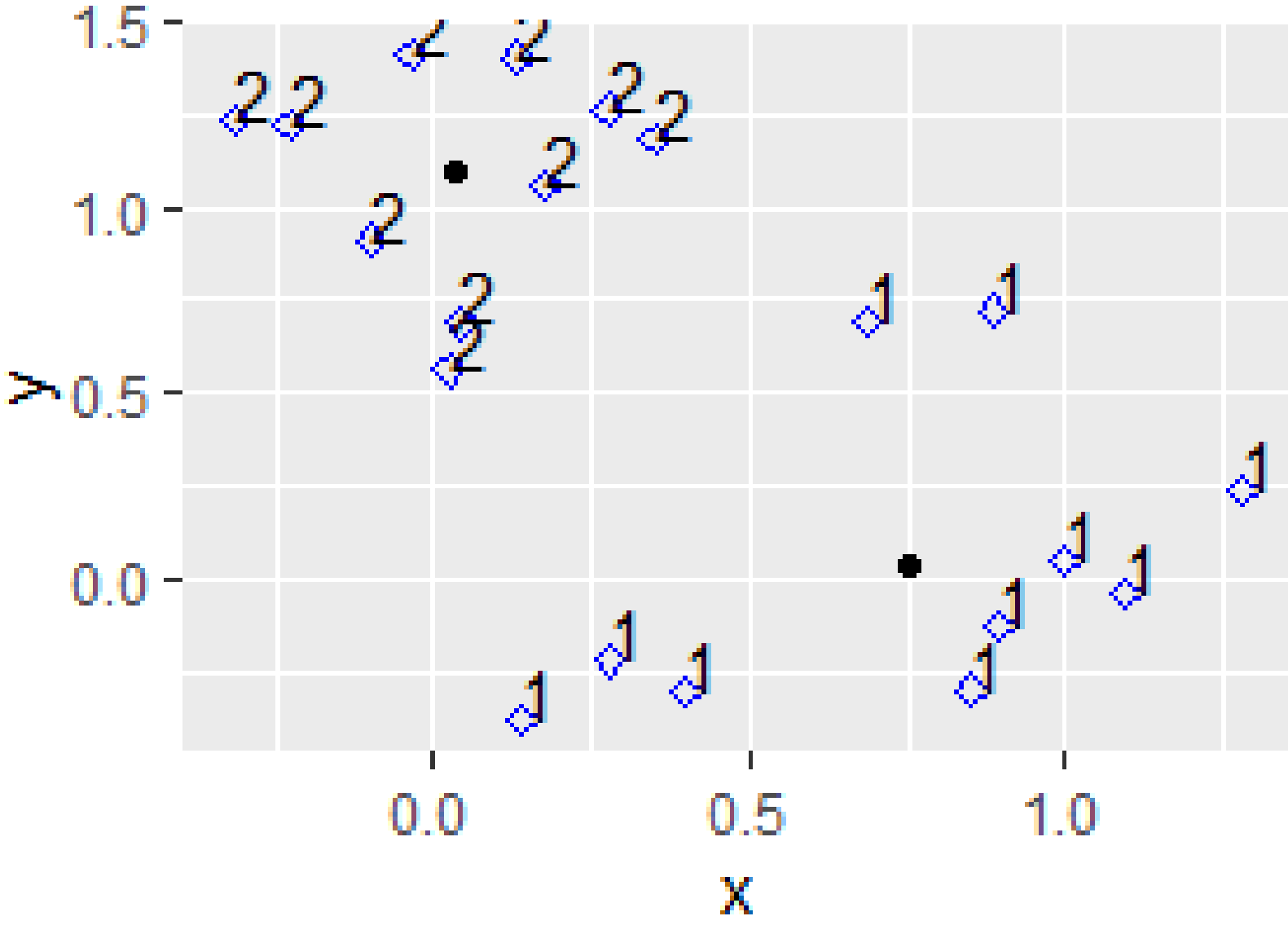
Cluster Analysis

Simulation Example: See code cluster.R



Cluster Analysis

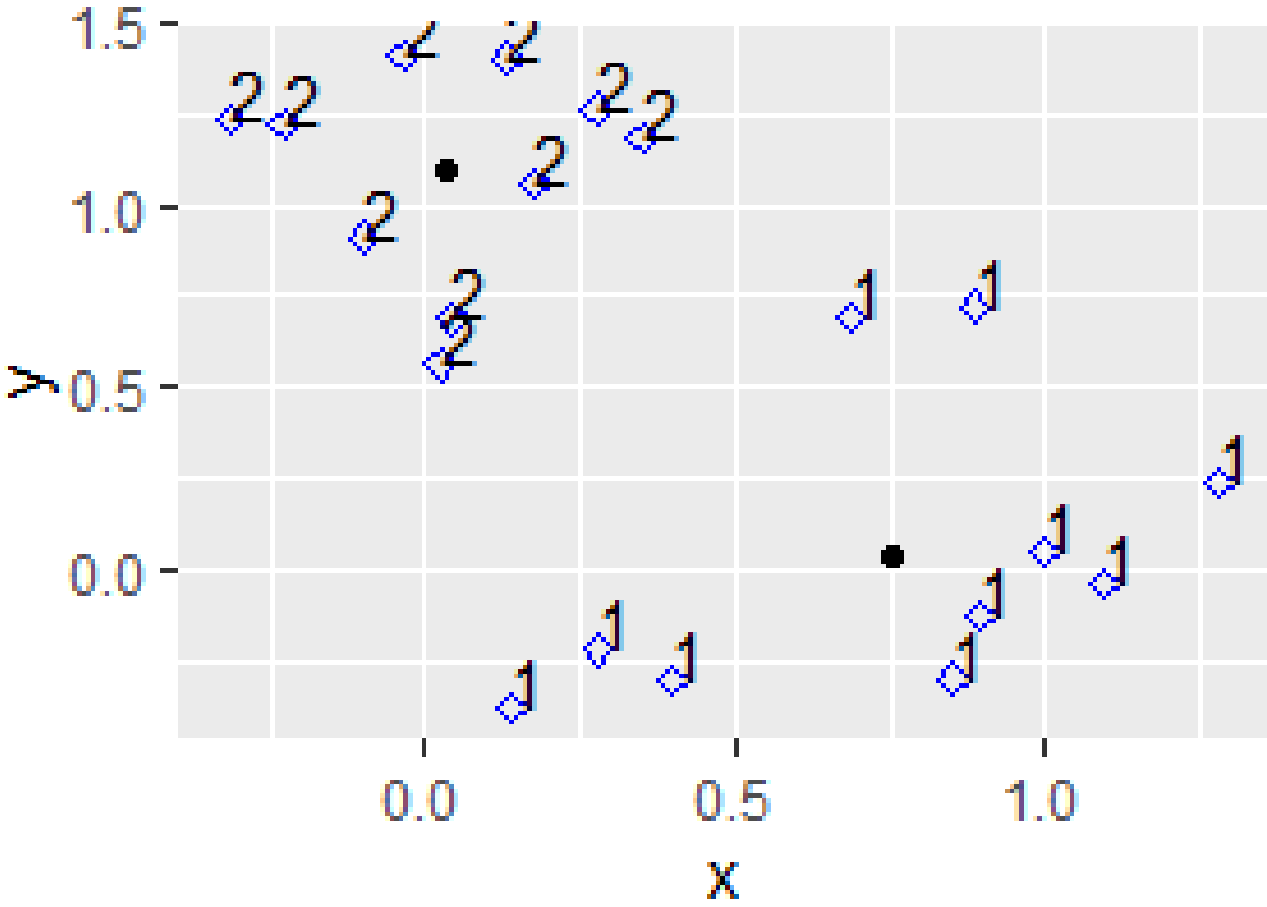
Simulation Example: See code cluster.R



Cluster Analysis

Example: See code cluster.R

kmeans(dataframe, centers, iter.max = 10, nstart = 1, algorithm = c("Hartigan-Wong", "Lloyd", "Forgy", "MacQueen"), trace=FALSE)



Kernel K-means Clustering Algorithm

Cluster Analysis

Kernels K-means Clustering Algorithm

Using kernels like $K(x, y) = h(x)^T h(y)$ so that

$$SSE(\mathcal{C}) = \sum_{k=1}^{K} \sum_{X_i \in C_k} ||h(X_i) - \mu_k^h||^2$$

where

$$\mu_k^h = \frac{1}{|C_k|} \sum_{X_j \in C_k} h(X_j)$$

Cluster Analysis

Kernels K-means Clustering Algorithm

and so

$$h(X_i) - \mu_k^h = h(X_i) - \frac{1}{|C_k|} \sum_{X_j \in C_k} h(X_j)$$

$$= \frac{1}{|C_k|} \sum_{X_j \in C_k} (h(X_i) - h(X_j))$$

and

$$||h(X_i) - \mu_k^h||^2 = (h(X_i) - \mu_k^h)^T (h(X_i) - \mu_k^h)$$

$$= \frac{1}{|C_k|^2} \sum_{X_j \in C_k} \sum_{X_\ell \in C_k} (h(X_i) - h(X_j))^T (h(X_i) - h(X_\ell))$$

$$= \frac{1}{|C_k|^2} \sum_{X_j \in C_k} \sum_{X_\ell \in C_k} (K(X_i, X_i) - 2K(X_i, X_j) + K(X_j, X_\ell))$$

DIY in R

1. Carry out K-means Cluster analysis using *kmeans* on the gene data. Use different maximum number of iterations *iter.max* and different number of initial points *nstart*.
2. Carry out kernel K-means Cluster analysis using *kkmeans* in the library *kernlab* on the gene data.

Please study the different methods in the ISL book.

