Statistisches Data Mining (StDM)
Woche 2

Zurich University of Applied Sciences

- *Oliver Dürr*
- Institut für Datenanalyse und Prozessdesign
- Zürcher Hochschule für Angewandte Wissenschaften

- oliver.duerr@zhaw.ch

- Winterthur, 27 September 2016

No laptops, no phones, no problems

Multitasking senkt Lerneffizienz:

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

Bewertung

- Es gibt eine freiwillige Zwischenprüfung. Die Prüfung dauert 45-60 Minuten und ergibt eine erste Vornote, die zu 15% in Endnote zählt, falls sie besser ist als die Note der Schlussprüfung.
- Die freiwillige Bearbeitung einer Hausarbeit ergibt eine zweite Vornote, die zu 10% in Endnote zählt, falls sie besser ist als die Note der Schlussprüfung
- Die Endprüfung ist obligatorisch und dauert 90 Minuten.
- Die Modulendnote ist der grössere Wert von
  [0.1x HA + 0.15 x Zwischenprüfung + 0.75 x Endprüfung] (*) und
  [0.15 x Zwischenprüfung + 0.85 x Endprüfung] (*) und
  [0.1 x HA + 0.9 x Endprüfung] (*) und
  [Endprüfung] (*).

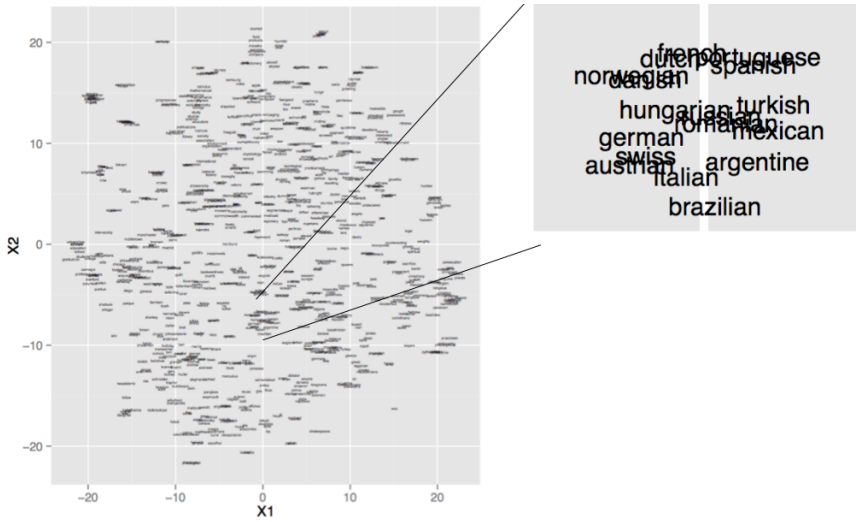
Die Vorleistungen (Zwischenprüfung, Praktika) sind insofern fakultative, als sie nur dann zählt, wenn sie die Endnote verbessert. Es wird keine Nachprüfungen für die Vorleistungen geben. Die Endprüfung kann bei begründetem Fernbleiben (Krankheit mit Arztzeugnis, Militärdienst, etc.) nachgeholt werden.

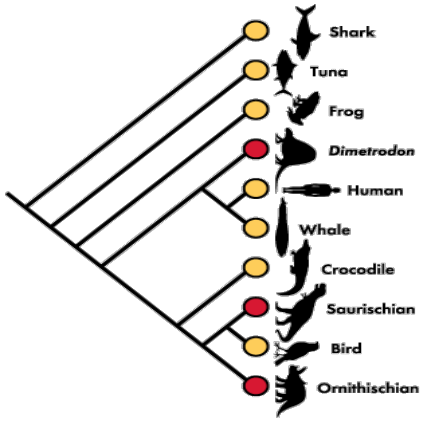
ZP in der Woche 8 (8 November)

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





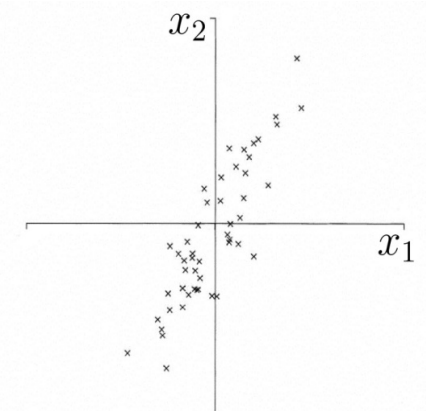
Part II (Supervised Learning)

