CS6375: Machine Learning

Gautam Kunapuli

Clustering

THE UNIVERSITY OF TEXAS AT DALLAS
Erik Jonsson School of Engineering and Computer Science

Supervised vs Unsupervised Learning

Supervised learning: Given **labeled** data $(\boldsymbol{x}_i, y_i), i = 1, \ldots, n$, learn a function $f : \boldsymbol{x} \rightarrow y$

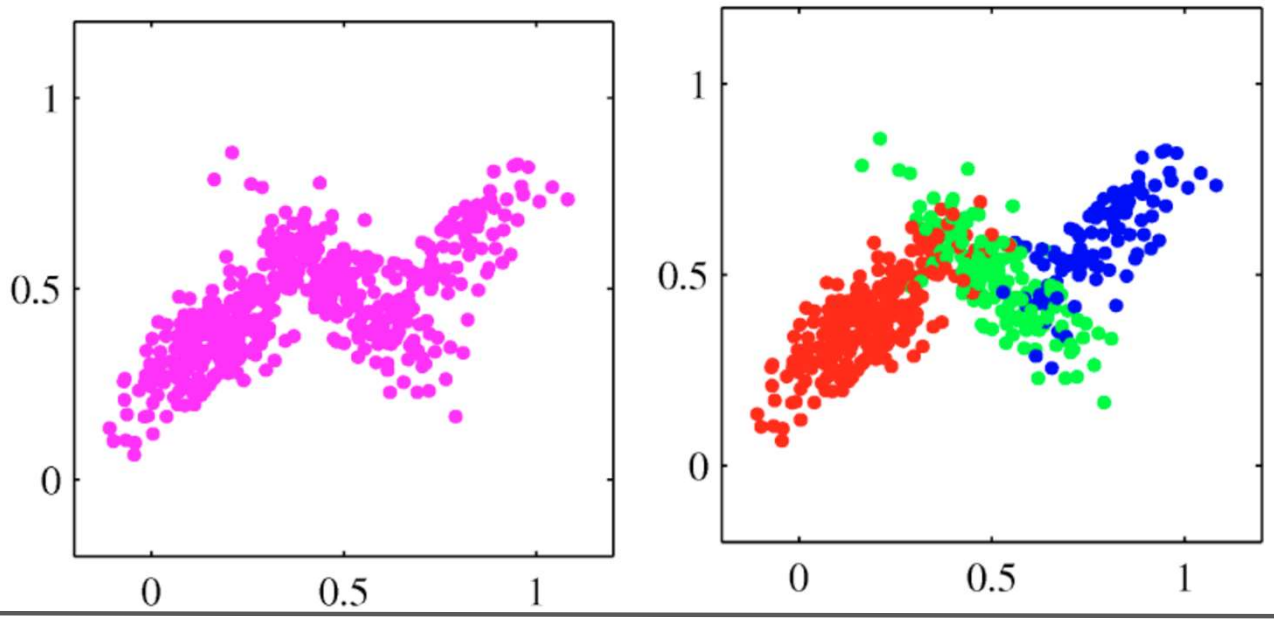
- Categorical y : **classification**
- Continuous y : **regression**

Unsupervised learning: Given **unlabeled** data $\boldsymbol{x}_i, i = 1, \ldots, n$, can we infer the underlying structure of X?

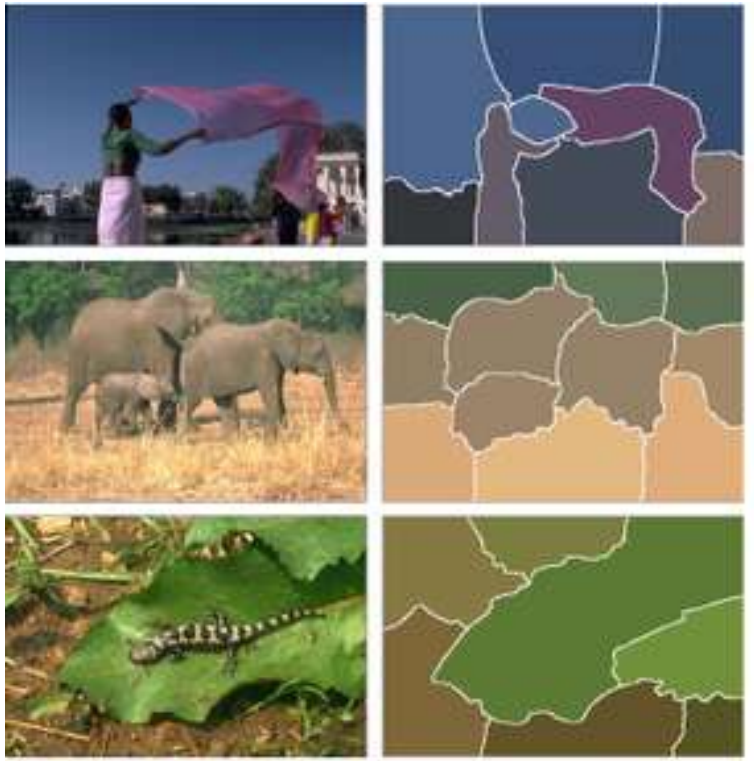
Why do unsupervised learning?