DIY in R

1. Carry out K-means Cluster analysis using *kmeans* on the gene data. Use different maximum number of iterations *iter.max* and different number of initial points *nstart*.
2. Carry out kernel K-means Cluster analysis using *kkmeans* in the library *kernlab* on the gene data.

Please study the different methods in the ISL book.

DIY in R

1. Carry out K-means Cluster analysis using *kmeans* on the gene data. Use different maximum number of iterations *iter.max* and different number of initial points *nstart*.
2. Carry out kernel K-means Cluster analysis using *kkmeans* in the library *kernlab* on the gene data.

Please study the different methods in the ISL book.

DIY in R

1. Carry out K-means Cluster analysis using *kmeans* on the gene data. Use different maximum number of iterations *iter.max* and different number of initial points *nstart*.
2. Carry out kernel K-means Cluster analysis using *kkmeans* in the library *kernlab* on the gene data.

Please study the different methods in the ISL book.

Hierarchical Clustering

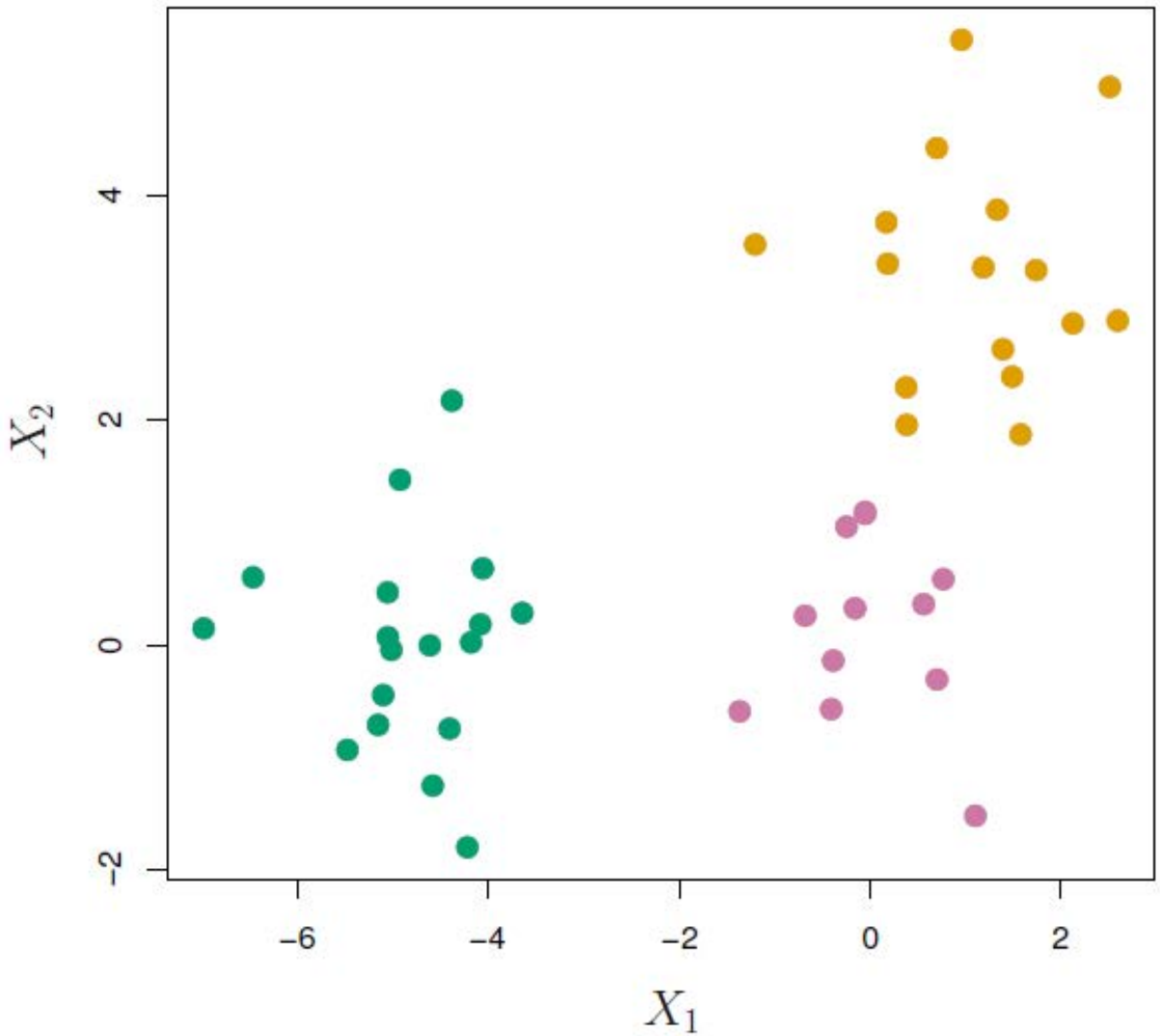
Hierarchical Clustering

- At the top, there is one cluster that contains all of the data points which sequentially divides up till it ends with n clusters that contain one observation at the bottom.
- Two approaches: divisive (top-bottom) and agglomerative (bottom-up)
- Results are represented Using a dendrogram

Hierarchical Clustering

Agglomerative (bottom-up)

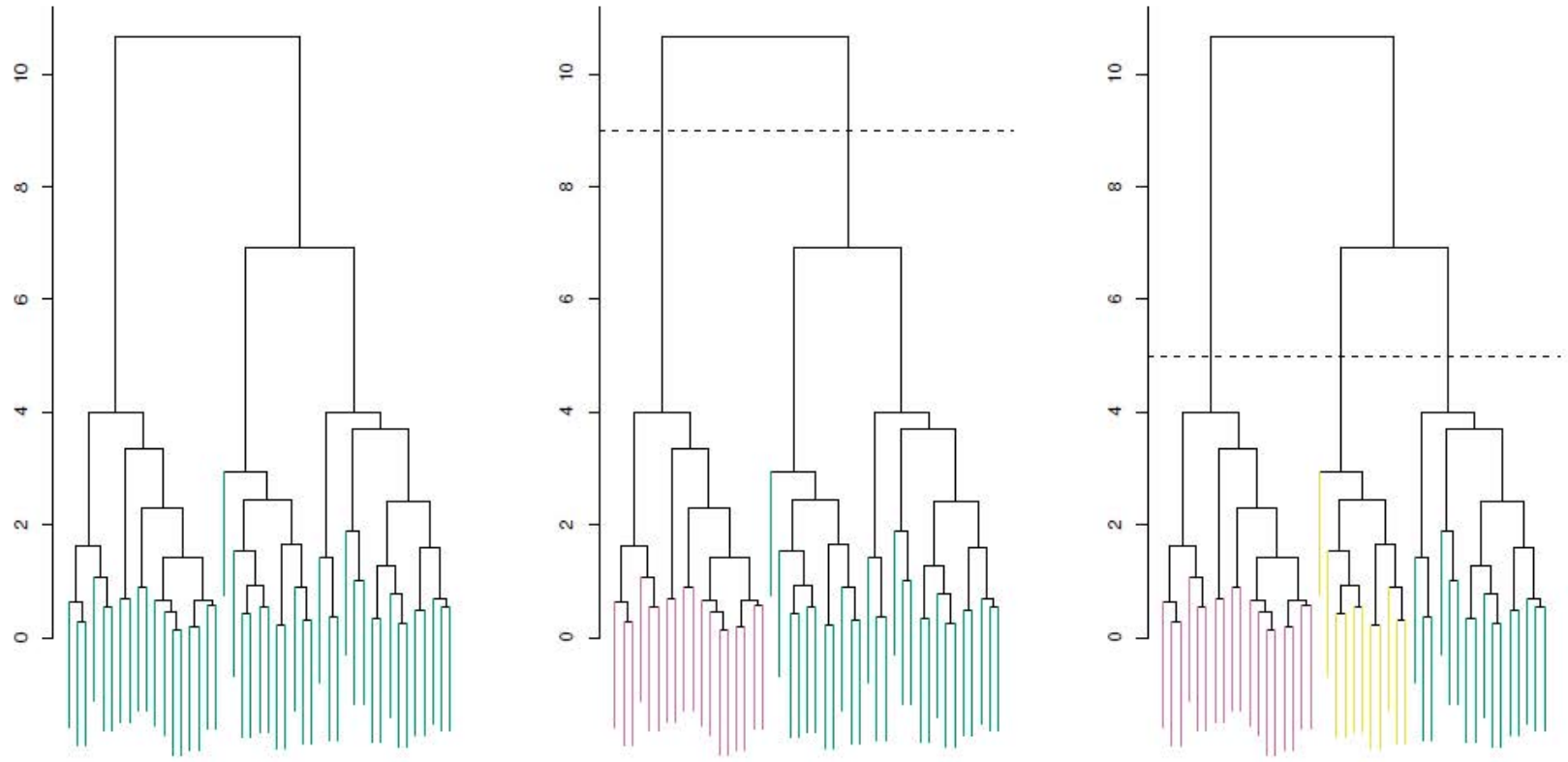
Example



Hierarchical Clustering

Agglomerative (bottom-up)

Example



Hierarchical Clustering

Agglomerative (bottom-up)

- Initially start with n clusters with one of the observations in each one of them.
- At each step, merge the closest two clusters to reduce the number of clusters by one.
- Closeness is measured using a dissimilarity function.

Hierarchical Clustering

Divisive (top-bottom)

- Initially start with 1 cluster with the n observations in it.
- At each step, divide the farthest two clusters to increase the number of clusters by one.
- Distances are measured using a dissimilarity function.

