# Clustering

Egor Malkov

University of Minnesota
FRB Minneapolis

Machine Learning and Big Data Workshop
October 26, 2020

# Prologue

*This essay (...) is concerned with the similarities of culture-wholes, or considerable blocks of cultures like the Sun dance.*

`(Driver and Kroeber, 1932: origination of cluster analysis)`

# General Idea

**Clustering** is the process of grouping similar objects together.

- Objects within each cluster are more closely related to one another than objects assigned to different clusters.

Sometimes the goal is to arrange the clusters into a natural hierarchy.

- Successively group the clusters so that at each level of the hierarchy, clusters within the same group are more similar to each other than those in different groups.

Use cluster analysis to form descriptive statistics to ascertain whether or not the data consists of a set distinct subgroups, each with substantially different properties.

**Two central questions:**

1. What does "similar" mean? → dissimilarity measures.
2. How to do grouping? → clustering algorithms.

# Outline

# Inputs and Output

# Inputs for Cluster Analysis

*Specifying an appropriate dissimilarity measure is far more important in obtaining success with clustering than choice of clustering algorithm.*

(Hastie, Tibshirani, Friedman: ESL)

**Proximity Matrix**: Data is in terms of the proximity between pairs of objects.

- Respondents are asked to judge by how much certain objects differ from one another.
- If the triangle inequality $d_{ii'} \leq d_{ik} + d_{i'k}$, $\forall k \in \{1, ..., N\}$ does not hold (often!) $\Rightarrow$ the algorithms that assume *distances* cannot be used with this data.

**Feature-Based Dissimilarity**: Observations $x_{ij}$ for $i = 1, ..., N$, variables $j = 1, ..., J$.

$$\underbrace{D\left(x_i, x_{i'}\right)}_{\text{dissimilarity, objects } i \text{ and } i'} = \sum_{j=1}^{J} \underbrace{d_j\left(x_{ij}, x_{i'j}\right)}_{\text{dissimilarity, variable } j}$$

Most common choice: $d_j\left(x_{ij}, x_{i'j}\right) = \left(x_{ij} - x_{i'j}\right)^2$.

- Other choices are possible, and can lead to potentially different results.
- For nonquantitative data (e.g., categorical), squared distance may not be appropriate.
- Sometimes desirable to assign different weights to variables.

# Different Types of Data

**Quantitative Data**: Use a monotone-increasing function of the absolute difference.

$$d\left(x_i, x_{i'}\right) = f\left(\left|x_i - x_{i'}\right|\right)$$

- Squared-error loss function, identity loss function, correlation.

**Ordinal Data**: Replace $M$ original values by $\tilde{x}_i = (i - 1/2)/M$, with $i = 1, ..., M$, and then treat as quantitative variables.

**Categorical Data**: Assign the difference between pairs of values (e.g., for a variable with $M$ distinct values define matrix $L$ with $L_{mm} = 0$, $L_{mm'} = L_{m'm} \geq 0$).

- A researcher's choice!
- Rule of thumb: Assign a distance of 1 if the variables take different values, and 0 otherwise.

# Object Dissimilarity

**Goal**: Combine the variable dissimilarities into a single overall measure of dissimilarity $D(x_i, x_{i'})$ between two objects or observations $(x_i, x_{i'})$.

$$D\left(x_i, x_{i'}\right) = \sum_{j=1}^{J} \omega_j d_j\left(x_{ij}, x_{i'j}\right), \qquad \sum_{j=1}^{J} \omega_j = 1$$

Note that $\omega_j = 1, \forall j$ does not necessarily give all attributes equal influence.

- The influence of $X_j$ on $D\left(x_i, x_{i'}\right)$ depends upon its relative contribution to

$$\bar{D} = \frac{1}{N^2} \sum_{i=1}^{N} \sum_{i'=1}^{N} D\left(x_i, x_{i'}\right) = \sum_{j=1}^{J} \omega_j \times \underbrace{\frac{1}{N^2} \sum_{i=1}^{N} \sum_{i'=1}^{N} d_j\left(x_{ij}, x_{i'j}\right)}_{\bar{d}_j} \equiv \sum_{j=1}^{J} \omega_j \bar{d}_j$$

- Setting $\omega_j \sim 1/\bar{d}_j$ would give equal influence to all variables.
- With squared-error distance, $\bar{d}_j = 2\widehat{\mathrm{Var}}\left(X_j\right)$.