- raw data cheap; labeled data expensive
- save memory/computation.
- **reduce noise** in high-dimensional data
- useful in **exploratory data analysis**
- **pre-processing** for supervised learning
  - e.g., pca for dimensionality reduction

Basic Idea for Clustering: **discover groups** such that samples **within a group are more similar** to each other **than samples across groups**

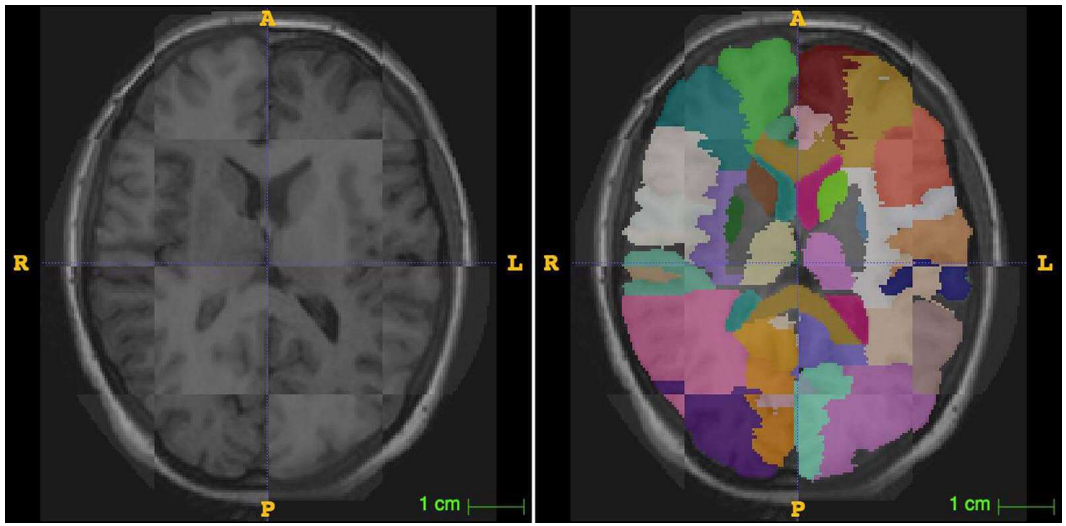


Example: Image Segmentation



Example: Partition a digital image of pixels into **segments** (also known as super-pixels). The goal of **segmentation** is to extract a **higher-order representation** that is more meaningful and (semantically) easier to analyze.

Medical image segmentation partitions a medical image into different meaningful segments. Segments often correspond to different tissues, organs, diseases, pathologies. Medical image segmentation is challenging due to low **contrast**, **noise**, **differences in individuals** etc.,



k-Means Clustering

Ingredients of k-means

- a **distance function** to identify the "closest" cluster centers
- a **loss function** to evaluate clusters
- an **algorithm** that optimizes this loss function