Hierarchical Clustering

Agglomerative (bottom-up)

Example

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 $i = n, n-1, \ldots, 2$:

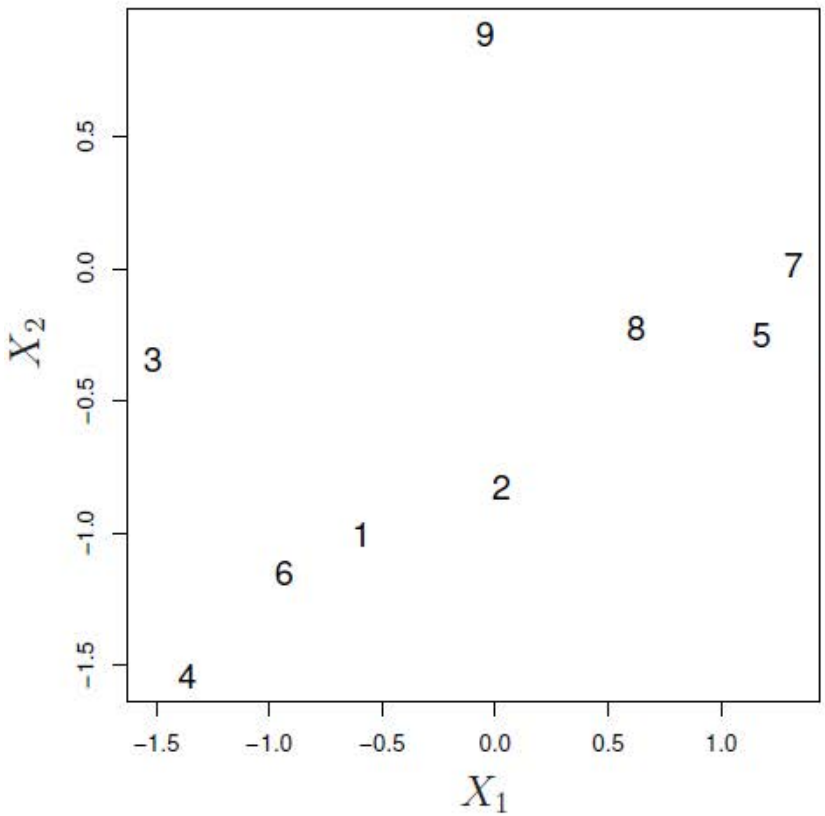
   (a) Examine all pairwise inter-cluster dissimilarities among the i clusters and identify the pair of clusters that are least dissimilar (that is, most similar). Fuse these two clusters. The dissimilarity between these two clusters indicates the height in the dendrogram at which the fusion should be placed.

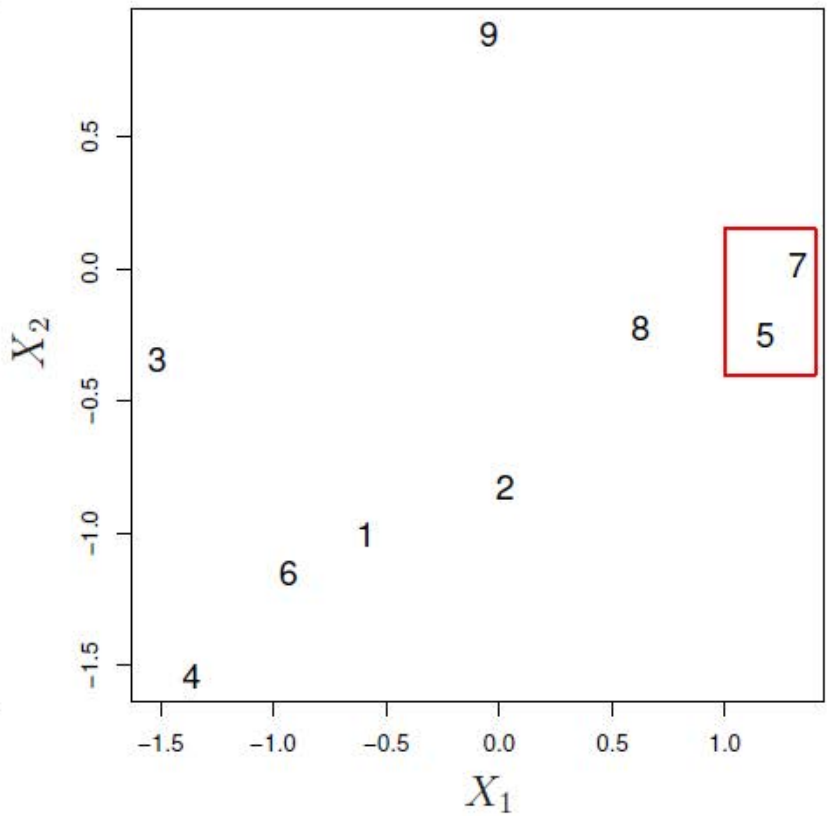
   (b) Compute the new pairwise inter-cluster dissimilarities among the $i-1$ remaining clusters.

Hierarchical Clustering

Agglomerative (bottom-up)

Example

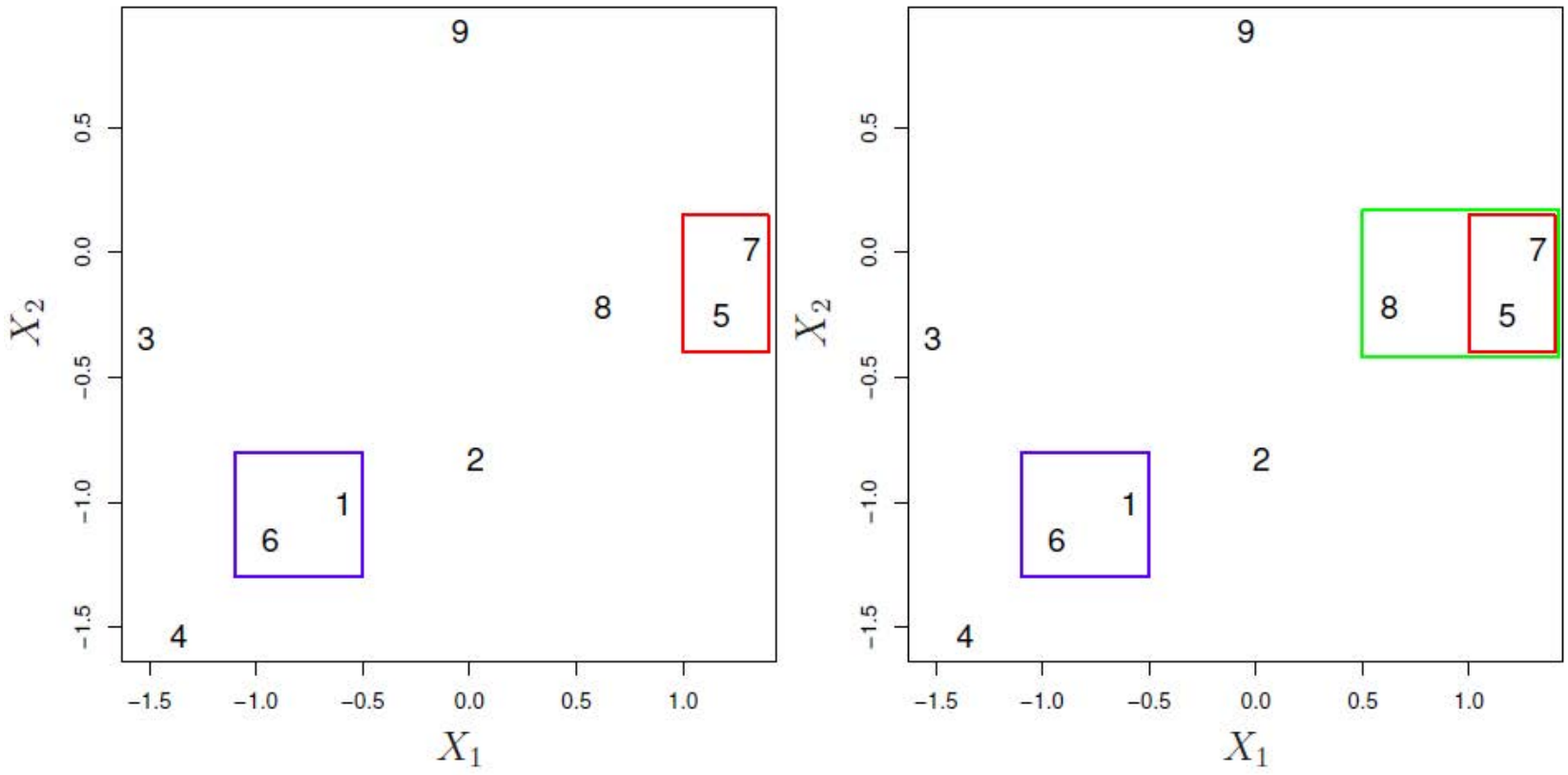




Hierarchical Clustering

Agglomerative (bottom-up)