# Caveats

## Weighting

- Setting $\omega_j = 1/\bar{d}_j$ for all variables can be highly counterproductive. Example
- Assign a higher influence to the variables that are more relevant in separating the groups: it is possible that a clustering algorithm cannot uncover some of them.
- No substitute for careful thought in the context of each individual problem.

## Missing Observations

- Omit each observation pair $x_{ij}$, $x_{i'j}$ that has at least one missing value when computing the dissimilarity between observations $x_i$ and $x_{i'}$.
- When both observations have no measured values in common, drop them.
- Impute the missing values.
- For categorical variables, consider the value "missing" as just another category.

# Evaluating the Output of Clustering Algorithms

*The validation of clustering structures is the most difficult and frustrating part of cluster analysis. Without a strong effort in this direction, cluster analysis will remain a black art accessible only to those true believers who have experience and great courage.*

(Jain and Dubes, 1988)

Two types of output:

1. **Flat Clustering**: Partition the objects into disjoint sets.
2. **Hierarchical Clustering**: Create a nested tree of partitions.

Flat clusterings are usually faster to create, but hierarchical are often more useful.

Most hierarchical clustering algorithms are deterministic and do not require the specification of $K$, the number of clusters, whereas most flat clustering algorithms are sensitive to the initial conditions and require some model selection method for $K$.

Clustering performance measures:

- The likelihood of a test set (*probabilistic methods*).
- Purity, Rand index, mutual information (*non-probabilistic methods*).

# Clustering Algorithms

# Overview

**Goal**: Partition the observations into "clusters" so that the pairwise dissimilarities between those assigned to the same cluster tend to be smaller than those in different clusters.

**Combinatorial Algorithms**

- Work on the data with no direct reference to an underlying probability model.

**Mixture Modeling**

- Suppose that the data is an i.i.d. sample from some population described by a p.d.f.
- Characterize this p.d.f. by a mixture of component density functions (each component density describes one of the clusters).
- Use maximum likelihood or corresponding Bayesian approaches to fit the model.

**Mode Seeking ("Bump Hunting")**

- Take a nonparametric perspective by directly estimating distinct modes of the p.d.f.
- Observations "closest" to each respective mode define the individual clusters.

# Combinatorial Algorithms

The most popular clustering algorithms directly assign each observation to a group or cluster without regard to a probability model describing the data.

Each observation is uniquely labeled by an integer $i \in \{1, ..., N\}$.

A *prespecified* number of clusters $K < N$ is postulated, and each one is labeled by an integer $k \in \{1, ..., K\}$.

Each observation is assigned to one and only one cluster.

- Encoder $k = C(i)$ assigns the $i$th observation to the $k$th cluster.

Find the particular encoder $C^*(i)$ that achieves the required goal, based on the dissimilarities $d(x_i, x_{i'})$ between every pair of observations.

# Combinatorial Algorithms

Specify a loss function and minimize it using some combinatorial optimization algorithm:

$$W(C) = \frac{1}{2} \sum_{k=1}^{K} \sum_{C(i)=k} \sum_{C(i')=k} d\left(x_i, x_{i'}\right)$$

Minimize W over all possible assignments of the $N$ data points to $K$ clusters.

- Such optimization is feasible only for very small data sets.
- The number of distinct assignments grows very rapidly (e.g. with $N = 19$ and $K = 4$ we already have $\backsimeq 10^{10}$ distinct assignments).

Practical clustering algorithms examine only a very small fraction of all possible $k = C(i)$.

- Identify a small subset that is likely to contain the optimal one, or at least a good suboptimal partition.
- Such feasible strategies are based on *iterative greedy descent* (initial partition $\rightarrow$ iterate by changing the cluster assignments in such a way that the value of the criterion is improved from its previous value $\rightarrow$ stop when no improvement).

# K-Means Algorithm

The K-means algorithm is one of the most popular iterative descent clustering methods.