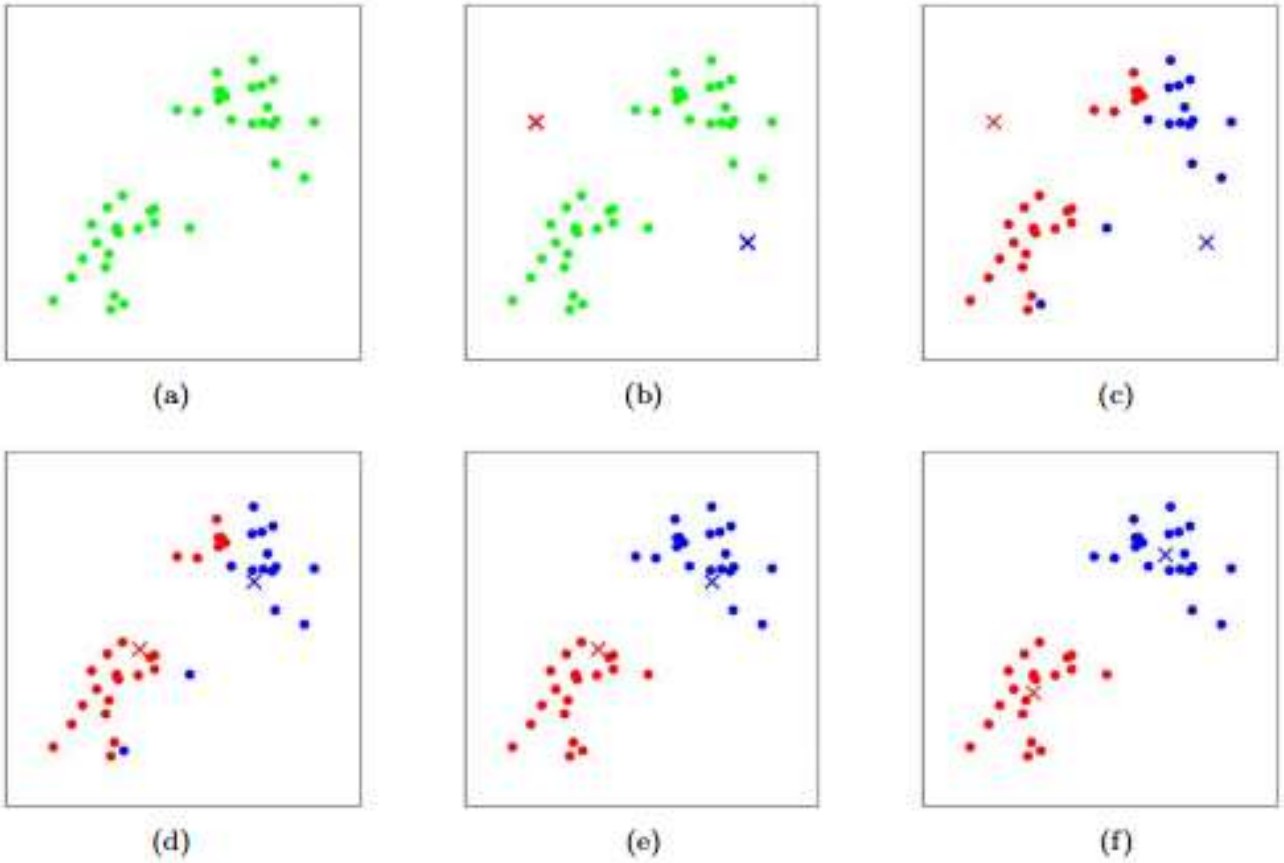
k-means Clustering

Given: Unlabeled data, $\boldsymbol{x}_i, i = 1, \ldots, n$

Initialize: Pick k random points as cluster centers $\mu_j, j = 1, \ldots, k$

while not converged do

- **Assign** data points $\boldsymbol{x}_i$ to **closest** cluster center μ_j
- **Update** the cluster centers μ_j to the mean (average) of the points assigned to that cluster
- if the assignments no longer change, **converged** = **true**



k-Means: Distance Functions

Properties of a distance function (*also applies to k-nearest neighbors*)

- **symmetry** $d(\boldsymbol{x}, \boldsymbol{z}) = d(\boldsymbol{z}, \boldsymbol{x})$
  - if symmetry does not hold, we can say that $\boldsymbol{x}$ looks like $\boldsymbol{z}$, but $\boldsymbol{z}$ does not look like $\boldsymbol{x}$, which is not meaningful
  - Euclidean distance is symmetric, but KL divergence is not!
- **positivity**, $d(\boldsymbol{x}, \boldsymbol{z}) \geq 0$; and self-similarity $d(\boldsymbol{x}, \boldsymbol{z}) = 0$ if and only if $\boldsymbol{x} = \boldsymbol{z}$
  - if these do not hold, the distance function cannot tell apart two different objects, which is not useful
- **triangle inequality**: $d(\boldsymbol{a}, \boldsymbol{b}) + d(\boldsymbol{b}, \boldsymbol{c}) \geq d(\boldsymbol{a}, \boldsymbol{c})$
  - if the triangle inequality does not hold, we can say $\boldsymbol{a}$ is like $\boldsymbol{b}$ and $\boldsymbol{b}$ is like $\boldsymbol{c}$ but $\boldsymbol{a}$ is not like $\boldsymbol{c}$, which is not meaningful

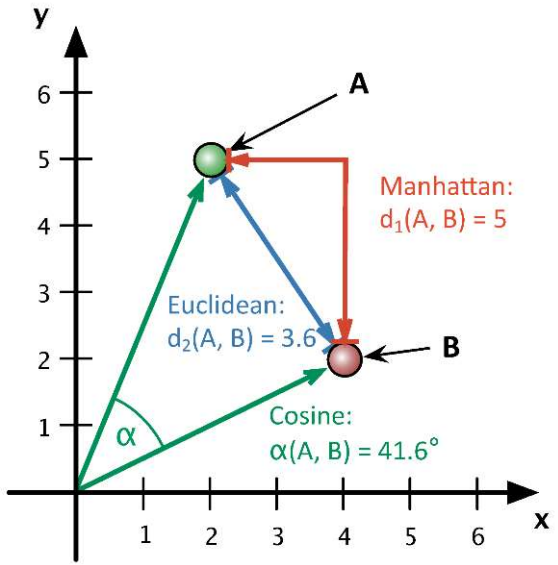
Two commonly-used distance measures produce two commonly used clustering algorithms. Consider a d-dimensional data set (d features) with points $\boldsymbol{x}, \boldsymbol{z}$

k-Means uses Euclidean distance:

$$d_2(\boldsymbol{x}, \boldsymbol{z}) = \|\boldsymbol{x} - \boldsymbol{z}\|_2 = \sqrt{(x_1 - z_1)^2 + \cdots + (x_d - z_d)^2}$$

k-Medioids uses Manhattan distance:

$$d_1(\boldsymbol{x}, \boldsymbol{z}) = \|\boldsymbol{x} - \boldsymbol{z}\|_1 = |x_1 - z_1| + \cdots + |x_d - z_d|$$



k-Means: Loss Function

The **key idea** behind **k-means clustering** is to find k clusters each described by a **prototype** (cluster centers) $\boldsymbol{\mu}_j, j = 1, \ldots, k$

- Assignment of training example $\boldsymbol{x}_i$ to clusters can be represented by **responsibilities** $r_{ij} \in \{0, 1\}$
  - $r_{ij} = 1$ if example $\boldsymbol{x}_i$ is assigned to the j-th cluster
  - need to ensure that $\sum_{j=1}^{k} r_{ij} = 1$ to ensure that the example $\boldsymbol{x}_i$ is assigned to one and only one cluster
  - cluster assignments for each data point can be read off the **responsibility matrix**
  - columns sum give us the **size of each cluster**

responsibilities for a data set with 6 examples and 3 clusters

$$\begin{array}{c} \\ \boldsymbol{x}_1 \\ \boldsymbol{x}_2 \\ \boldsymbol{x}_3 \\ \boldsymbol{x}_4 \\ \boldsymbol{x}_5 \\ \boldsymbol{x}_6 \end{array} \begin{array}{c} \begin{array}{ccc} \text{cl}_1 & \text{cl}_2 & \text{cl}_3 \end{array} \\ \begin{bmatrix} 0 & 0 & 1 \\ 0 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix} \end{array}$$

