

Statistisches Data Mining (StDM)

Woche 4

Oliver Dürr

Institut für Datenanalyse und Prozessdesign
Zürcher Hochschule für Angewandte Wissenschaften

oliver.duerr@zhaw.ch

Winterthur, 11 Oktober 2016

No laptops, no phones, no problems



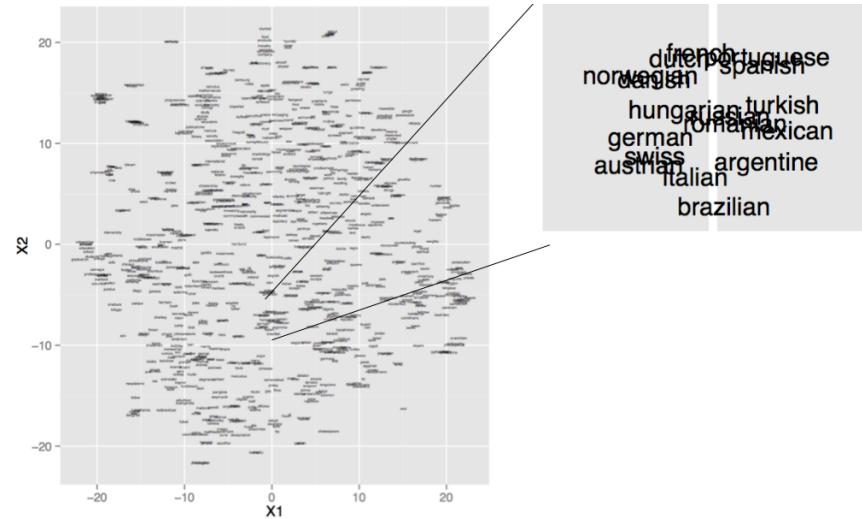
Multitasking senkt Lerneffizienz:

- **Keine Laptops im Theorie-Unterricht Deckel zu oder fast zu (Sleep modus)**

Overview of the semester

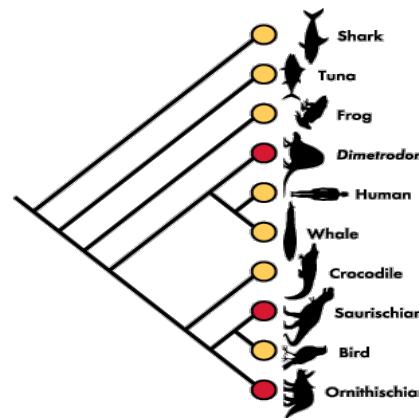
Part I (Unsupervised Learning)

- Dimension Reduction
 - PCA
- Similarities, Distance between objects
 - Euclidian, L-Norms, Gower,...
- Visualizing Similarities (in 2D)
 - MDS, t-SNE
- Clustering
 - K-Means
 - Hierarchical Clustering



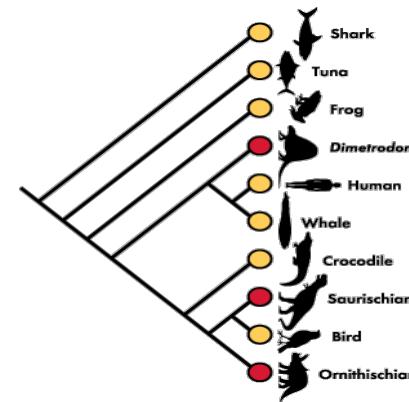
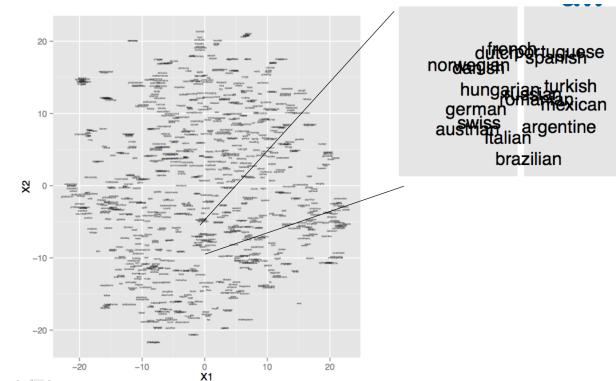
Part II (Supervised Learning)

- ...



Overview: Unsupervised learning

- Methods to visualize Data (dimension reduction in metric spaces)
 - PCA
- Distances
 - **Definition of distances**
 - **Euclidean and Minkowski Distance**
 - **Binary Data**
 - **Categorical Data**
 - **Mixed data types**
- Methods to visualize distance (in 2D)
 - Multidimensional Scaling (MDS)
 - Linear Metric MDS
 - Non-Linear Metric MDS
 - isoMDS
 - t-SNE
- Clustering approaches
 - Grouping of data
- Skript Andreas Ruckstuhl



Dissimilarities for Categorical and Mixed Data

Similarity measures between binary vectors

$$o_1 = (o_{11}, o_{12}, \dots, o_{1p}) \text{ and } o_2 = (o_{21}, o_{22}, \dots, o_{2p})$$

Common situation is that objects, o_1 and o_2 , have only binary attributes, like for example gender (f/M), driving license (yes/no), Nobel price holder (yes/no). We code them with 0 and 1.

We distinguish between symmetric and asymmetric binary variables.

In symmetric binary variable both levels have roughly comparable frequencies
example: gender

In asymmetric binary variable both levels have very different frequencies
many 0s few 1s.

Example: Nobel price holder = 1. It's nothing discriminative to be not a noble price winner. Therefore 0 don't count.