Quantitative data + (potentially weighted) squared Euclidean distance:

$$d\left(x_i, x_{i'}\right) = \sum_{j=1}^{J} \left(x_{ij} - x_{i'j}\right)^2 = ||x_i - x_{i'}||^2$$

Loss function:

$$W(C) = \frac{1}{2} \sum_{k=1}^{K} \sum_{C(i)=k} \sum_{C(i')=k} ||x_i - x_{i'}||^2 = \sum_{k=1}^{K} N_k \sum_{C(i)=k} ||x_i - \bar{x}_k||^2 \to \min$$

1. Start with a guess for the cluster centers.
2. Identify the closest cluster center for each point.
3. Replace each cluster center by the coordinate-wise average of all data points that are closest to it. ▸ Example

# K-Means Algorithm

▸ K-Medoids

1. For a given cluster assignment $C$, the total cluster variance

$$\min_{\{m_k\}_{k=1}^{K}} \sum_{k=1}^{K} N_k \sum_{C(i)=k} ||x_i - m_k||^2$$

is minimized with respect to $\{m_1, ..., m_K\}$ yielding the means of the currently assigned clusters

$$\bar{x}_S = \arg\min_m \sum_{i \in S} ||x_i - m||^2$$

2. Given a current set of means $\{m_1, ..., m_K\}$, the total cluster variance is minimized by assigning each observation to the closest (current) cluster mean. That is,

$$C(i) = \arg\min_{1 \le k \le K} ||x_i - m_k||^2$$

3. Steps 1 and 2 are iterated until the assignments do not change.

# K-Means Algorithm as EM Algorithm

$$\mathcal{L}(\boldsymbol{\theta}) = \sum_{i=1}^{N} \log p\left( \underbrace{\boldsymbol{x}_i}_{\text{observed}} |\boldsymbol{\theta} \right) = \sum_{i=1}^{N} \log \left[ \sum_{\boldsymbol{z}_i} p\left( \boldsymbol{x}_i, \underbrace{\boldsymbol{z}_i}_{\text{hidden or missing}} |\boldsymbol{\theta} \right) \right] \rightarrow \max$$

**Expectation Maximization Algorithm**

EM algorithm exploits the fact that if the data were fully observed, then the max likelihood/max a posteriori probability (ML/MAP) estimate would be easy to compute.

Iterative algorithm which alternates between inferring the missing values given parameters (*E step*), and then optimizing the parameters given the "filled in" data (*M step*).

Complete data log likelihood: $\mathcal{L}_c(\boldsymbol{\theta}) \coloneqq \sum_{i=1}^{N} \log p(\boldsymbol{x}_i, \boldsymbol{z}_i|\boldsymbol{\theta})$.

Expected complete data log likelihood (iteration $t$): $Q\left(\boldsymbol{\theta}, \boldsymbol{\theta}^{t-1}\right) = \mathbb{E}\left[\mathcal{L}_c(\boldsymbol{\theta}) |\mathcal{D}, \boldsymbol{\theta}^{t-1}\right]$.

**E Step**: Compute the *expected sufficient statistics* $Q\left(\boldsymbol{\theta}, \boldsymbol{\theta}^{t-1}\right)$.
**M Step**: $\boldsymbol{\theta}^t = \arg\max_{\boldsymbol{\theta}} Q\left(\boldsymbol{\theta}, \boldsymbol{\theta}^{t-1}\right)$.

# K-Means Algorithm as EM Algorithm

**E Step**: Find the Euclidean distance between $N$ observations and $K$ cluster centers.

**M Step**: Given the hard cluster assignments, update each cluster center by computing the mean of all points assigned to it.

Repeat the following steps until convergence:

1. Assign each data point to its closest cluster center:

$$C(i) = \arg \min_{1 \leq k \leq K} ||\boldsymbol{x}_i - \boldsymbol{\mu}_k||_2^2$$

2. Update each cluster center by computing the mean of all points assigned to it:

$$\boldsymbol{\mu}_k = \frac{1}{N_k} \sum_{i:C(i)=k} \boldsymbol{x}_i$$

This is called *hard EM* (deterministic assignments). Use *soft EM* (probabilistic assignments) for estimating a Gaussian mixture model.

# Vector Quantization

We can often see the photo requirements like...

- In JPEG (.jpg) file format
- Equal to or less than 240 kB (kilobytes) in file size
- In a square aspect ratio (height must equal width)
- 600x600 pixels in dimension

K-means clustering algorithm represents a key tool in the area of image and signal compression (*vector quantization*). ▸ Example

K-means clustering algorithm (Lloyd's algorithm) approximates each pixel block by its closest cluster centroid.

# K-Medoids Clustering

K-means: quantitative data + squared Euclidean distance.

- Lack robustness against outliers that produce very large distances.