The responsibilities can be used to define a **loss function**:

$$J = \sum_{i=1}^{n} \sum_{j=1}^{k} r_{ij} \cdot d(\boldsymbol{x}_i, \boldsymbol{\mu}_j)$$

- if a training example is assigned to a cluster that is not closest to it, the loss function will **increase**
- e.g., $\boldsymbol{x}_i$ is closer to cluster **3** rather than to cluster **5**; it should be assigned to cluster **3**, otherwise the loss function will be higher since $d(\boldsymbol{x}_i, \boldsymbol{\mu}_5) > d(\boldsymbol{x}_i, \boldsymbol{\mu}_3)$

the loss function depends on the choice of **distance function**, which is **application dependent**

- need to consider the type of features
  - **Categorical**, **ordinal** or **continuous**
- can learn distance function from data
  - ML research area called **metric learning**

k-Means: Algorithm

How do we minimize loss with respect to assignments and cluster centers $(r_{ij}, \boldsymbol{\mu}_j)$?

- Chicken and egg problem
- If prototypes $(\boldsymbol{\mu}_j)$ known, can assign data points to clusters and get responsibilities (r_{ij})
- If responsibilities (r_{ij}) known, can compute prototypes $(\boldsymbol{\mu}_j)$ by averaging data points in each cluster

Objective

$$\min_{\mu}\min_{C} \sum_{i=1}^{k} \sum_{x \in C_i} |x - \mu_i|^2$$

1. Fix μ, optimize C:

$$\min_{C} \sum_{i=1}^{k} \sum_{x \in C_i} |x - \mu_i|^2 = \min_{c} \sum_{i}^{n} \left|x_i - \mu_{x_i}\right|^2$$

Step 1 of kmeans

▶ **Implementing Step 1**: fix prototypes, update cluster assignment of data points to closest prototype

2. Fix C, optimize μ:

$$\min_{\mu} \sum_{i=1}^{k} \sum_{x \in C_i} |x - \mu_i|^2$$

– Take partial derivative of μ_i and set to zero, we have

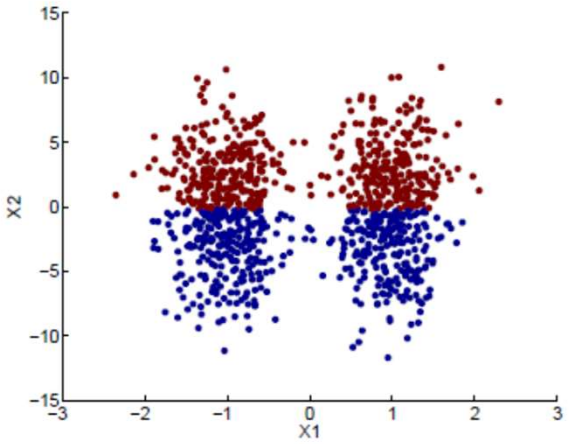
$$\mu_i = \frac{1}{|C_i|} \sum_{x \in C_i} x$$

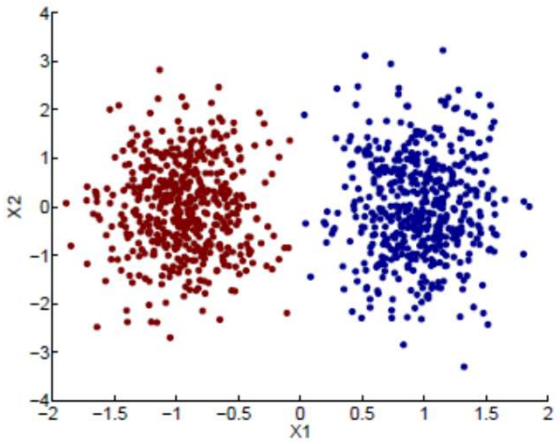
Step 2 of kmeans

▶ **Implementing Step 2**: fix cluster assignments, update prototypes using data points

- **guaranteed to converge** in a **finite** number of iterations
- **alternating minimization algorithm** is a variant of **expectation-maximization** (EM)
- Running time per iteration
  - E-step: fix prototypes, update assignment of data points to closest prototype is $O(kn)$
  - M-step: fix cluster assignments, update prototypes using data points is $O(n)$

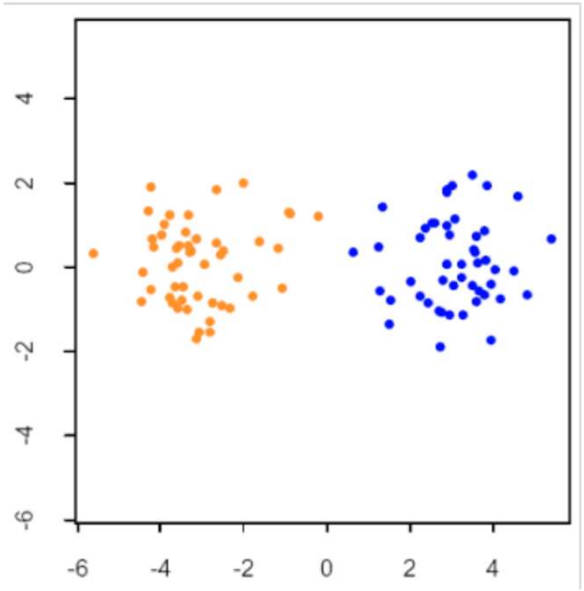
To Standardize or To Not Standardize?