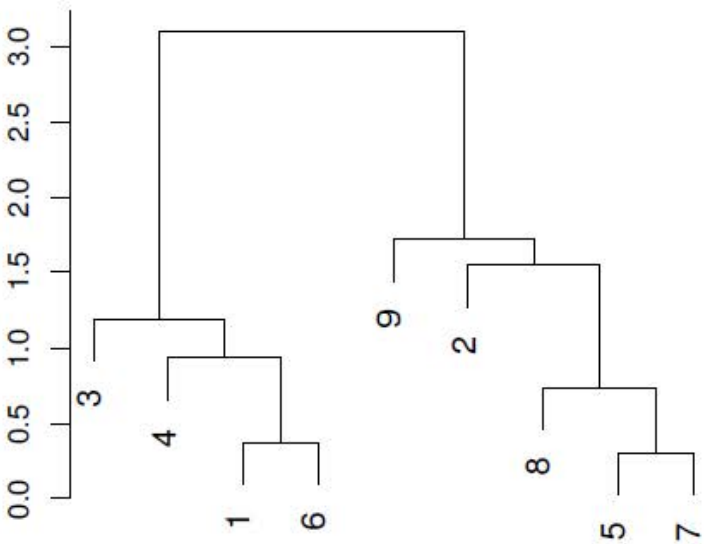
Example

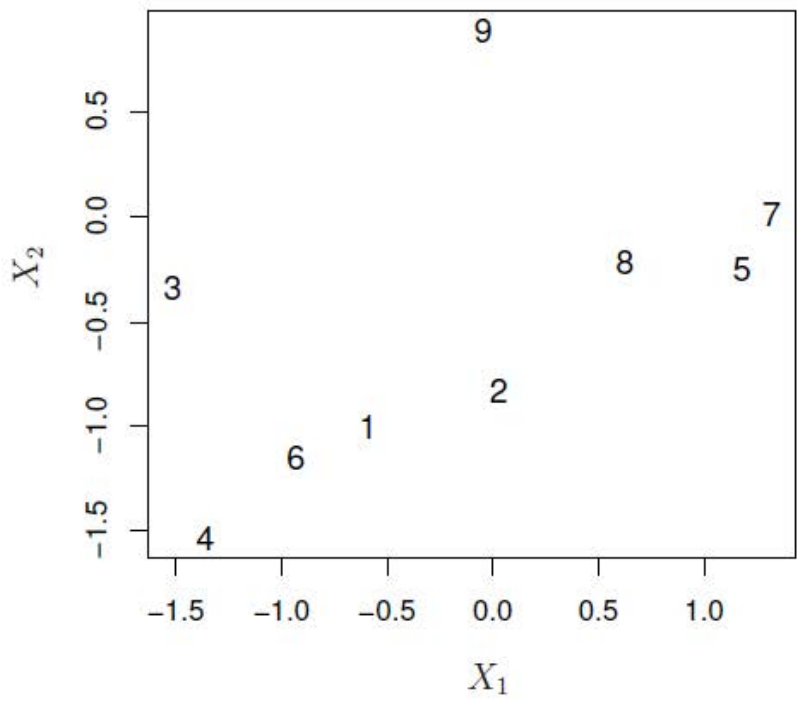


Hierarchical Clustering

Agglomerative (bottom-up)

Example





Hierarchical Clustering

Types of dissimilarities (linkage) between clusters

- Single linkage:

$$d_{single}(C_1, C_2) = \min_{i \in C_1, j \in C_2} d(x_i, x_j)$$

But, resulting clusters could be highly spread out points (non-compact).

- Complete linkage:

$$d_{complete}(C_1, C_2) = \max_{i \in C_1, j \in C_2} d(x_i, x_j)$$

But, resulting clusters could be compact but not enough far apart.

- Average linkage:

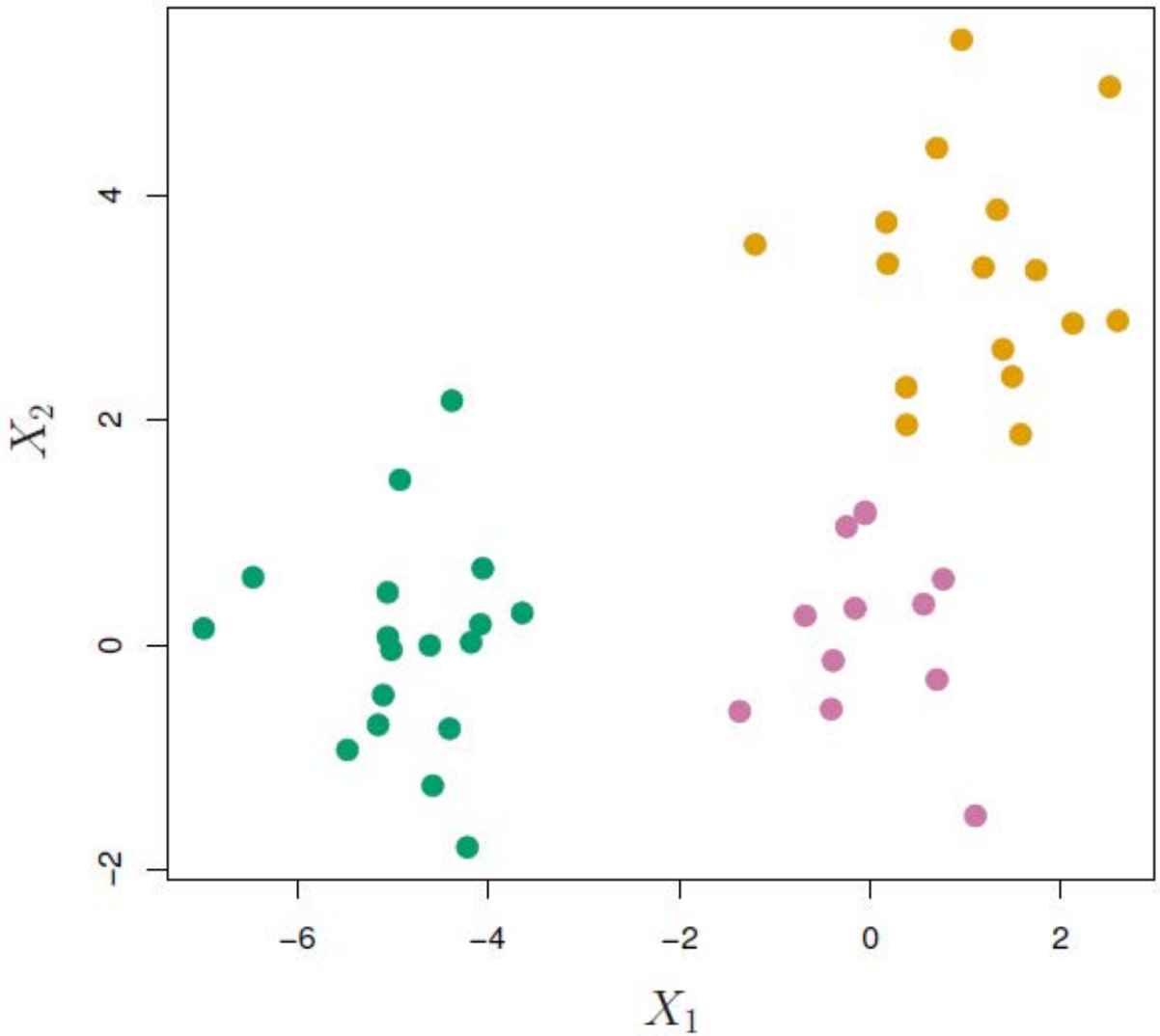
$$d_{single}(C_1, C_2) = \frac{1}{|C_1||C_2|} \sum_{i \in C_1, j \in C_2} d(x_i, x_j)$$

It strikes a balance between both.

Hierarchical Clustering

Agglomerative (bottom-up)

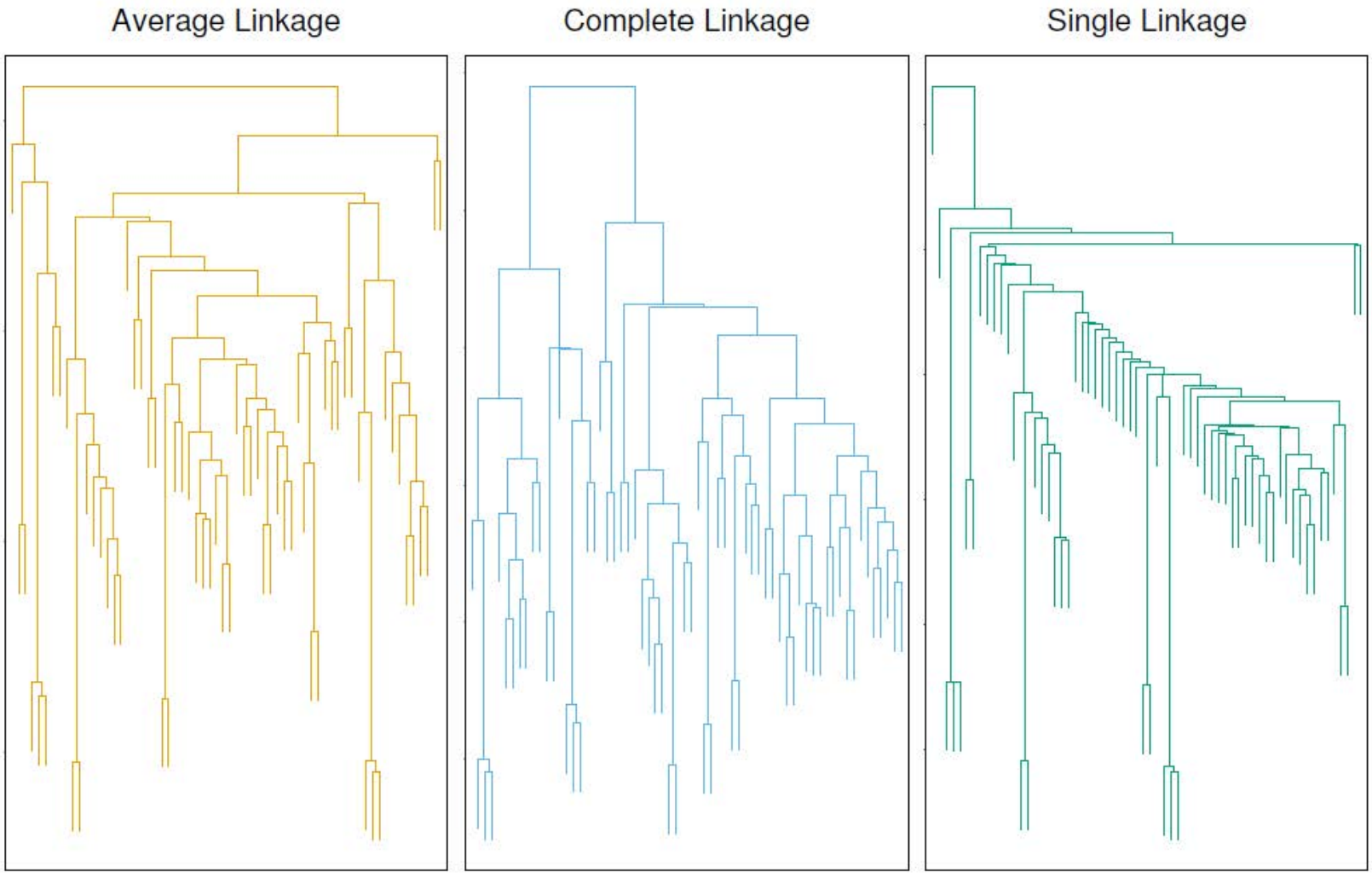
Example



Hierarchical Clustering

Agglomerative (bottom-up)