K-medoids uses an *actual* point as the center of a cluster instead of the *mean* point.

- **Medoid** is the most centrally located object of the cluster, with minimum sum of distances to other points. ▸ Example

Replace the first (minimization) step in ▸ K-Means by an explicit optimization with respect to $\{m_1, ..., m_K\}$.

- The only step of the K-means algorithm that assumes squared Euclidean distance.

K-medoids clustering can also be applied to data described by proximity matrices.

- K-means needs feature-based dissimilarity measures as inputs.

**Cost**: K-medoids is far more computationally intensive than K-means.

# K-Medoids Clustering

1. For a given cluster assignment $C$ find the *observation* in the cluster minimizing total distance to other points in that cluster:

$$i_k^* = \arg \min_{i:C(i)=k} \sum_{C(i')=k} D\left(x_i, x_{i'}\right)$$

   Then $m_k = x_{i_k^*}$, $k = 1, 2, ..., K$ are the current estimates of the cluster centers.

2. Given a current set of cluster centers $\{m_1, ..., m_K\}$, minimize the total error by assigning each observation to the closest (current) cluster center:

$$C(i) = \arg \min_{1 \leq k \leq K} D\left(x_i, m_k\right)$$

3. Iterate steps 1 and 2 until the assignments do not change.

# Affinity Propagation

K-medoids algorithm can suffer from local minima problem.

**Affinity Propagation**

Works substantially better in practice (Frey and Dueck, 2007).

Each data point must choose another data point as its centroid (some data points will choose themselves as centroids).

- This will *automatically* determine the number of clusters.

The algorithm proceeds by updating two matrices until either the cluster boundaries remain unchanged over a number of iterations, or some # of iterations is reached:

- The "responsibility" matrix $R$ containing values $r(i, k)$ that quantify how well-suited $x_k$ is to serve as the centroid for $x_i$, relative to other candidate centroids for $x_i$.
- The "availability" matrix $A$ containing values $a(i, k)$ that represent how "appropriate" it would be for $x_i$ to pick $x_k$ as its centroid, taking into account other points' preference for $x_k$ as a centroid.

# Practical Issues: Number of Clusters and Initialization

In order to apply K-means or K-medoids a researcher must select $K$ and an initialization.

### Initialization

Specify an initial set of centers $\{m_1, ..., m_K\}$ or $\{i_1, ..., i_K\}$ or an initial encoder $C(i)$.

Suggestions range from simple random initialization to a deliberate strategy based on forward stepwise assignment.

### Number of Clusters

A choice for the number of clusters $K$ depends on the goal.

Look for a kink in the sum of squares curve (or its logarithm) as a function of $K$ to locate the optimal number of clusters.

An automatic way of locating this kink — Gap statistic (Tibshirani et al., 2001). ▸ Example

- Compare the curve $\log W_K$ to the curve obtained from data uniformly distributed over a rectangle containing the data.
- Optimal number of clusters — the gap between the two curves is largest.

# Hierarchical Clustering

Hierarchical clustering methods produce hierarchical representations in which the clusters at each level of the hierarchy are created by merging clusters at the next lower level.

- At the lowest level, each cluster contains a single observation. At the highest level there is only one cluster containing all of the data.

Do not require to specify the number of clusters and a starting configuration assignment.

- Need to specify a measure of dissimilarity between (disjoint) groups of observations, based on the pairwise dissimilarities among the observations in the two groups.

Strategies for hierarchical clustering: **agglomerative** and **divisive**.

**Dendrogram** is a highly interpretable description of the hierarchical clustering. ▸ Example

- Dissimilarity between merged clusters is monotone increasing with the level of merger.
- **Be careful!** Different hierarchical methods, as well as small changes in the data, can lead to quite different dendrograms.
- **Be careful!** Hierarchical methods impose hierarchical structure whether or not such structure actually exists in the data.
- Probabilistic version of hierarchical clustering solves both these problems.

# Agglomerative Clustering

Start at the bottom and at each level recursively merge a selected pair of clusters into a single cluster.

- The pair chosen for merging consists of the two groups with the smallest dissimilarity.

Need to define a measure of dissimilarity between two clusters!