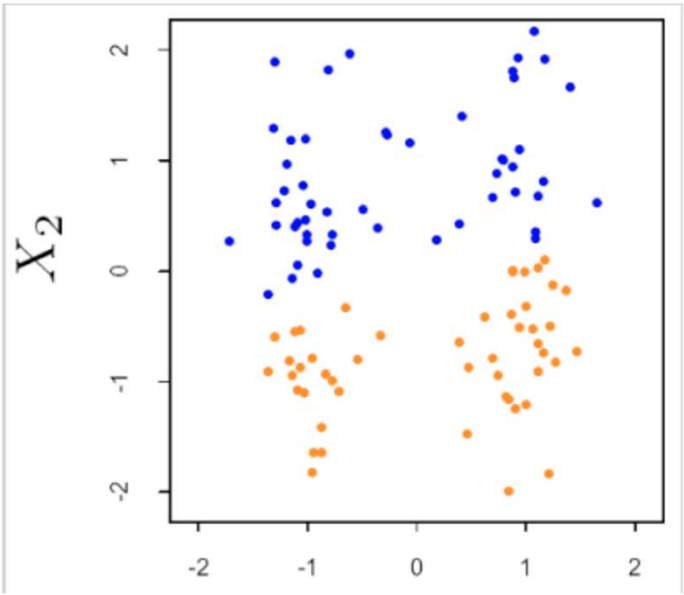




Without standardization

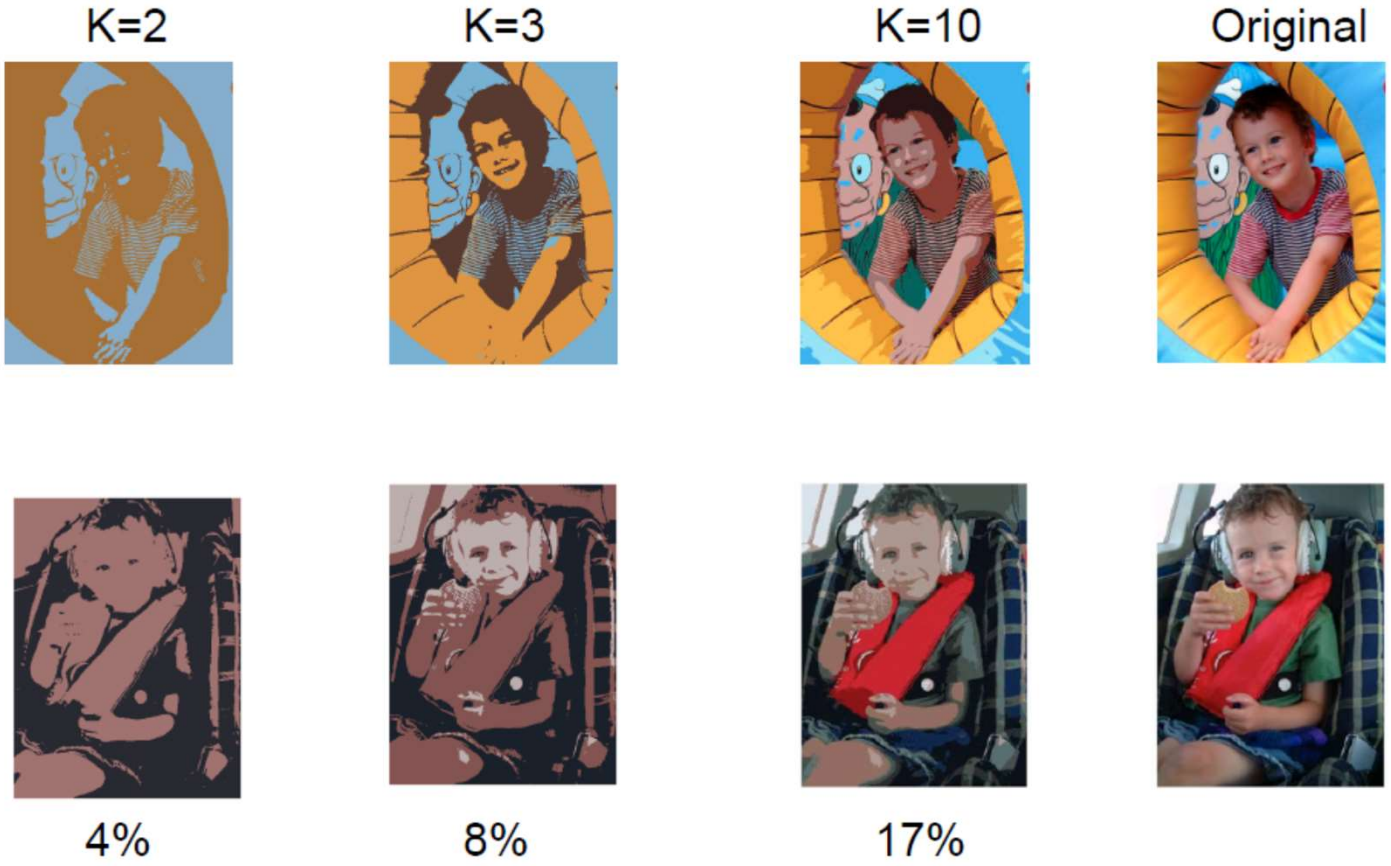
With standardization





How To Select k?