Matching Coefficient

Similarity measures for “symmetric” binary vectors

$$o_1 = (o_{11}, o_{12}, \dots, o_{1p}) \text{ and } o_2 = (o_{21}, o_{22}, \dots, o_{2p})$$

The objects, o_1 and o_2 , have only binary attributes

Compute for the **symmetric binary variable** (could be only a subset of the p binary variables) the **Simple Matching Coefficients**:

$$\text{SMC} = \# \text{ matches} / \# \text{ attributes}$$

Corresponding to the **proportion of matching features over all features**

$$\text{SMC} = (M_{11} + M_{00}) / (M_{01} + M_{10} + M_{11} + M_{00})$$

M_{01} = the number of attributes where o_{1i} was 0 and o_{2i} was 1

M_{10} = the number of attributes where o_{1i} was 1 and o_{2i} was 0

M_{00} = the number of attributes where o_{1i} was 0 and o_{2i} was 0

M_{11} = the number of attributes where o_{1i} was 1 and o_{2i} was 1

1 – SMC is a distance

Matching Coefficient

Similarity measures for “asymmetric” binary vectors

$$o_1 = (o_{11}, o_{12}, \dots, o_{1p}) \text{ and } o_2 = (o_{21}, o_{22}, \dots, o_{2p})$$

The objects, o_1 and o_2 , have only binary attributes

Compute for the [asymmetric binary variable](#) (could be only a subset of the p binary variables) the [Jaccard Coefficient](#):

$$J = \# \text{ both-1-matches} / \# \text{ of not-both-zero attributes values}$$

Corresponding to the proportion of matching features over [those features which are 1 in at least one of both observations](#). “0’s don’t count”

$$J = (M_{11}) / (M_{01} + M_{10} + M_{11})$$

M_{01} = the number of attributes where o_{1i} was 0 and o_{2i} was 1

M_{10} = the number of attributes where o_{1i} was 1 and o_{2i} was 0

M_{11} = the number of attributes where o_{1i} was 1 and o_{2i} was 1

1 – J is a distance (Jaccard Distance)

Example: How similar are two given binary vectors?

$p = 1 0 0 0 0 0 0 0 0 0$

$q = 0 0 0 0 0 1 0 0 1$

$M_{01} = 2$ (the number of attributes where p was 0 and q was 1)

$M_{10} = 1$ (the number of attributes where p was 1 and q was 0)

$M_{00} = 7$ (the number of attributes where p was 0 and q was 0)

$M_{11} = 0$ (the number of attributes where p was 1 and q was 1)

$$SMC = (M_{11} + M_{00}) / (M_{01} + M_{10} + M_{11} + M_{00})$$

$$SMC = (M_{11} + M_{00}) / (M_{01} + M_{10} + M_{11} + M_{00}) = (0+7) / (2+1+0+7) = 0.7$$

$$J = (M_{11}) / (M_{01} + M_{10} + M_{11})$$

$$J = (M_{11}) / (M_{01} + M_{10} + M_{11}) = 0 / (2 + 1 + 0) = 0$$

More than 2 levels (Nominal Data)

- Simple matching coefficient
- mm: Number of variables in which object i and j mismatch
- p number of features

$$d(i, j) = \frac{mm}{p}$$

Proportion of variables,
in which people disagree

- Character strings can be understood as nominal data.
- Also called Hamming-Distance if with strings of same length (sometimes w/o dividing by p)
- What is the Hamming-Distance between:
 - HOUSE
 - MOUSE

Gower's dissimilarity for mixed data types

Idea: Use distance measure d_{ij} between 0 and 1 for each variable or feature:

- k_{th} variable is binary, nominal:
Use discussed methods, e.g.

$$d_{ij}^{(k)} = \frac{M_{11} + M_{00}}{M_{01} + M_{10} + M_{11} + M_{00}}$$

- k_{th} variable is numeric:

x_{ik} : Value for object i in variable k
 R_k : Range of variable k for all objects

$$d_{ij}^{(k)} = \frac{|x_{ik} - x_{jk}|}{R_k}$$

- Ordinal: Use ranks, then same like with numeric variables

Aggregate distance measures over all variables/feature/dimensions:

$$d_{ij} = \frac{1}{p} \sum_{k=1}^p d_{ij}^{(k)}$$

Dissimilarity for mixed data types with R-function "daisy" calculating Gower's dissimilarity

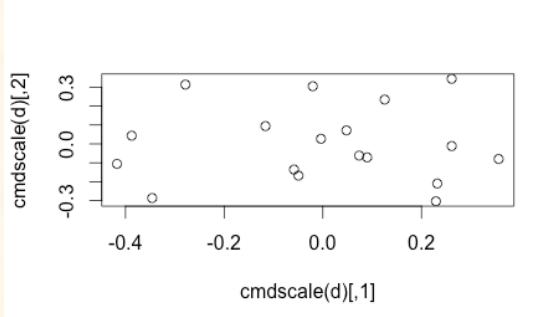
```
> str(flower)
```

```
'data.frame': 18 obs. of 8 variables:  
 $ V1: Factor w/ 2 levels "0","1": 1 2 1 1 1 1 1 1 1 2 2 ...  
 $ V2: Factor w/ 2 levels "0","1": 2 1 2 1 2 2 1 1 2 2 ...  
 $ V3: Factor w/ 2 levels "0","1": 2 1 1 2 1 1 1 2 1 1 ...  
 $ V4: Factor w/ 5 levels "1","2","3",...: 4 2 3 4 5 4 4 2 3 5 ...  
 $ V5: Ord.factor w/ 3 levels "1"<"2"<"3": 3 1 3 2 2 3 3 2 1 2 ...  
 $ V6: Ord.factor w/ 18 levels "1"<"2"<"3"<...: 15 3 1 16 2 12 ...  
 $ V7: num 25 150 150 125 20 50 40 100 25 100 ...  
 $ V8: num 15 50 50 50 15 40 20 15 15 60 ...
```

```
> dist=daisy(flower, type=list(asymm=c(1, 3), symm=2, ordratio=7))  
> str(dist)
```

```
Classes 'dissimilarity', 'dist' atomic [1:153] 0.901 0.618 ...  
 ..- attr(*, "Size")= int 18  
 ..- attr(*, "Metric")= chr "mixed"  
 ..- attr(*, "Types")= chr [1:8] "A" "S" "A" "N" ...
```

```
> plot(cmdscale(d))
```