- …

PCA Recap

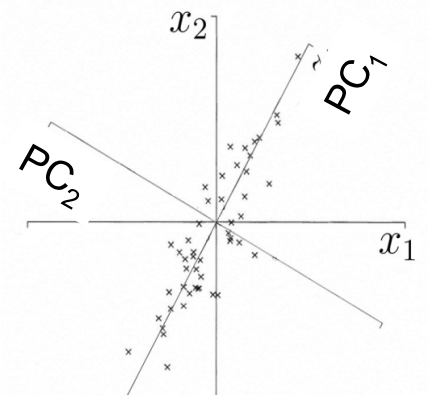
Too many features: PCA 2D → 1D

2D



PCA (rotation)

1D



Rotation (x_1,x_2) → (PC_1,PC_2) and dropping PC_2

2D

Example	X_1	X_2
1	5.1	3.5
2	4.9	3
3	3.3	3.2
4	5.1	3.5
…	…	…
150	4.9	3

PCA (rotation)

1D

Example	PC_1	PC_2
1	4.1	3.5
2	4.9	3
3	3.3	3.2
4	5.1	3.5
…	…	…
150	4.9	3

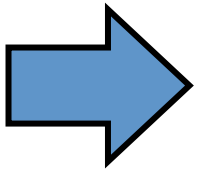
Too many features: PCA 3000D → 10D

30000 D

10D

PCA (rotation)

Not possible to draw



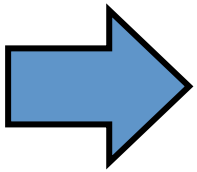
Not possible to draw

Rotation and dropping $(x_1,x_2,x_{30000}) \rightarrow (PC_1,PC_2,...,PC_{10})$

30000 D

Example	X1	X2		X30'0000
1	5.1	3.5	...	6
2	4.9	3	...	7.3
3	3.3	3.2	...	8.9
...	...	...		
150	4.9	3		0.3

PCA (rotation)

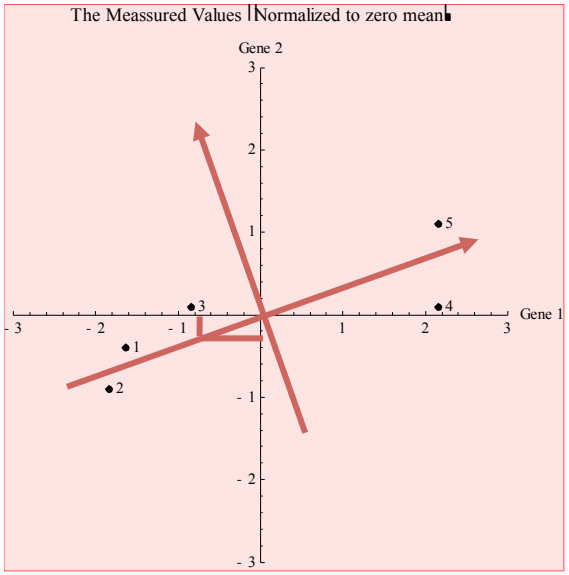


10D

Example	PC1	PC2		PC30'000
1	1.1	2.5	...	0.5
2	4.9	3	...	3
3	3.3	3.2	...	3.2
4	5.1	3.5	...	3.5
...	...	...		...
150	4.9	3		3

PCA as Eigenvalues

```
> co = t(X) %*% X
> co
       X_1  X_2
X_1 16.112 4.82
X_2  4.820 2.20
> eig = eigen(co)
> U = eig$vectors
> X %*% U
           [,1]       [,2]
[1,]  1.6846468 -0.1075419
[2,]  2.0247206  0.3100102
[3,]  0.7719022 -0.3460737
[4,] -2.0914512  0.5490283
[5,] -2.3898185 -0.4054229
```




```
> res = prcomp(X, scale. = FALSE)
> res$x
             PC1        PC2
[1,] -1.6846468  0.1075419
[2,] -2.0247206 -0.3100102
[3,] -0.7719022  0.3460737
[4,]  2.0914512 -0.5490283
[5,]  2.3898185  0.4054229
```