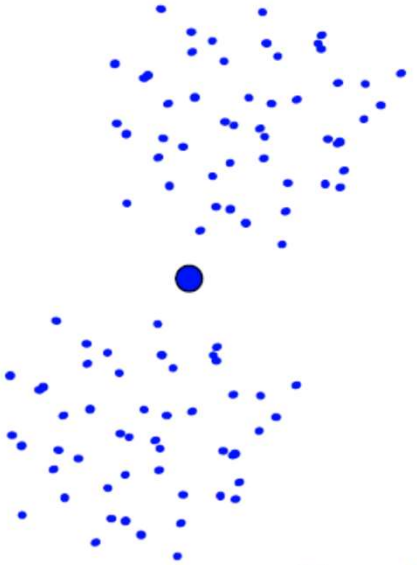
- **Cross-validation**: Partition data into two sets. Estimate prototypes on train and use these to compute the loss function on validation
- **Stability of clusters**: Measure the change in the clusters obtained by resampling or splitting the data.
- **Non-parametric approach**: Place a prior on k
- **Gap statistic**: select a k such that the "compactness of clustering" is best compared to a reference distribution (that has no obvious clustering) see [Tibshirani et al., 2001] for more details



Limitations of k-Means

- k-means will **converge to a local minima**
- **different initializations** can lead to very **different results**
- **run k-means several times** with **random** starting points, pick clustering with **smallest loss**

Would be better to have one cluster here



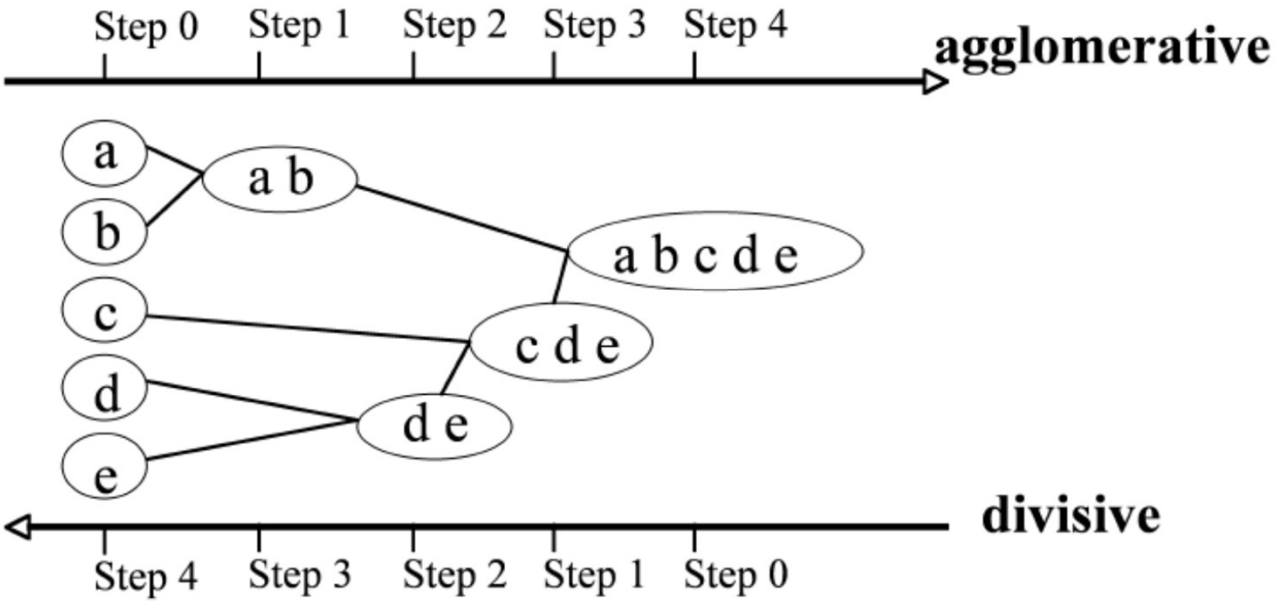
… and two clusters here

- k-means performs **hard clustering**
  - cluster assignments using **responsibilities** $r_{ij} \in \{0, 1\}$
  - a small perturbation to a data point (noise) to flip it to another cluster (instability)
  - assumes spherical clusters and equal probabilities for each cluster
- **soft clustering** can **soften** assignments to a range $r_{ij} \in [0, 1]$
  - interpretation: training example can now belong to more than one cluster with a probability r_{ij}
  - approaches: fuzzy clustering, **Gaussian mixture models**
  - compare with perceptron vs. logistic regression

Hierarchical Clustering

Another limitation of k-means: clusters change arbitrarily for different k

Solution: organize clusters in a hierarchical way by grouping similar clusters

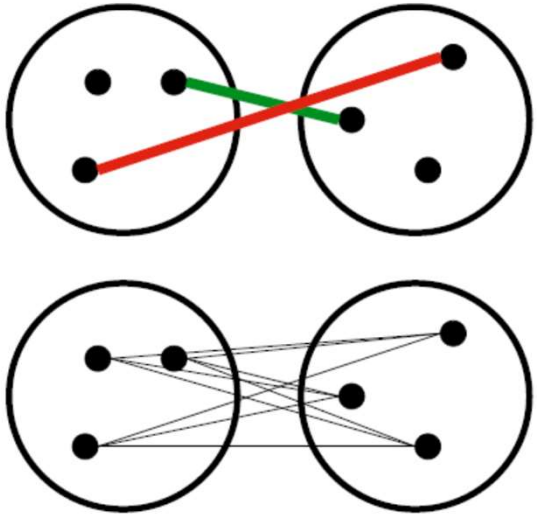


- **Bottom-up (agglomerative)**: Recursively merge two groups with the smallest **between-cluster similarity**
- **Top-down (divisive)**: Recursively split a **least-coherent** (e.g. largest diameter) cluster
- Users can then choose a cut through the hierarchy to represent the most natural division into clusters (e.g. where intergroup similarity exceeds some threshold).