Dissimilarity $d(G, H)$ between groups $G$ and $H$. ▸ Figure

- Single linkage: $d_{SL}(G, H) = \min_{i \in G, i' \in H} d_{ii'}$.
- Complete linkage: $d_{CL}(G, H) = \max_{i \in G, i' \in H} d_{ii'}$.
- Group average: $d_{GA}(G, H) = \frac{1}{N_G N_H} \sum_{i \in G} \sum_{i' \in H} d_{ii'}$.

**Be careful!** Small dissimilarities — results may differ. ▸ Example

To increase speed, for large $N$, a common heuristic is to first run K-means and then apply hierarchical clustering to the estimated cluster centers.

# Divisive Clustering

Start at the top and at each level recursively split one of the existing clusters at that level into two new clusters.

- Produce two new groups with the largest between-group dissimilarity.

Divisive clustering is less popular than agglomerative clustering, but it has advantages.

- Faster.
- Splitting decisions are made in the context of seeing all the data, whereas agglomerative methods make myopic merge decisions.

**Implementation**

- The divisive paradigm can be employed by recursively applying any of the combinatorial methods such as K-means or K-medoids, with $K = 2$, to perform the splits at each iteration.
  - Depends on the starting configuration specified at each step.
- Recursive splitting procedures proposed by Macnaughton-Smith et al. (1964), etc.

# Mode Seeking ("Bump Hunting")

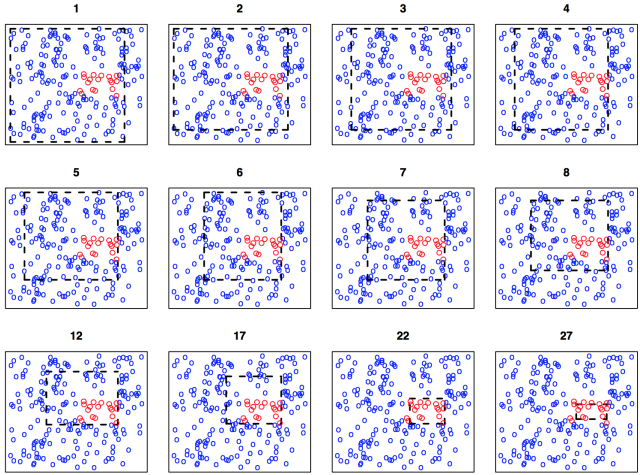
**PRIM Algorithm** is one example: in the variable space find boxes in which the response average is high (finding maxima in the target function, or *bump hunting*).

**PRIM (Patient Rule Induction Method) Algorithm**

1. Start with all of the training data, and a maximal box containing all of the data.
2. Shrink the box by compressing one face, so as to peel off the fraction $\alpha$ of observations having either the highest values of a predictor $X_j$, or the lowest. Choose the peeling (e.g., $\alpha = 0.05$) that produces the highest response mean in the remaining box.
3. Repeat step 2 until some minimal number of observations (say 10) remain in the box.
4. Expand the box along any face, as long as the resulting box mean increases.
5. Steps 1-4 give a sequence of boxes, with different numbers of observations in each box. Use cross-validation to choose a member of the sequence. Call the box $B_1$.
6. Remove the data in box $B_1$ from the dataset and repeat steps 2-5 to obtain a second box, and continue to get as many boxes as desired.

# PRIM Algorithm: An Example

Box Mean




Source: Hastie, Tibshirani, and Friedman.

# Concluding Remarks

Wide area of applications (below are examples just for K-means).

- Worker and firm heterogeneity & earnings dispersion (Bonhomme et al., 2019).
- Customer segmentation (Mihai-Florin et al., 2012).
- Insurance fraud detection (Ghorbani and Farzai, 2018).
- Identifying crime localities (Kumar Tayal et al., 2015).
- Goods delivery optimization (Mourelo Ferrandez et al., 2016).
- etc.

**Be careful** because clustering is art!

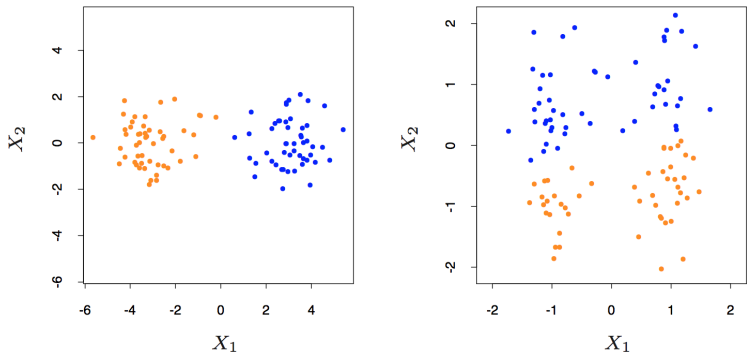
Consider investing time in Bayesian methods as they have several advantages over traditional distance-based agglomerative clustering algorithms.