Without calculation XTX

```
> U1 = svd(X)$v
> X %*% U
           [,1]       [,2]
[1,]  1.6846468 -0.1075419
[2,]  2.0247206  0.3100102
[3,]  0.7719022 -0.3460737
[4,] -2.0914512  0.5490283
[5,] -2.3898185 -0.4054229
```

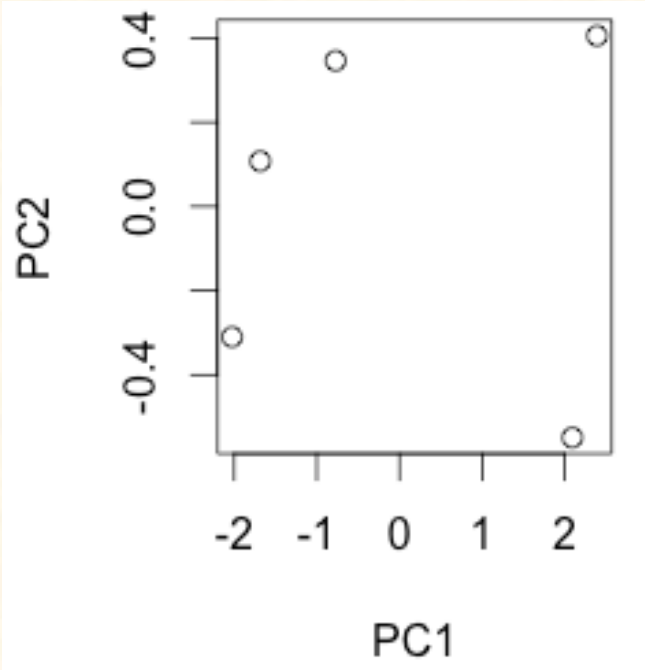
PCA as Eigenvalues

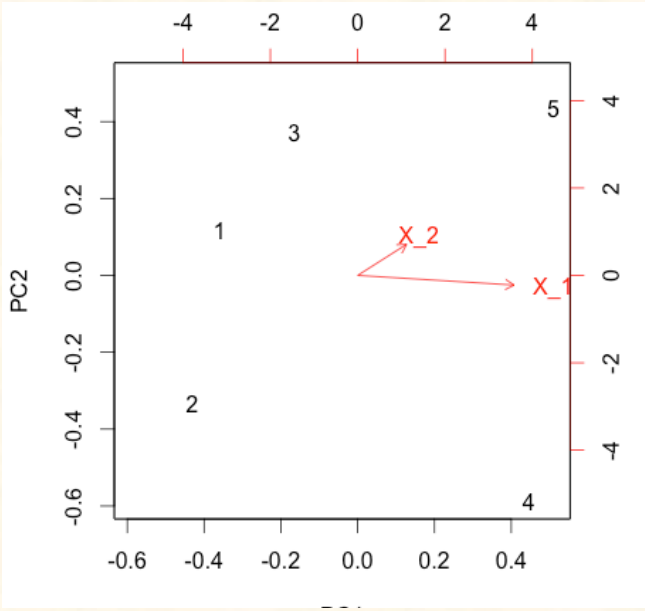
```
> res = prcomp(X, scale. = FALSE)
> Z = res$x          # The PCA-Transformed values
> res$sdev^2
[1] 4.4046904 0.1733096
> cov(X)
      X_1   X_2
X_1 4.028 1.205
X_2 1.205 0.550
> cov(Z)
              PC1           PC2
PC1 4.404690e+00 6.911247e-16
PC2 6.911247e-16 1.733096e-01
> cov(X)[1,1] + cov(X)[2,2]
[1] 4.578
> cov(Z)[1,1] + cov(Z)[2,2]
[1] 4.578
```

- After the PCA the covariance matrix is diagonal
- In the squared diagonal are the explained variances