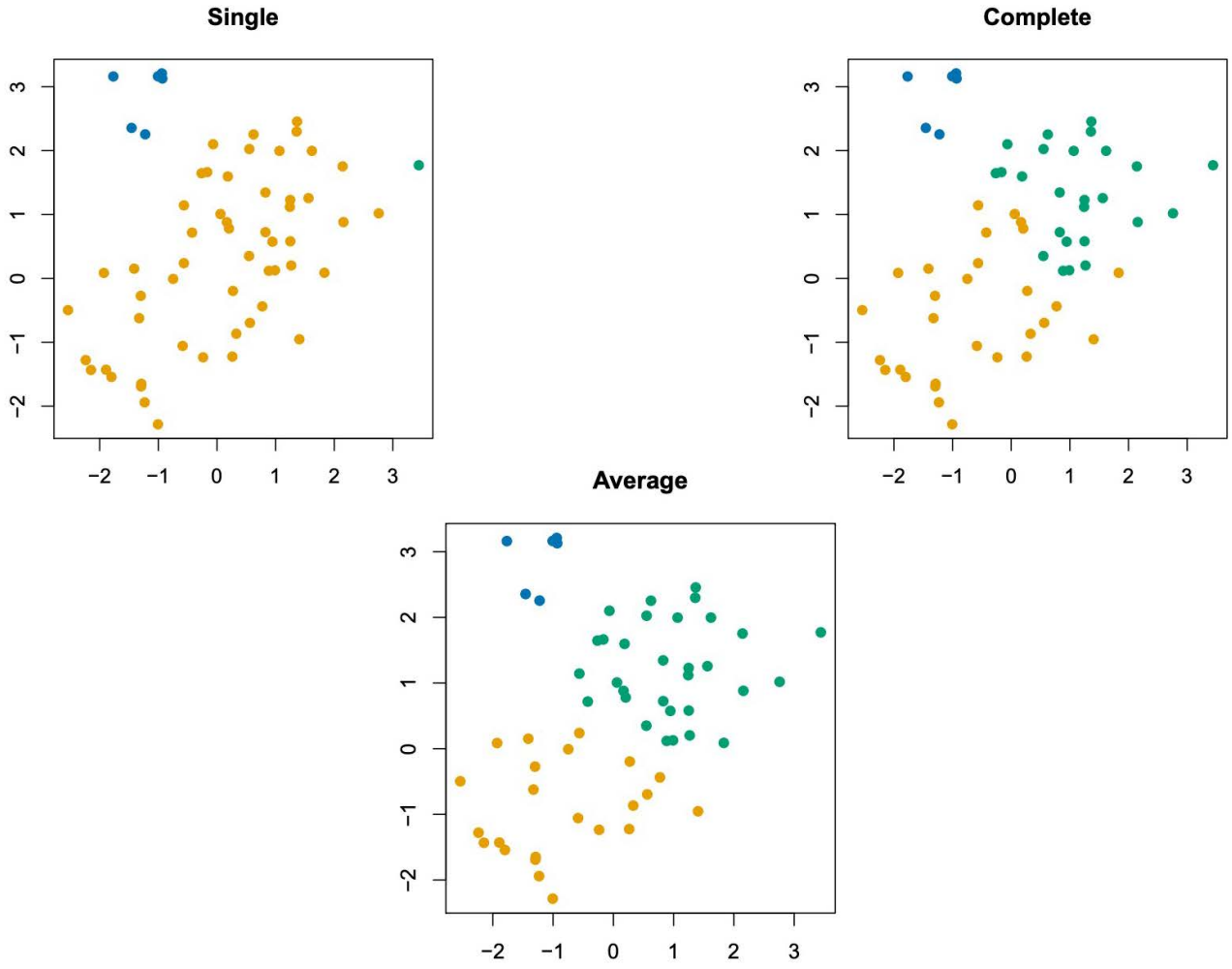
Example



Hierarchical Clustering

Agglomerative (bottom-up)

Example



Principal Component Analysis

Principal Component Analysis

- Goal: To find the best q linear approximations to a set of data $x_i \in \mathbb{R}^p$ for $i = 1, 2, \ldots, N$, where $q \leq p$.
- Assume that x_i is centered; that is, $\bar{x}_i = 0$. (Actually, it must be standardized.)
- The presentation of

$$x_i = f(\lambda) = \mu + V_q \lambda$$

where V_q is a $p \times q$ orthogonal matrix ($V_q^T V_q = I_q$), λ is $q \times 1$ parameter vector, and μ is $p \times 1$ parameter vector, could be found by minimizing the recostruction error

$$\min_{\mu, \{\lambda_i\}, V} \sum_{i=1}^{N} ||x_i - \mu - V_q \lambda_i||^2$$

Principal Component Analysis

- Assuming V_q is given, then $\hat{\mu} = \bar{x} = 0$ and $\hat{\lambda}_i = V_q^T(x_i - \bar{x}) = V_q^T x_i$, and the the problem becomes
- $$\min_{V_q} \sum_{i=1}^{N} ||x_i - H_q x_i||^2$$

  where $H_q = V_q V_q^T$ is a projection matrix.
- where the rank-q reconstruction $H_q x_i$ is the orthogonal projection of $x_i \in \mathbb{R}^p$ into the space spanned by the columns of V_q

Principal Component Analysis

- Consider the $N \times p$ data matrix X
- Using SVD of $X = U_{N\times p}\, D_p\, V^T_{p\times p}$ with diagonal matrix D_p with elements (singular values)
  $d_1 \geq d_2 \geq d_3 \geq \cdots \geq d_p \geq 0$, and
- $U^T U = I_p$ where column u_j is called left singular vector
- $V^T V = I_p$ where column v_j is called right singular (loadings) vector
- V_q is the first q columns of V
- principal components of X are the columns of UD

Principal Component Analysis

- The N optimal $\hat{\lambda}_i = V_q^T x_i$ are the rows of UD which are the first q principal components.
- Xv_1 has the highest variance among all linear combinations of the features and Xv_2 has the highest variance among all linear combinations Xv of the features such that v is orthogonal to v_1.
- Elements of Xv_j are called the scores of the j^{th} principal component.

Principal Component Analysis

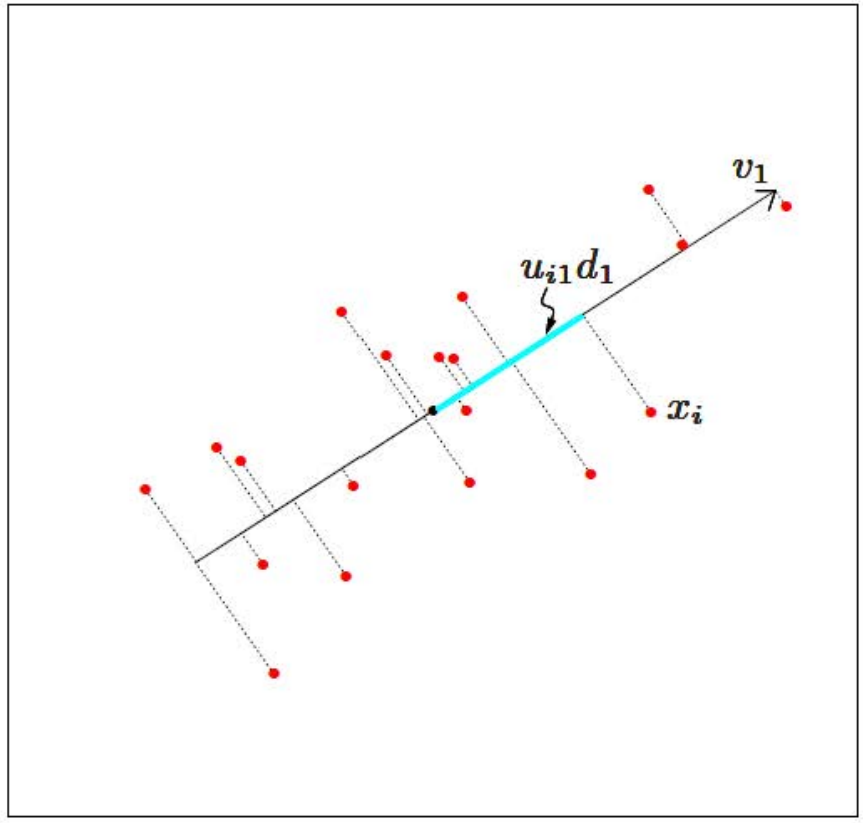


FIGURE 14.20. *The first linear principal component of a set of data. The line minimizes the total squared distance from each point to its orthogonal projection onto the line.*