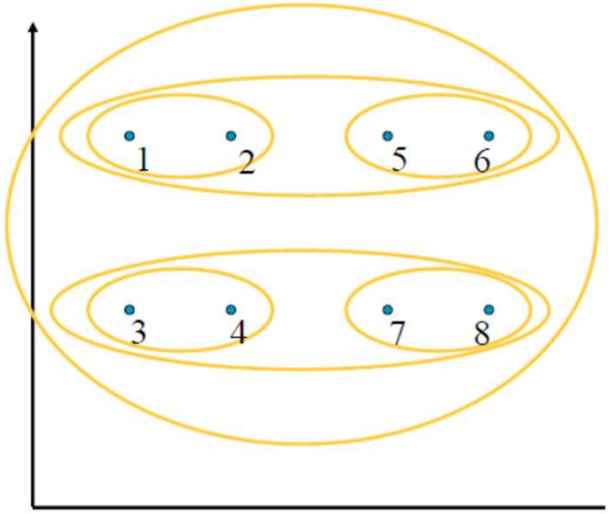
Agglomerative Clustering

Algorithm:
- **Initialize**: each training example is its own cluster
- **repeat until** only one cluster left
  - pick two "closest" clusters
  - merge them into a new cluster

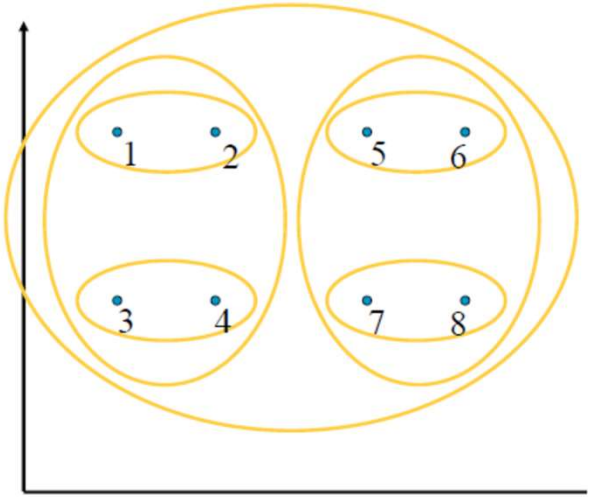
How do we define "closest" for clusters?
Closest pair (single-link clustering) tends to yield elongated clusters
Farthest pair (complete-link clustering) tends to yield rounder, more spherical clusters
Average of all pairs trades-off between single and complete link



Closest pair
(single-link clustering)



Farthest pair
(complete-link clustering)



Agglomerative Clustering

Algorithm:
- **Initialize**: each training example is its own cluster
- **repeat until** only one cluster left
  - pick two "closest" clusters
  - merge them into a new cluster

How do we define "closest" for clusters?
Closest pair (single-link clustering) tends to yield elongated clusters
Farthest pair (complete-link clustering) tends to yield rounder, more spherical clusters
Average of all pairs trades-off between single and complete link