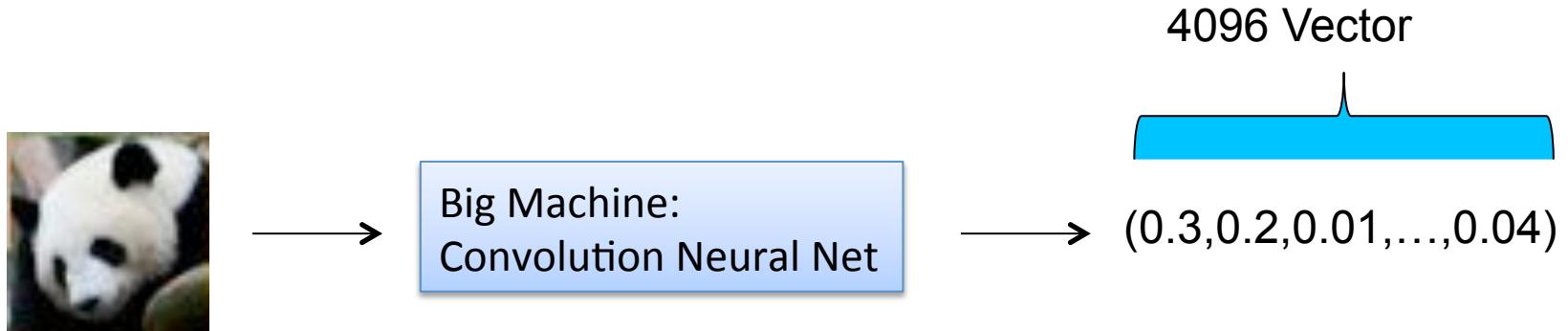
Special Distances

String Distances

- Package stringdist
 - Hamming Distance (see before)
 - Just count the number of mismatches
 - More distances
 - Longest common substring (Bioinformatic)
 - ...
- Many distances used in bioinformatics comparing DNAs
- Or use embeddings like word2vec
 - String (complicated procedure) → 100 (or so) dimensional vector
 - Calculate Euclidean Distance

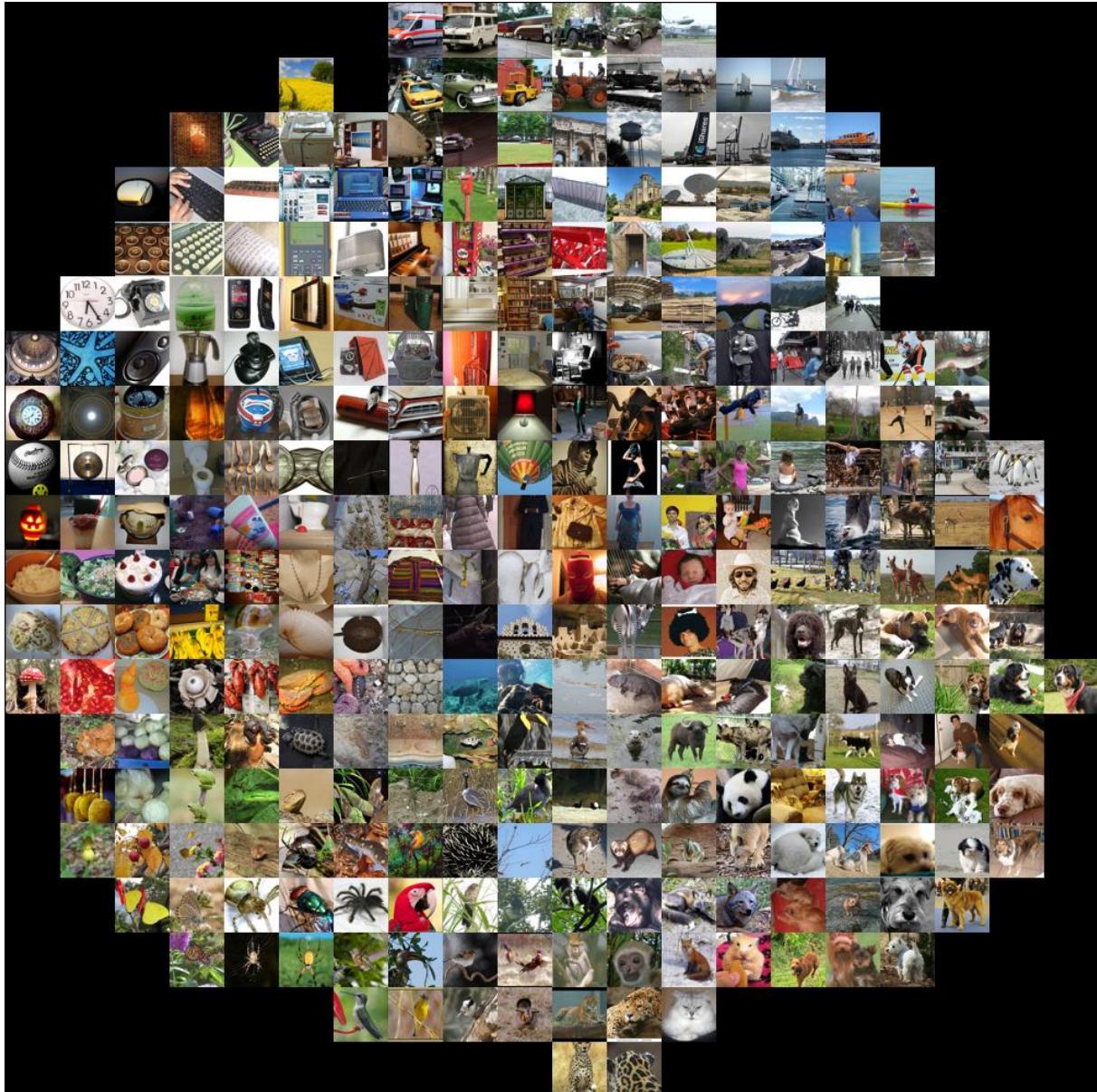
Distances between images [just for completeness]