Principal Component Analysis

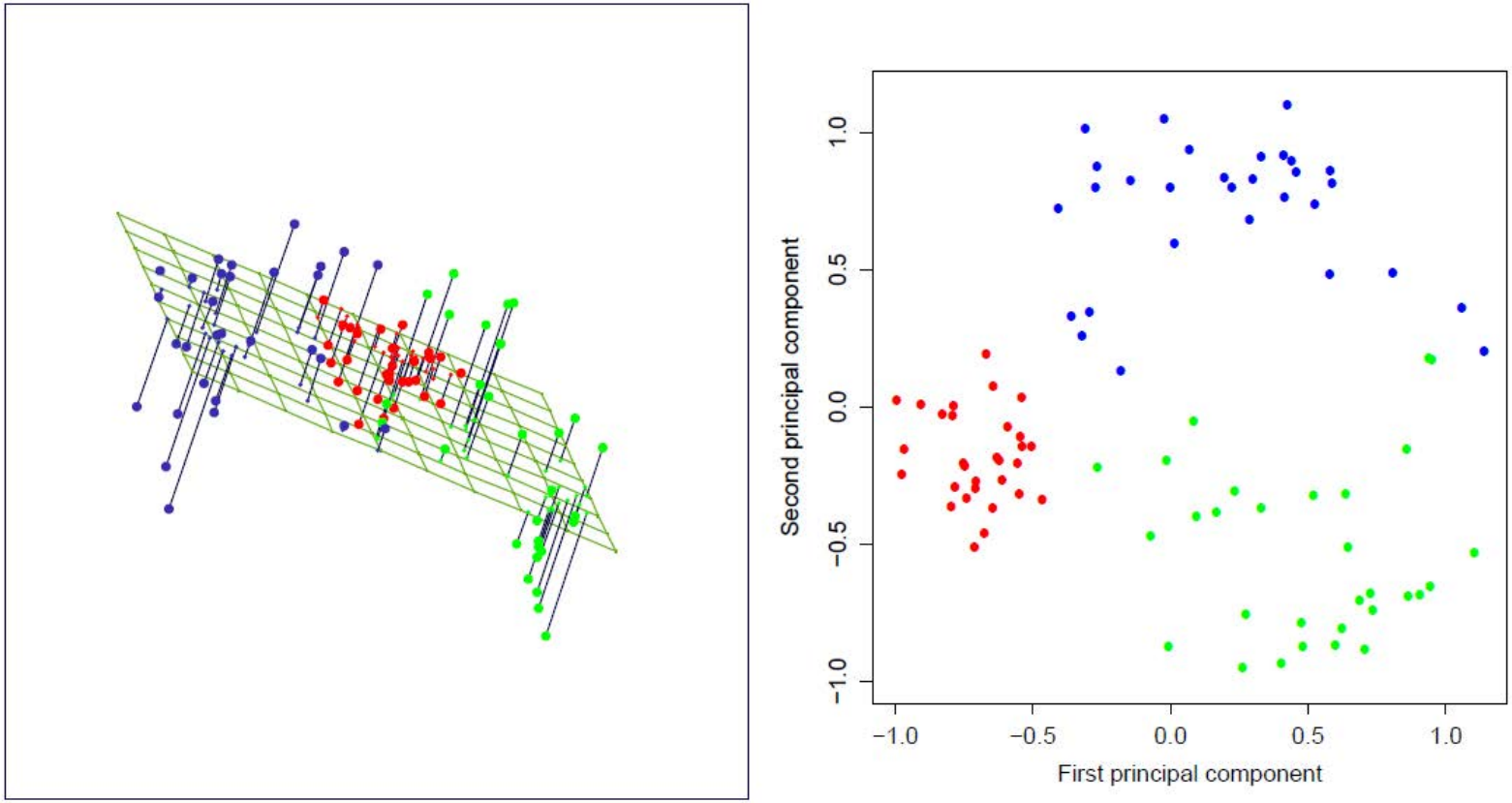
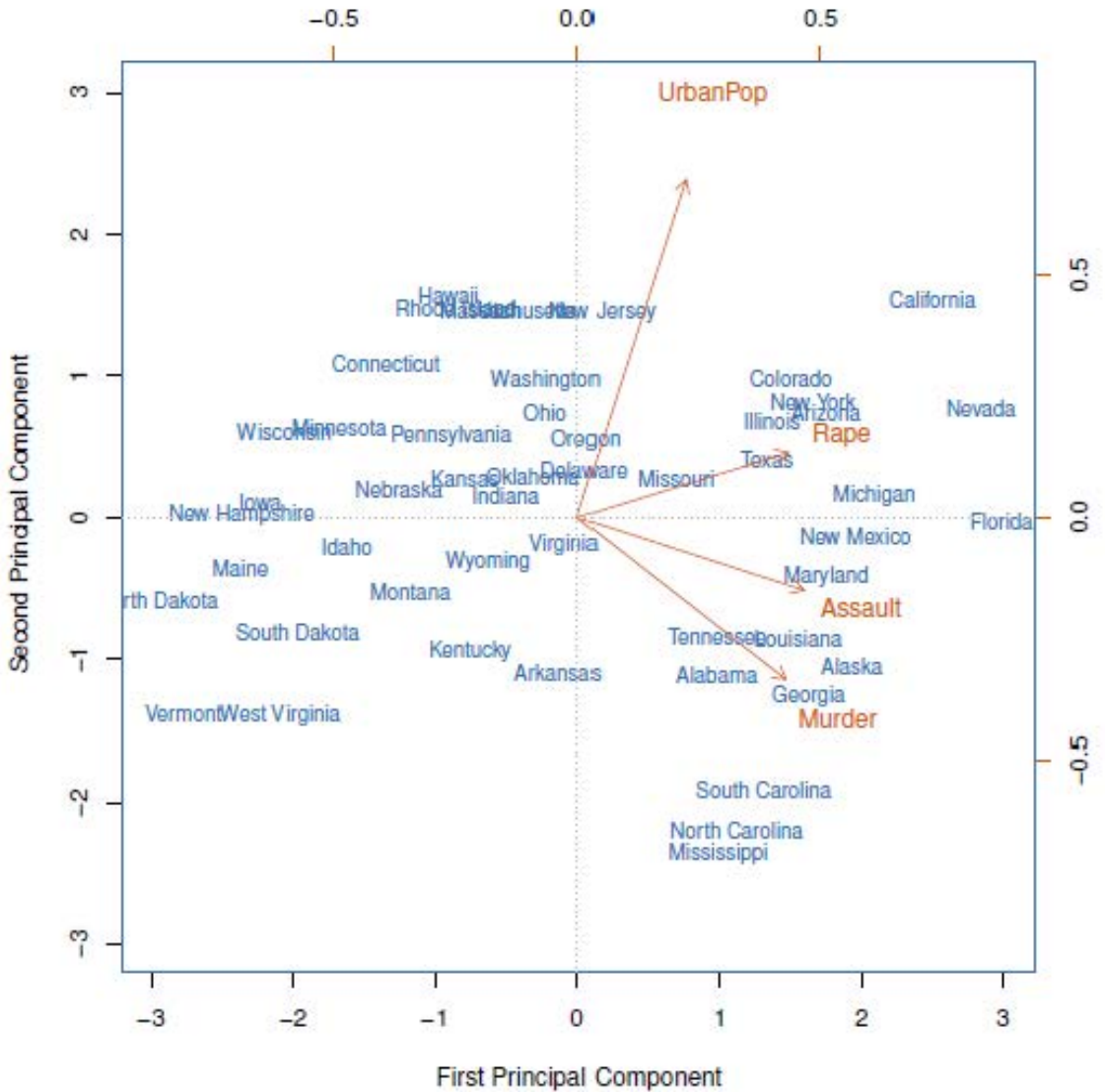


FIGURE 14.21. *The best rank-two linear approximation to the half-sphere data. The right panel shows the projected points with coordinates given by* $\mathbf{U}_2\mathbf{D}_2$*, the first two principal components of the data.*

Principal Component Analysis

Example: Arrest Data

$N = 50$ states with $p = 4$ features: Assault, Murder, and Rape as well as UrbanPop. First standardize them.



Principal Component Analysis

Example: Arrest Data

$N = 50$ states with $p = 4$ features: Assault, Murder, and Rape as well as UrbanPop. First standardize them.

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

TABLE 12.1. *The principal component loading vectors, ϕ_1 and ϕ_2, for the* USArrests *data. These are also displayed in Figure 12.1.*

Principal Component Analysis

Example: Arrest Data

$N = 50$ states with $p = 4$ features: Assault, Murder, and Rape as well as UrbanPop. First standardize them.