- Heller and Ghahramani (2005).

# Appendix

# Variables Weighting

Back



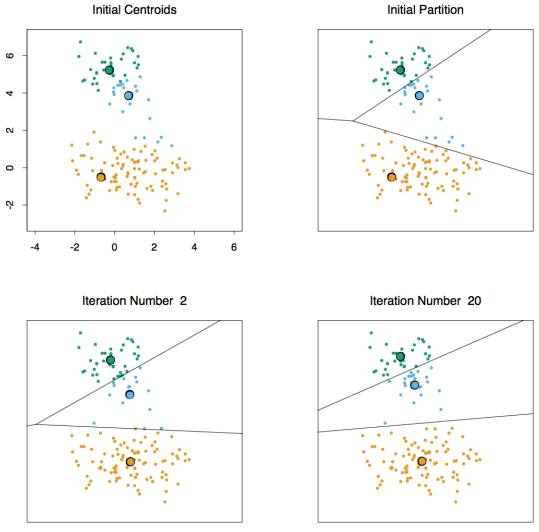

**Left:** K-means clustering ($K = 2$) applied to the *raw data*.

**Right:** K-means clustering applied to the *standardized data*, i.e. use weights $1/2\widehat{\mathrm{Var}}\,(X_j)$.

Source: Hastie, Tibshirani, and Friedman.

# K-Means Clustering Algorithm

Back




Source: Hastie, Tibshirani, and Friedman.

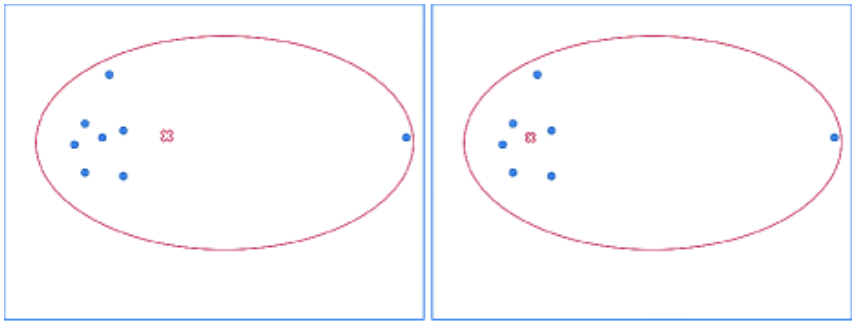
# Vector Quantization

Back

Source: Hastie, Tibshirani, and Friedman.

# Mean vs. Medoid

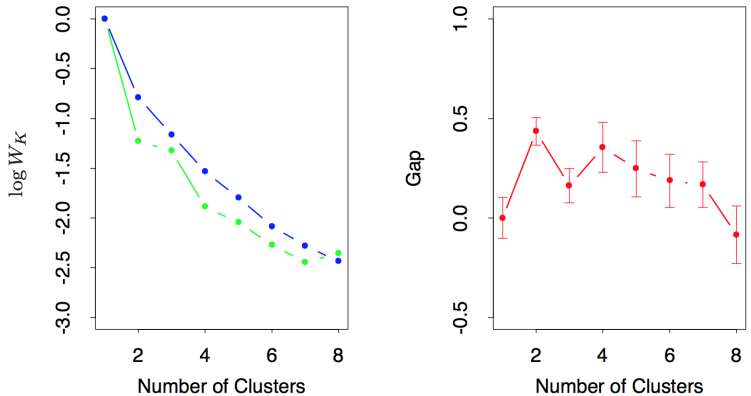
Back




Source: Jin and Han (2011).