- Dump approach use Euclidean distance between pixel values directly.
- Current state of the art, for image similarities: convolutional neural network



See: <http://cs.stanford.edu/people/karpathy/cnnembed/>

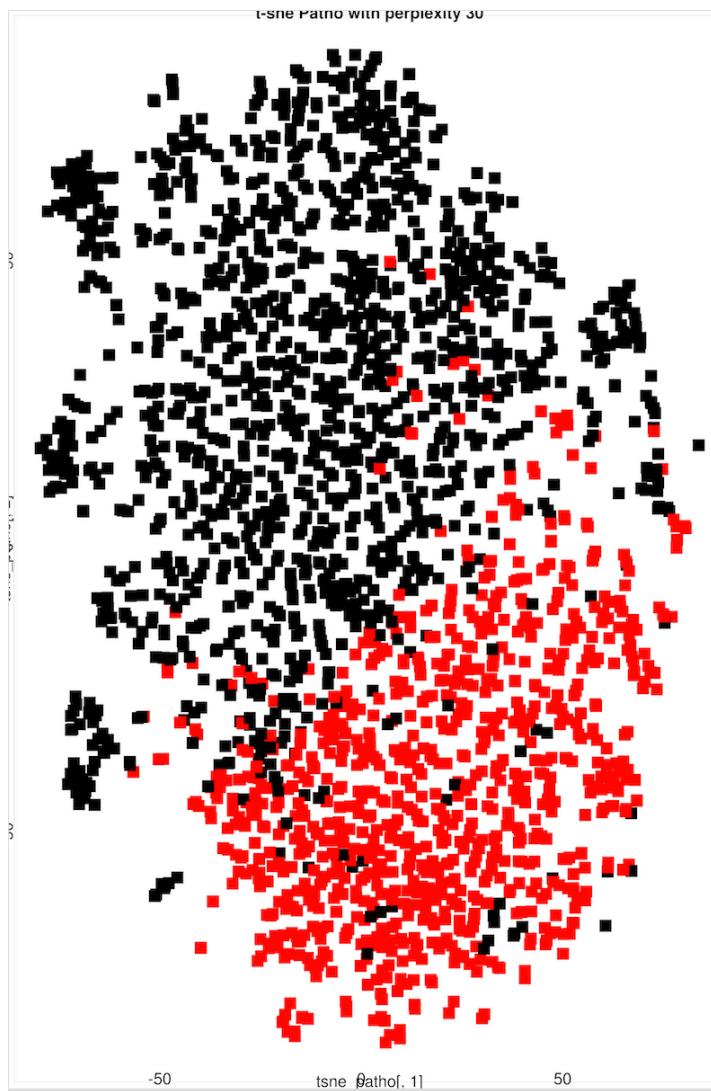
S
9



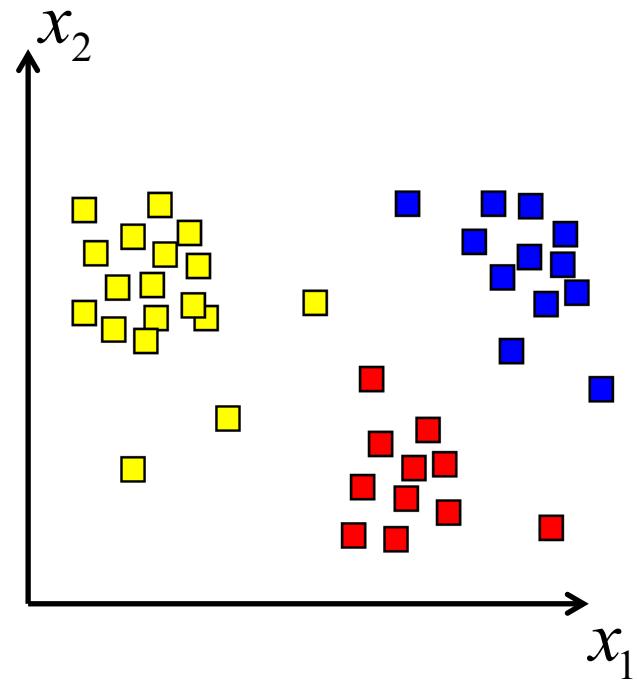
**Semantic
similarities not
pixel wise!**

See: <http://cs.stanford.edu/people/karpathy/cnnembed/> for more images

Example Project with Uni-Spital ZH



Clustering



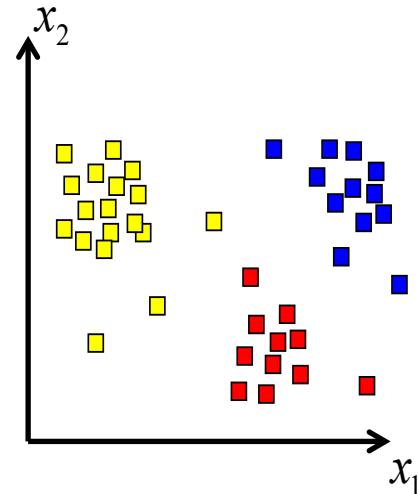
Now again in line with ILSR

- Inline again with ILSR
- See section 10.3 Clustering Methods in ILSR
- Aims:
 - PCA (and other dimension reduction methods) look to find a low-dimensional representation of the observations
 - Clustering looks to find homogeneous subgroups among the observations.
- Examples of applications
 - Personalized medicine
 - Segment into subgroups needing different medication
 - Market segmentation
 - ...