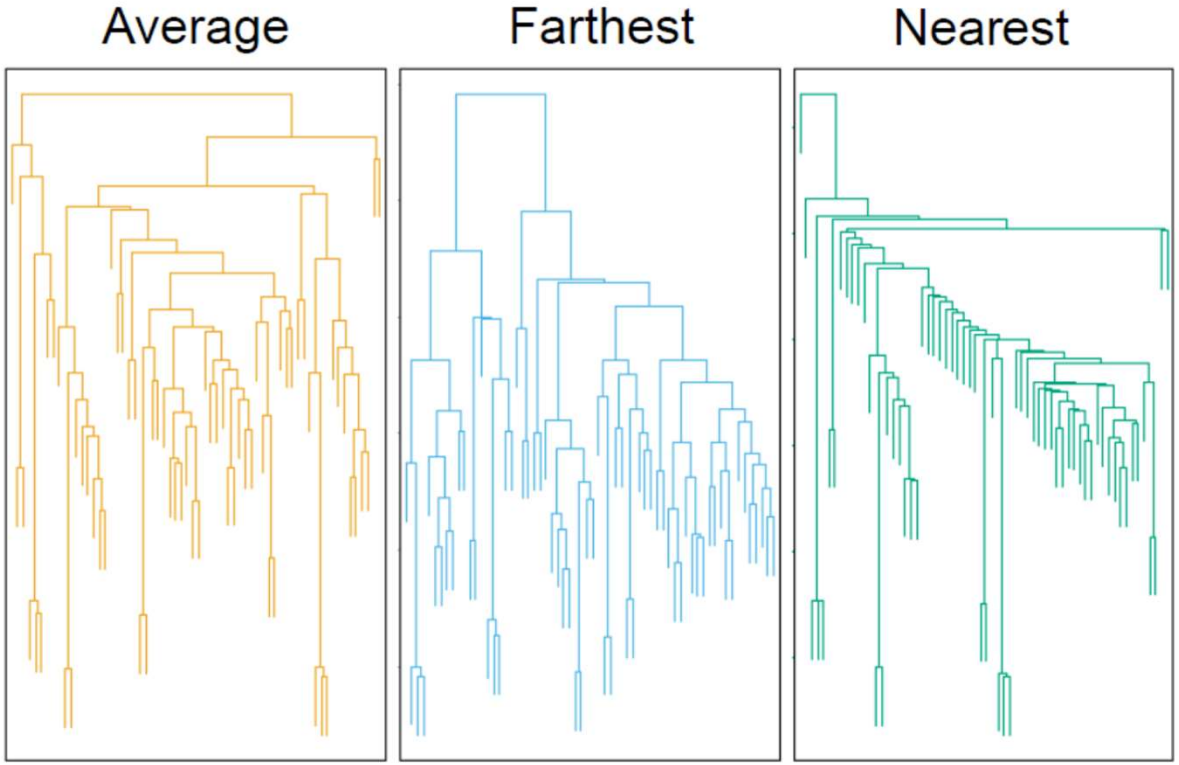
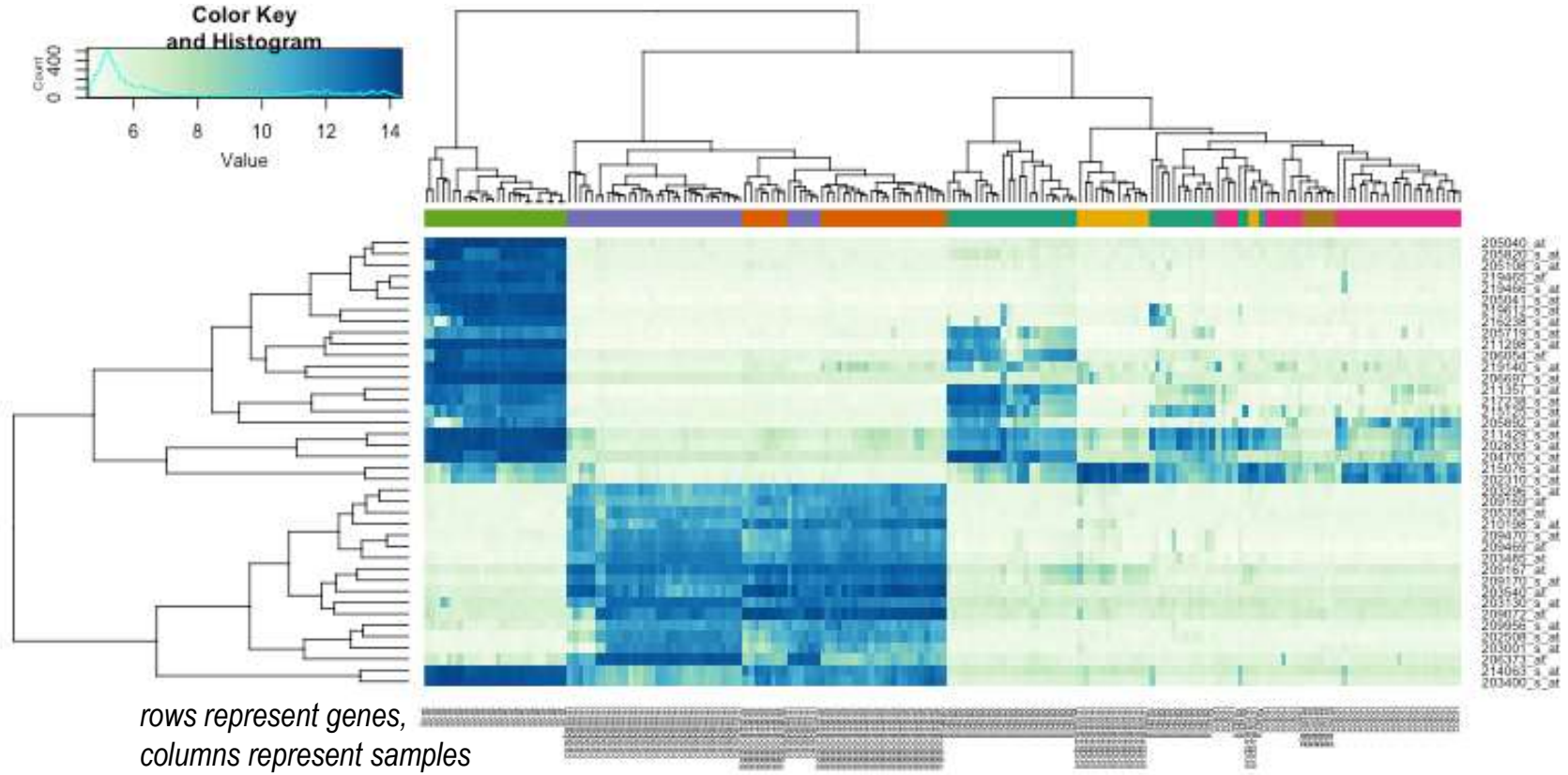


Figure by Hastie et al.,

Example: Gene Expression Analysis

Example: Discover patterns in **gene expression data**; e.g., new types of cancer from gene expression profiles, drug responses from genotypes etc.,

Gene expression analysis makes extensive use of **hierarchical clustering**, where **each example is assigned its own cluster**, and then clusters are grouped together **bottom-up**. Each level of the hierarchy can be interpreted as a clustering of different granularity and is visualized as a **dendrogram**.



rows represent genes,
columns represent samples

Image from http://genomicsclass.github.io/book/pages/clustering_and_heatmaps.html