# Choice of Number of Clusters

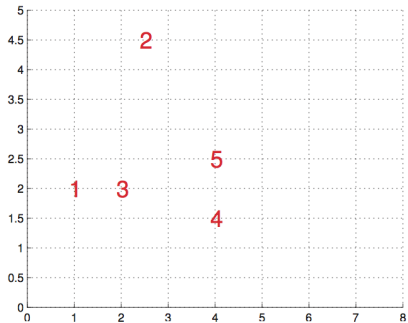
Back

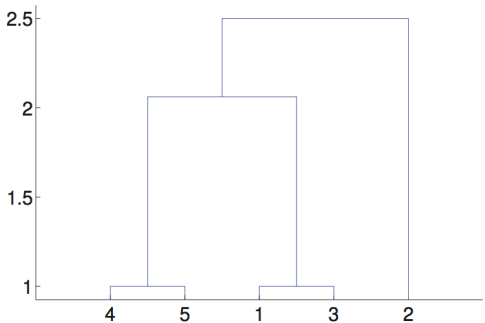



Notes: Left — observed (green) and expected (blue) values of $\log W_K$. Both curves are translated to equal zero at one cluster. Right — Gap curve, equal to the difference between the observed and expected values of $\log W_K$. The Gap estimate $K^*$ is the smallest $K$ producing a gap within one standard deviation of the gap at $K+1$; here $K^* = 2$. Source: Hastie, Tibshirani, and Friedman.

# Dendrogram

Back

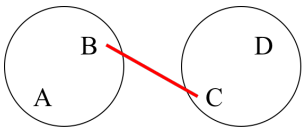


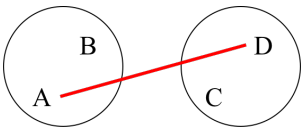

Notes: Left — An example of single link clustering using city block distance. Pairs (1,3) and (4,5) are both distance 1 apart, so get merged first. Right — The resulting dendrogram. If we cut the tree at any given height, we induce a clustering of a given size. Source: Murphy, based on Figure 7.5 of Alpaydin (2004).

# Linkages in Agglomerative Clustering

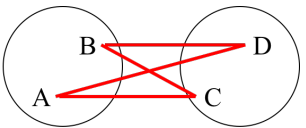
Back




Single linkage



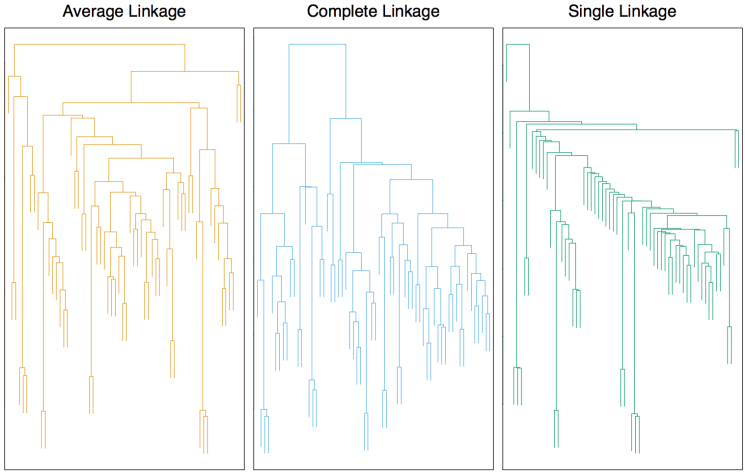

Complete linkage




Average linkage

# Linkages in Agglomerative Clustering

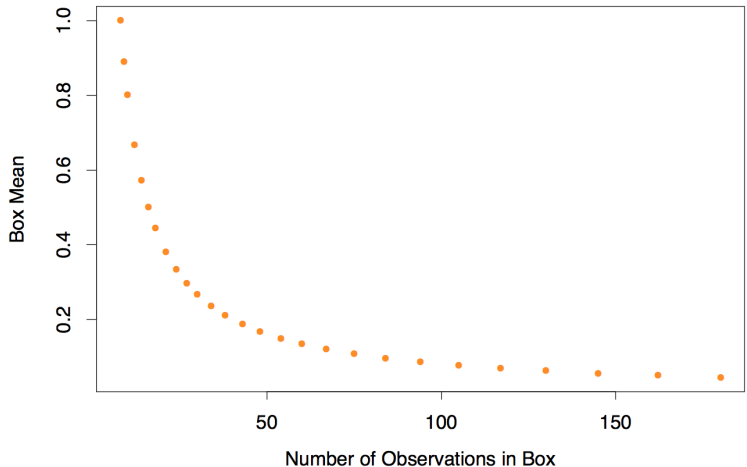
Back




Source: Hastie, Tibshirani, and Friedman.

# PRIM Algorithm: An Example

Back




Source: Hastie, Tibshirani, and Friedman.