Descriptive and unsupervised: Cluster Analysis

Cluster analysis or clustering is the task of **assigning** a set of objects **into groups**.

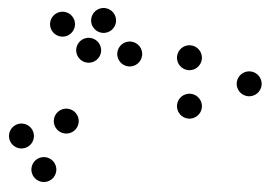
Objects in the **same cluster** should be **more similar** to each other than to those in other clusters.



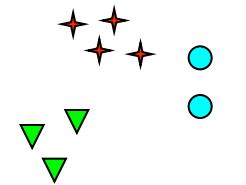
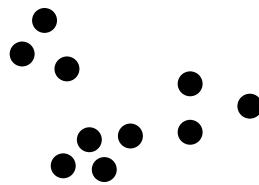
To perform clustering one must define a **measure of similarity or distance based on the observed values** describing different properties of the objects.

$$e.g. \text{ euclidean} : dist(o_k, o_l) = \sum_{i=1}^p (x_{ki} - x_{li})^2$$

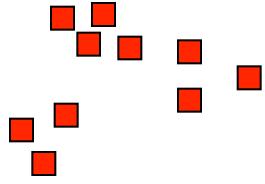
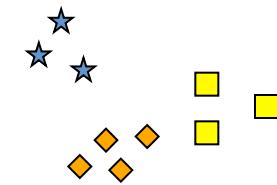
Notion of a Cluster can be Ambiguous



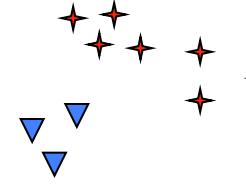
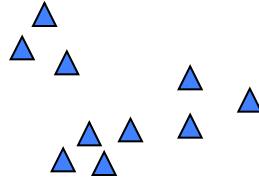
How many clusters?



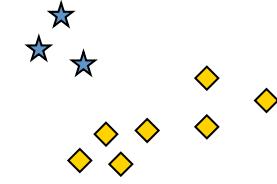
Six Clusters



Two Clusters

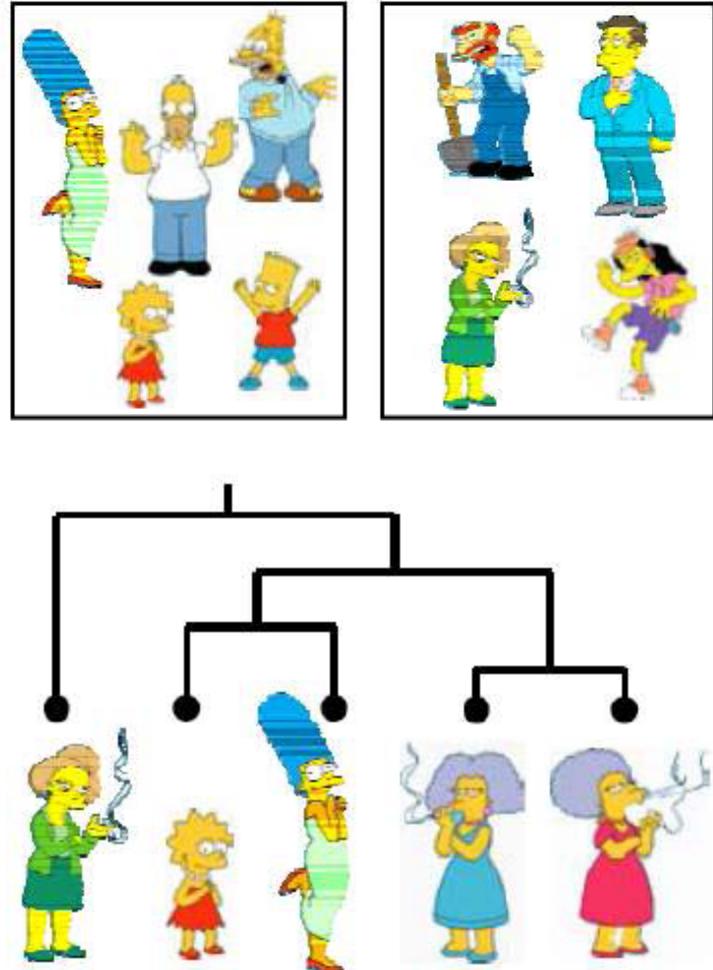


Four Clusters



Types of Clustering

- **Important distinction between hierarchical and partitional sets of clusters**
- **Partitional Clustering**
A division data objects into non-overlapping subsets (clusters) such that each data object is in exactly one subset
- **Hierarchical clustering**
A set of nested clusters organized as a hierarchical tree



Partitional Clustering

What is optimized in K-means Clustering ?

The goal in k-means is to partition the observations into K clusters such that the total within-cluster variation (WCV), summed over all K clusters C_k , is as small as possible.