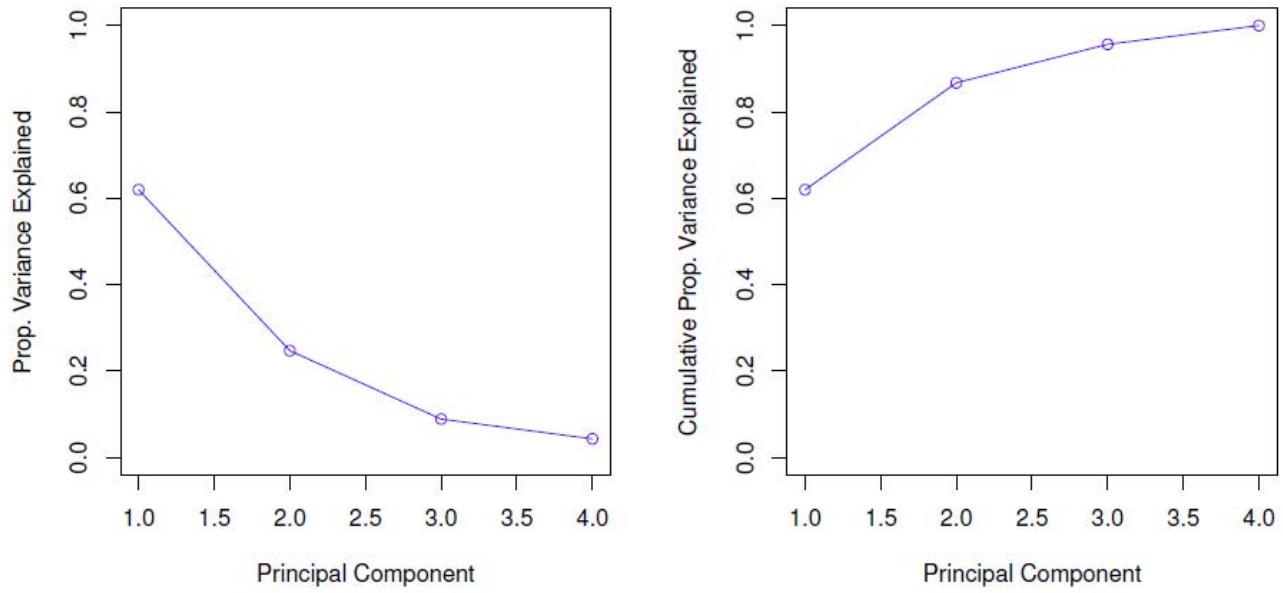


FIGURE 12.3. Left: *a scree plot depicting the proportion of variance explained by each of the four principal components in the* `USArrests` *data.* Right: *the cumulative proportion of variance explained by the four principal components in the* `USArrests` *data.*

Principal Component Analysis

Example: Arrest Data

$N = 50$ states with $p = 4$ features: Assault, Murder, and Rape as well as UrbanPop. First standardize them.

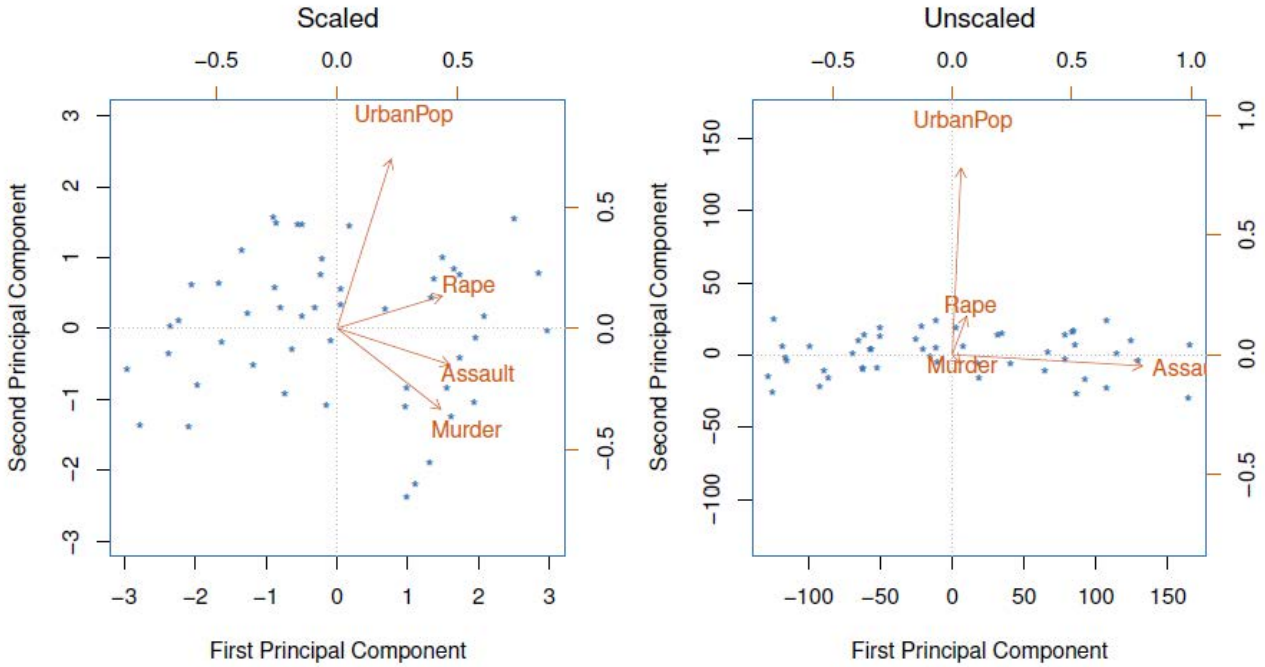


FIGURE 12.4. *Two principal component biplots for the* `USArrests` *data.* Left: *the same as Figure 12.1, with the variables scaled to have unit standard deviations.* Right: *principal components using unscaled data.* `Assault` *has by far the largest loading on the first principal component because it has the highest variance among the four variables. In general, scaling the variables to have standard deviation one is recommended.*

Principal Component Analysis

Example: Handwriting digits

The pictures digitized into 16x16 gray scale images and images of 658 handwritten 3's are the x_i's in $\mathbb{R}^{256}$. SVD is calculated for the 658x256 matrix X.
Twelve of the 256 possible principal components account for 63%
Fifty of the 256 possible principal components account for 90%

Principal Component Analysis

Example: Handwriting digits

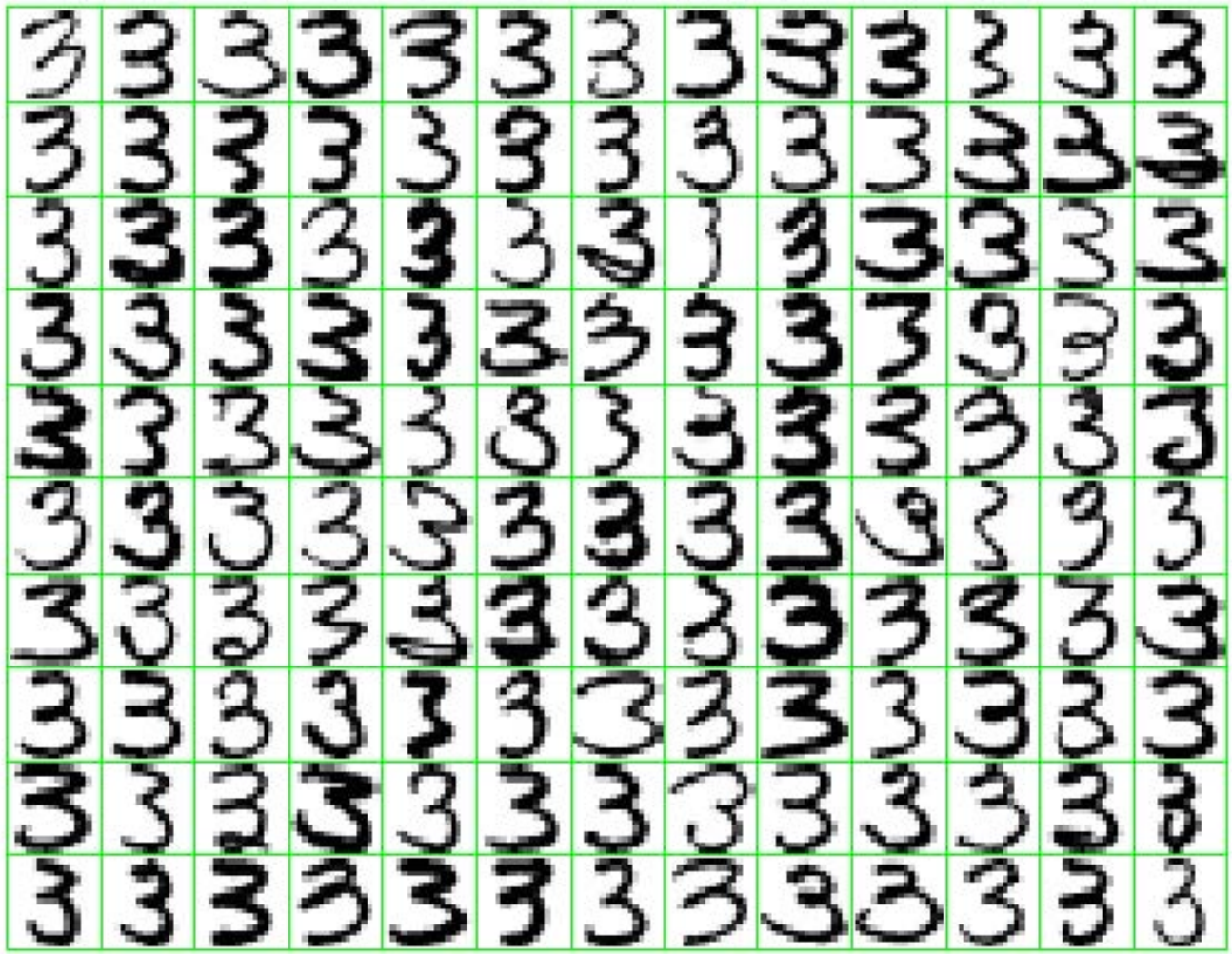


FIGURE 14.22. *A sample of* 130 *handwritten 3's shows a variety of writing styles.*