PCA in R

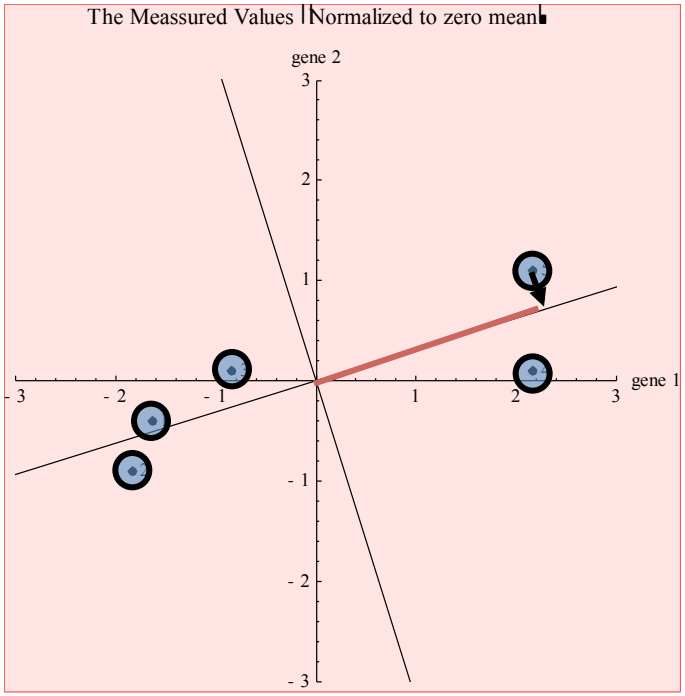
```
> res = prcomp(X, scale. = FALSE)
>   plot(res$x)
>   biplot(res, scale = TRUE)
>   res$rotation #Loadings
           PC1        PC2
X_1 0.9544511 -0.2983673
X_2 0.2983673  0.9544511
>   summary(res)
Importance of components:
                          PC1     PC2
Standard deviation     2.0987 0.41630
Proportion of Variance 0.9621 0.03786
Cumulative Proportion  0.9621 1.00000
```





Explained Variance definition and Example

Example Data Set 1



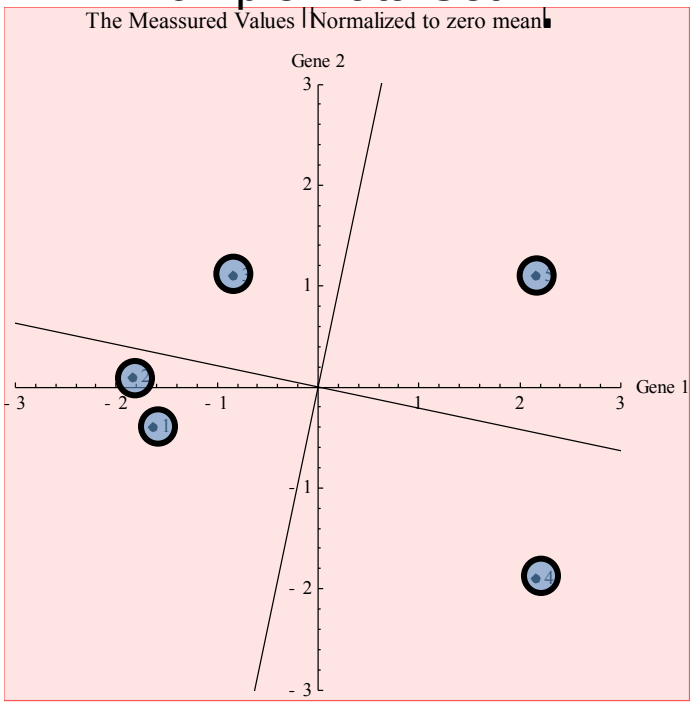
Variance:Sum of squares of all " ": —

{4.40469, 0.17331}

Explained variance percentage of total
96%=4.40 / (4.40+0.17), **4%.**

First component already
explains data to a great deal.

Example Data Set 2



Variance:

4.14257, 1.43543

Explained variance:
74%, 26%.