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

WCV is often based on Euclidian distances

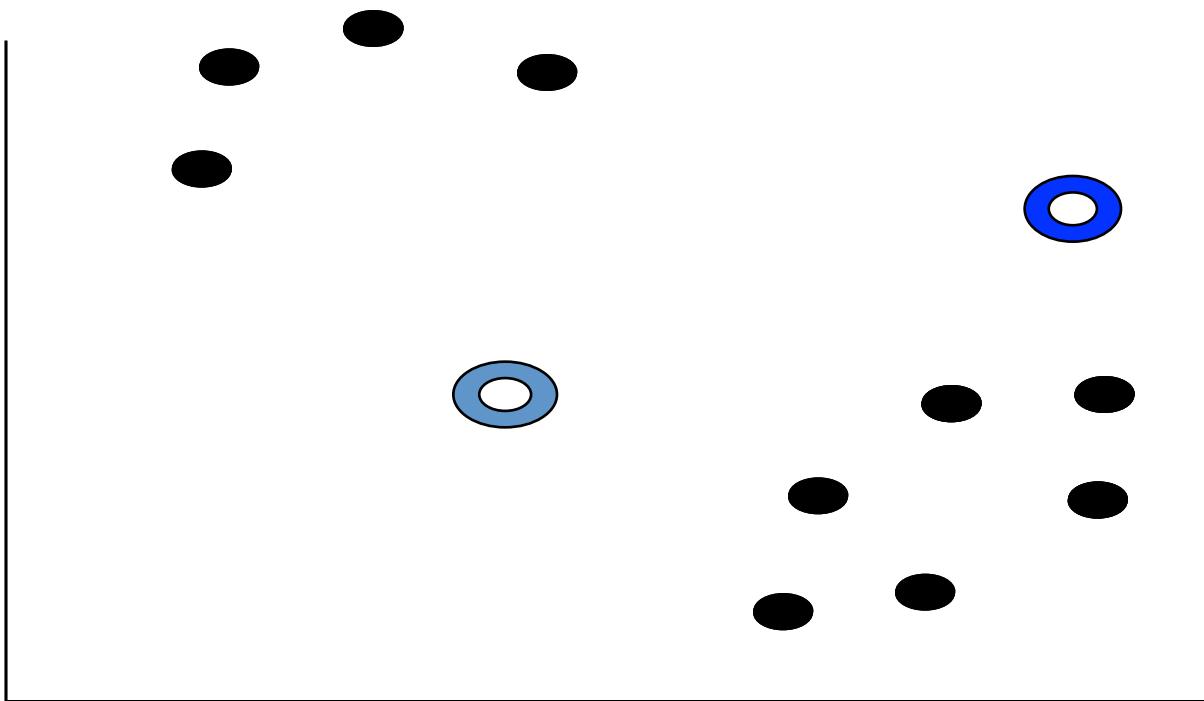
$$\text{WCV}(C_k) = \frac{1}{|C_k|} \sum_{i, i' \in C_k} \sum_{j=1}^p (x_{ij} - x_{i'j})^2$$

Squared Euclidian distance between data points i and i'

where $|C_k|$ denotes the number of observations in the k th cluster and p is the number of variables (dimensions).

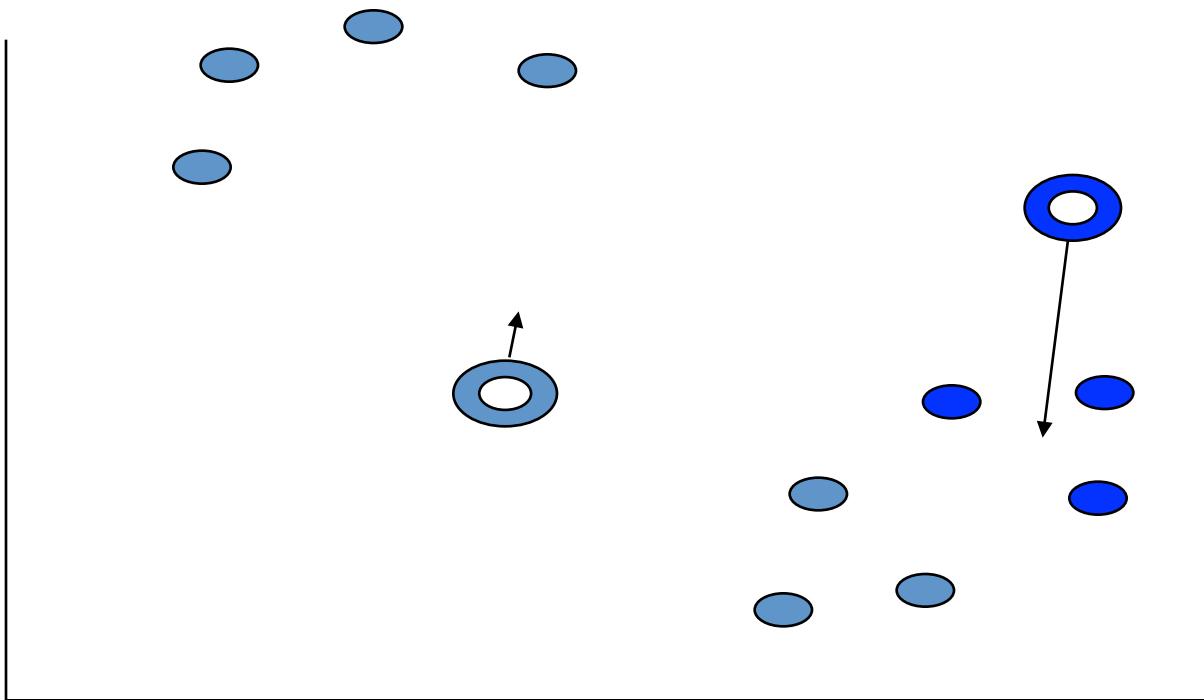
Partitioning Clustering: K-means

- + Given a number of objects and an initial (randomly chosen) set of cluster centers, assign each object to the closest cluster center