Principal Component Analysis

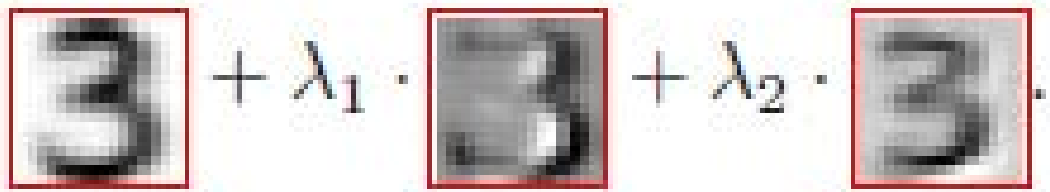
Example: Handwriting digits

The first component v_1 is due to the horizontal movement. It accounts for the lengthening of the lower tail of the 3,

The second component v_2 is due to the vertical movement. It accounts for the thickness of the 3

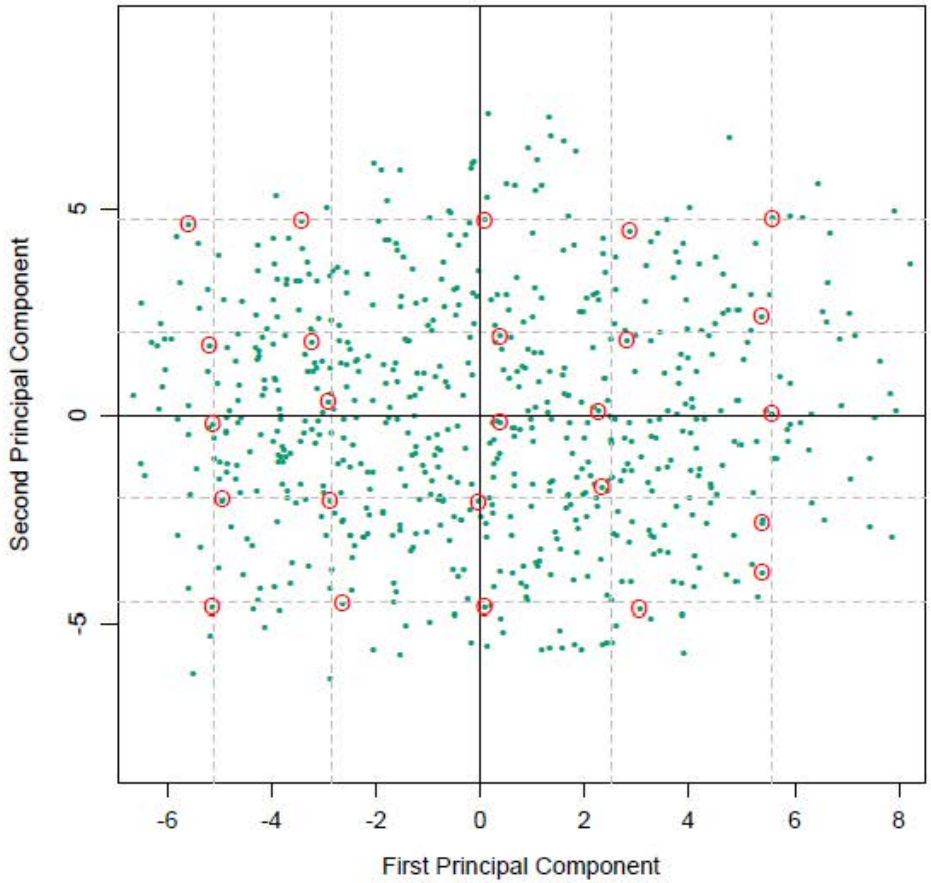
$$\hat{f}(\lambda) = \bar{x} + \lambda_1 v_1 + \lambda_2 v_2$$

$$= \bar{x} + \lambda_1 \cdot v_1 + \lambda_2 \cdot v_2.$$



Principal Component Analysis

Example: Handwriting digits



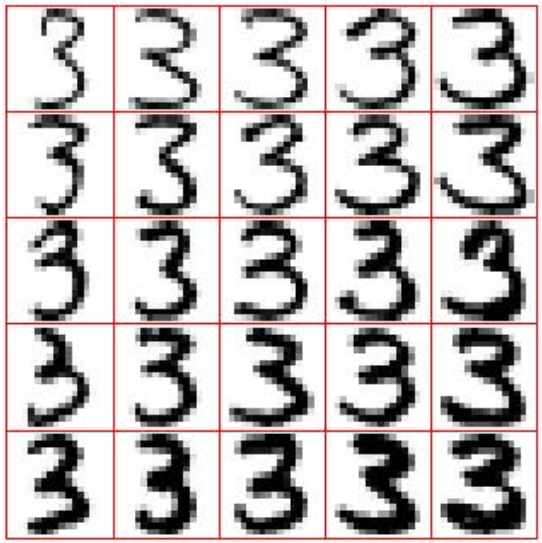


FIGURE 14.23. *(Left panel:) the first two principal components of the hand-written threes. The circled points are the closest projected images to the vertices of a grid, defined by the marginal quantiles of the principal components. (Right panel:) The images corresponding to the circled points. These show the nature of the first two principal components.*

Principal Component Analysis

Example: Procrustes Transformations and Shape Averaging

The S letter in a signature Suresh

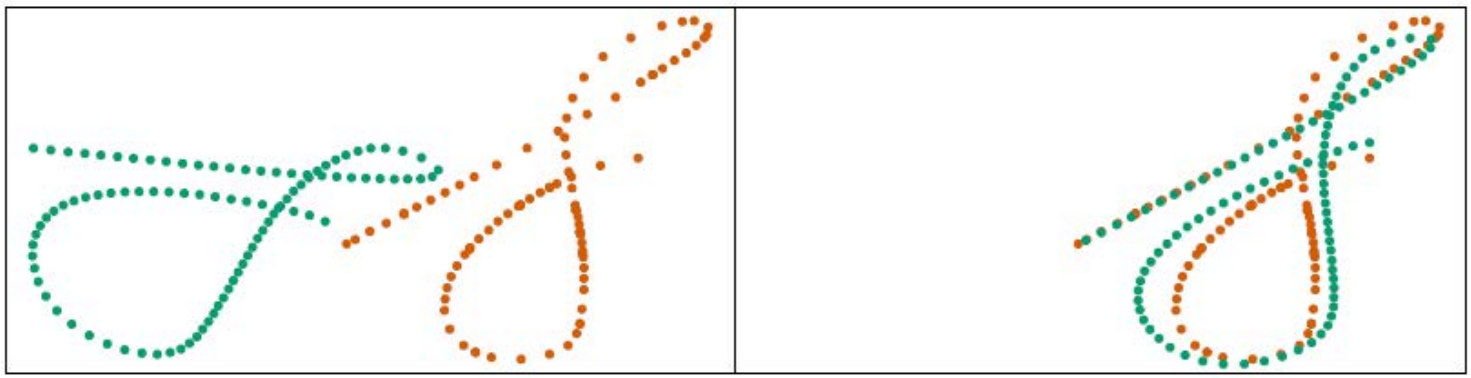


FIGURE 14.25. *(Left panel:) Two different digitized handwritten Ss, each represented by 96* corresponding *points in* $\mathbb{R}^2$. *The green S has been deliberately rotated and translated for visual effect. (Right panel:) A Procrustes transformation applies a translation and rotation to best match up the two set of points.*

Please read the example in ESL.

Principal Component Analysis

Example: Procrustes Transformations and Shape Averaging

The S letter in a signature Suresh

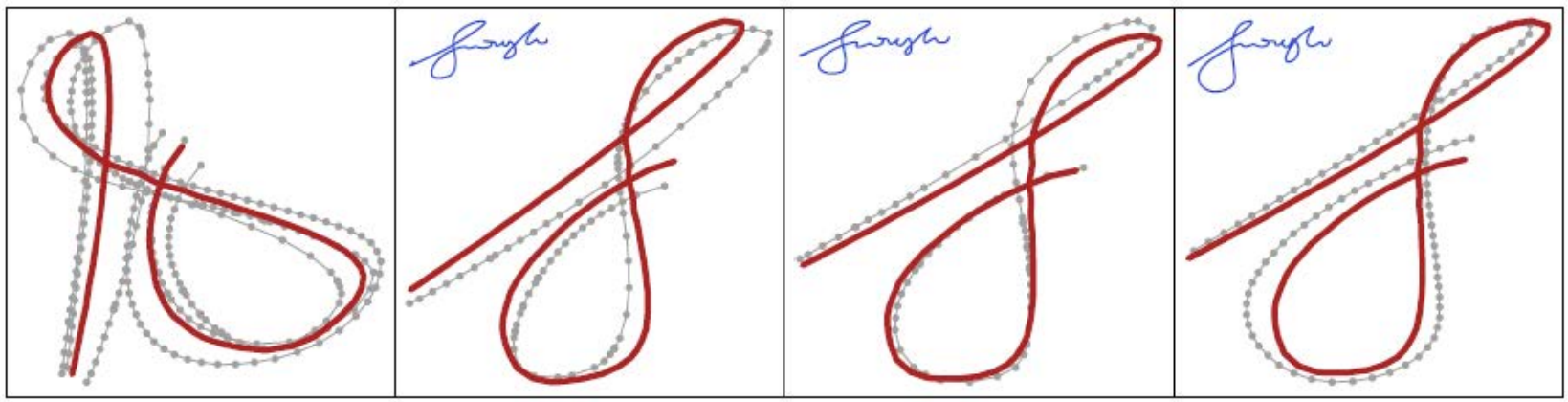


FIGURE 14.26. *The Procrustes average of three versions of the leading S in Suresh's signatures. The left panel shows the* preshape *average, with each of the shapes* $\mathbf{X}'_\ell$ *in preshape space superimposed. The right three panels map the preshape* $\mathbf{M}$ *separately to match each of the original S's.*

Please read the example in ESL.

End of Set 9