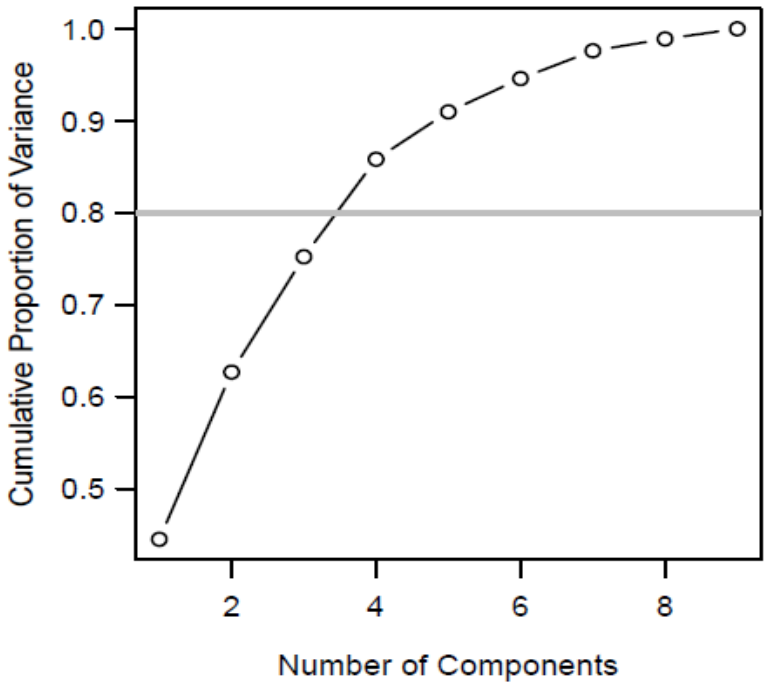
First component alone might
not be sufficient to explain the data.

How many PCs do we need? First criterion

The total variance can be calculated as:
(the total variance is preserved under rotation)

$$V_{total} = \sum_{j=1}^{p} \mathrm{var}(X_j) = \sum_{j=1}^{p} \mathrm{var}(Y_j) = \sum_{j=1}^{p} \lambda_j$$

Rule of thumb: ~80% of Var_{total} should be explained by the first k PCs

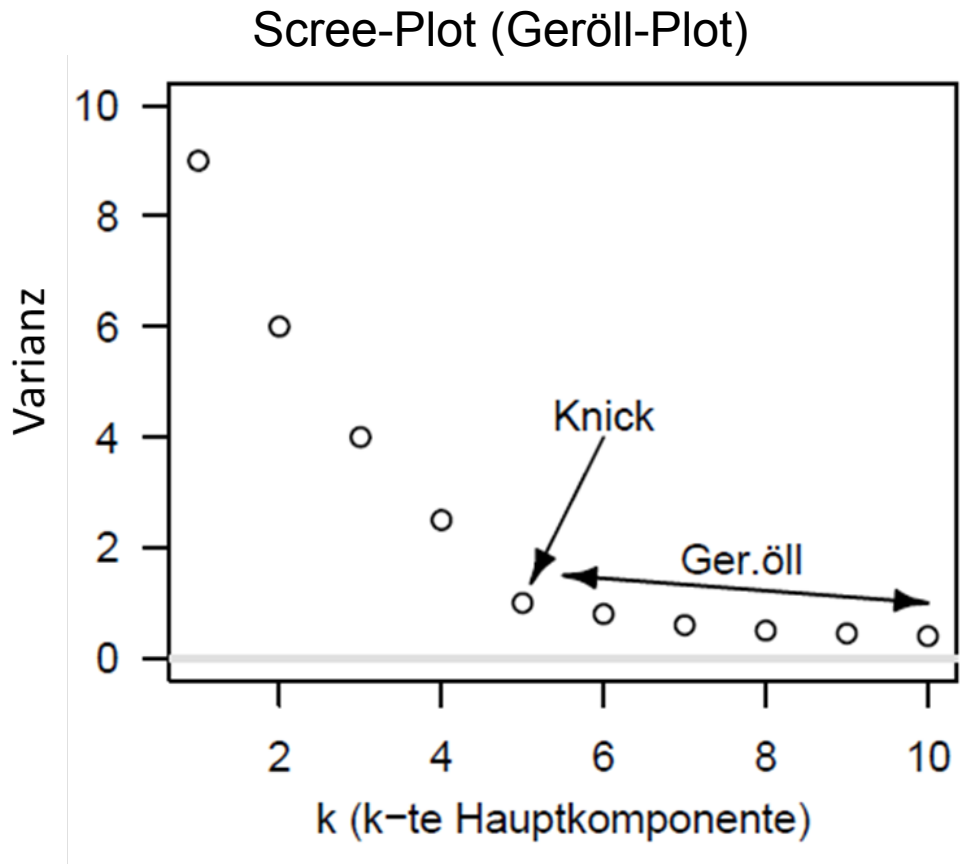


Gütekriterium der Approximation durch die ersten k Hauptkomponenten:

$$P_k = \frac{\sum_{j=1}^{k} \mathrm{var}(Y_j)}{V_{total}} \in [0,1]$$

How many PCs do we need? Second criterion

The position of the bend in the scree-plot indicates how many PCs are needed.
After the bend in the scree-plot we do not gain much when adding more PCs.