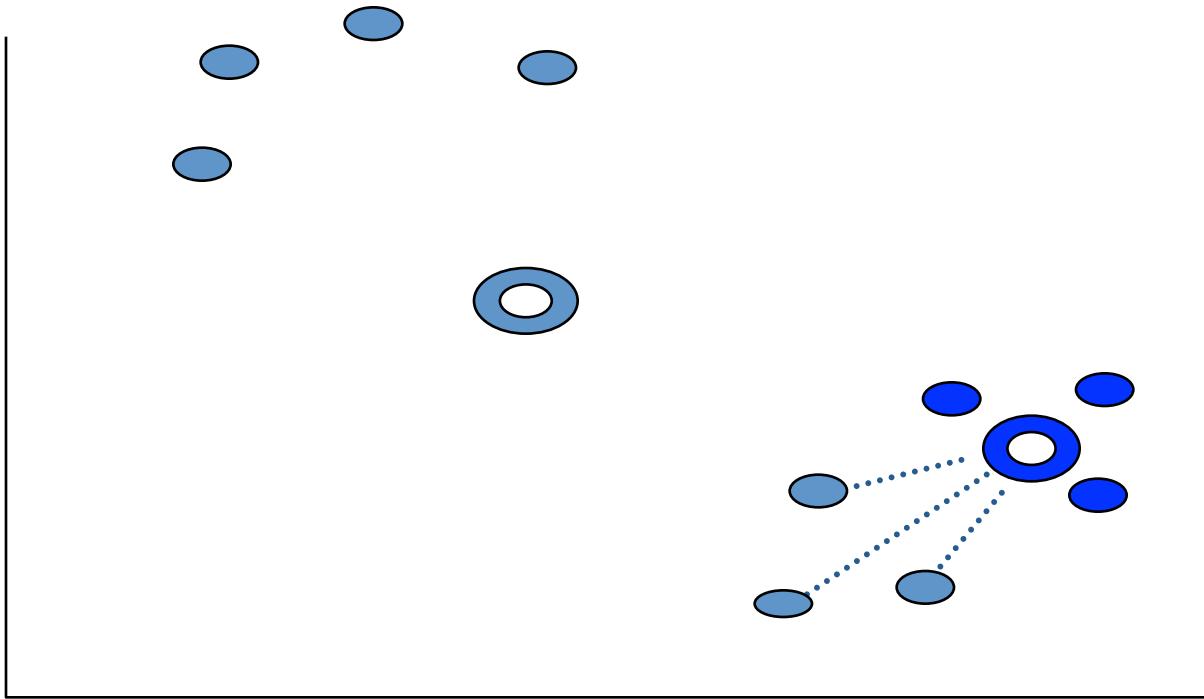
Partitioning Clustering: K-means

- + Update the coordinates of each cluster center to the average coordinate of the objects associated with it



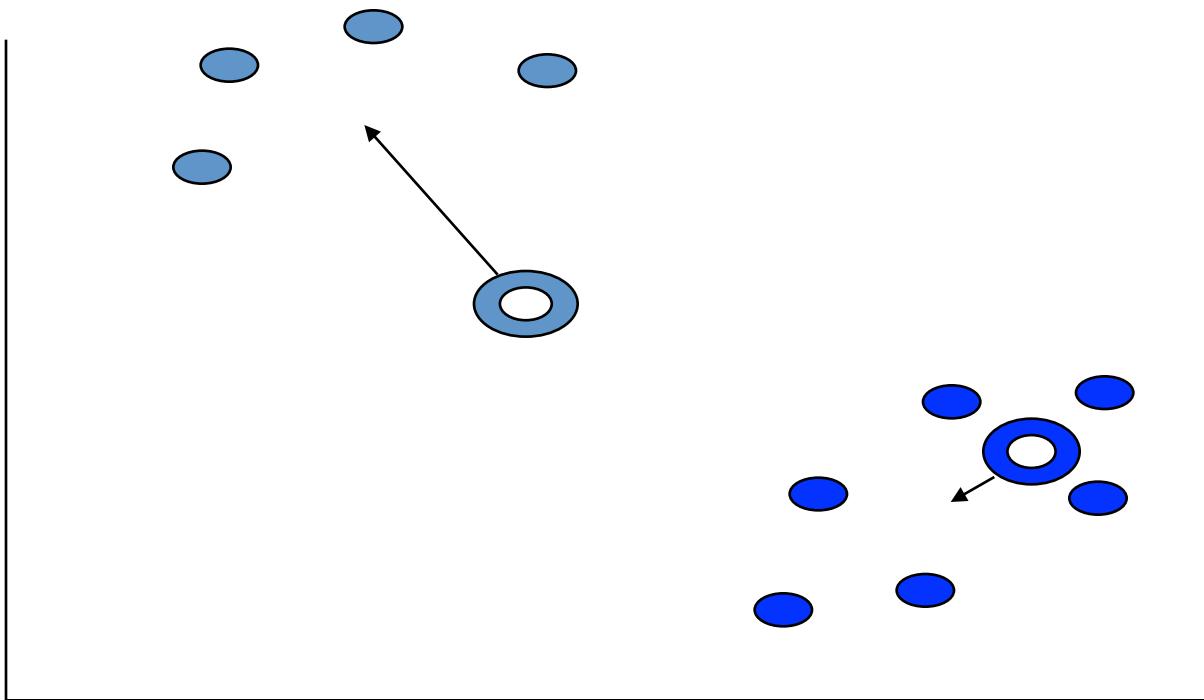
Partitioning Clustering: K-means

- + Re-assign each gene to the closest cluster centre



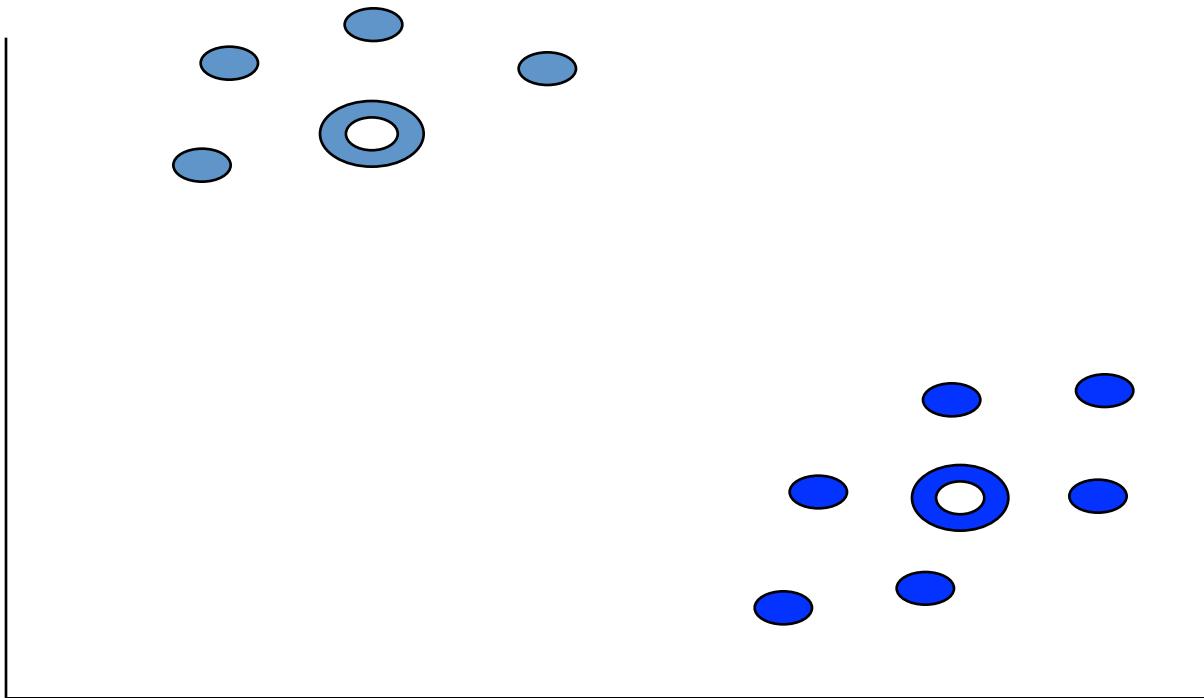
Partitioning Clustering: K-means

- + Recalculate the co-ordinates of each cluster centre according to the average co-ordinate of the genes associated with it



Partitioning Clustering: K-means

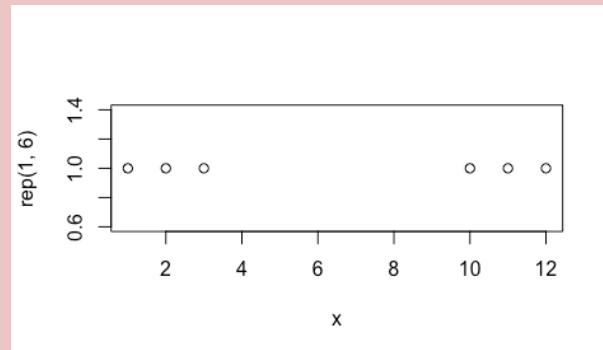
- + Repeat these steps until no reassignment is possible (or maximal number of iterations have been reached).



Clustern von Hand

Find a sensible k=2 clustering for the data-points

$x = c(1, 2, 3, 10, 11, 12)$

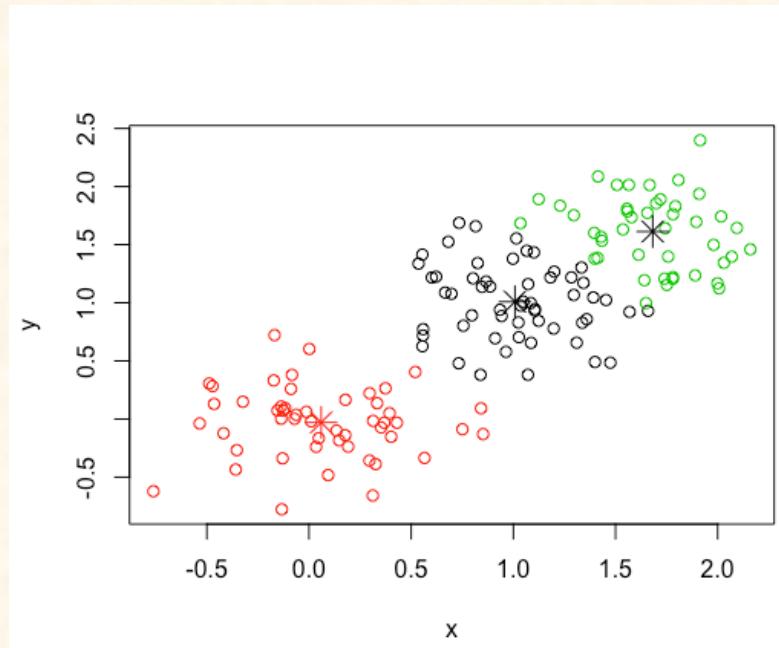


And calculate the within cluster variations

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

K-means clustering in R

```
x <- rbind(matrix(rnorm(100, sd = 0.3), ncol = 2),  
           matrix(rnorm(100, mean = 1, sd = 0.3), ncol = 2),  
           matrix(rnorm(100, mean = 1.6, sd = 0.3), ncol = 2)  
         )  
colnames(x) <- c("x", "y")  
(cl <- kmeans(x, 3))  
plot(x, col = cl$cluster)  
points(cl$centers, col = 1:2, pch = 8, cex = 2)
```



K-Means

- The k-means algorithm (see also animation) is a optimization of the total within cluster variation.

Algorithm 10.1 K-Means Clustering

1. Randomly assign a number, from 1 to K , to each of the observations. These serve as initial cluster assignments for the observations.
2. Iterate until the cluster assignments stop changing:
 - (a) For each of the K clusters, compute the cluster *centroid*. The k th cluster centroid is the vector of the p feature means for the observations in the k th cluster.
 - (b) Assign each observation to the cluster whose centroid is closest (where *closest* is defined using Euclidean distance).

-
- **Can be trapped in a local minima**, not a global optimization of WCV

How to choose the “best” number of clusters in K-means ?

- Run k-Means for several k - number of groups
- Determine minimized sum of WCV `sum(kmeans(dat, centers=k)$withinss)`
- Plot minimized sum of WCV vs. number of groups (k)
- Choose number of groups after the last big drop of