End of Recap

Scaling

Standardisieren, wenn die beobachteten Messgrössen in verschiedenen Einheiten (cm, m, kg…) vorliegen. Auch bei sehr grossen Unterschieden in den Varianzen der Variablen sollte man über deren Ursache nachdenken und ggf. auch dann standardisieren, wenn alle Variablen dieselbe Einheit haben.

Nicht standardisieren, wenn die gemessenen Variablen vergleichbar sind bezüglich ihrer Einheiten und ihrer Variabilität. Im Zweifelsfall skalieren.

```
prcomp(x, scale.=FALSE) #Default
```

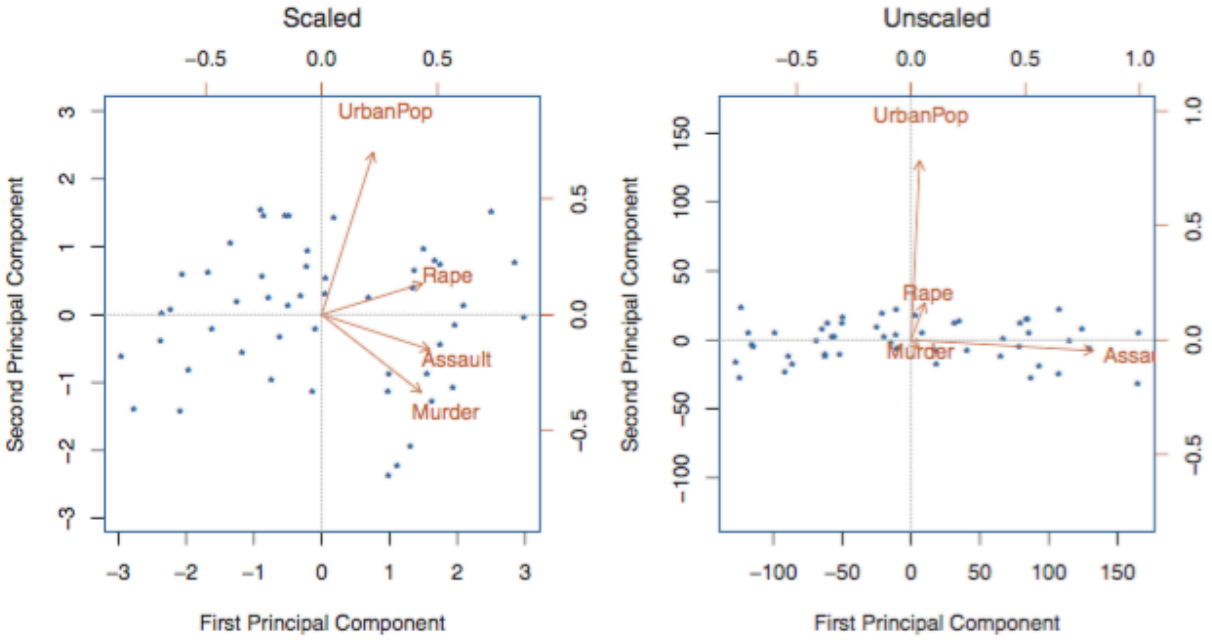
Example

```
> head(X)
           Murder Assault UrbanPop Rape
Alabama      13.2     236       58 21.2
Alaska       10.0     263       48 44.5
Arizona       8.1     294       80 31.0
Arkansas      8.8     190       50 19.5
California    9.0     276       91 40.6
Colorado      7.9     204       78 38.7
```

A data frame with 50 observations on 4 variables.

[,1] Murder	numeric Murder arrests (per 100,000)
[,2] Assault	numeric Assault arrests (per 100,000)
[,3] UrbanPop	numeric Percent urban population
[,4] Rape	numeric Rape arrests (per 100,000)

Would you scale?



Second Interpretation of the PCA

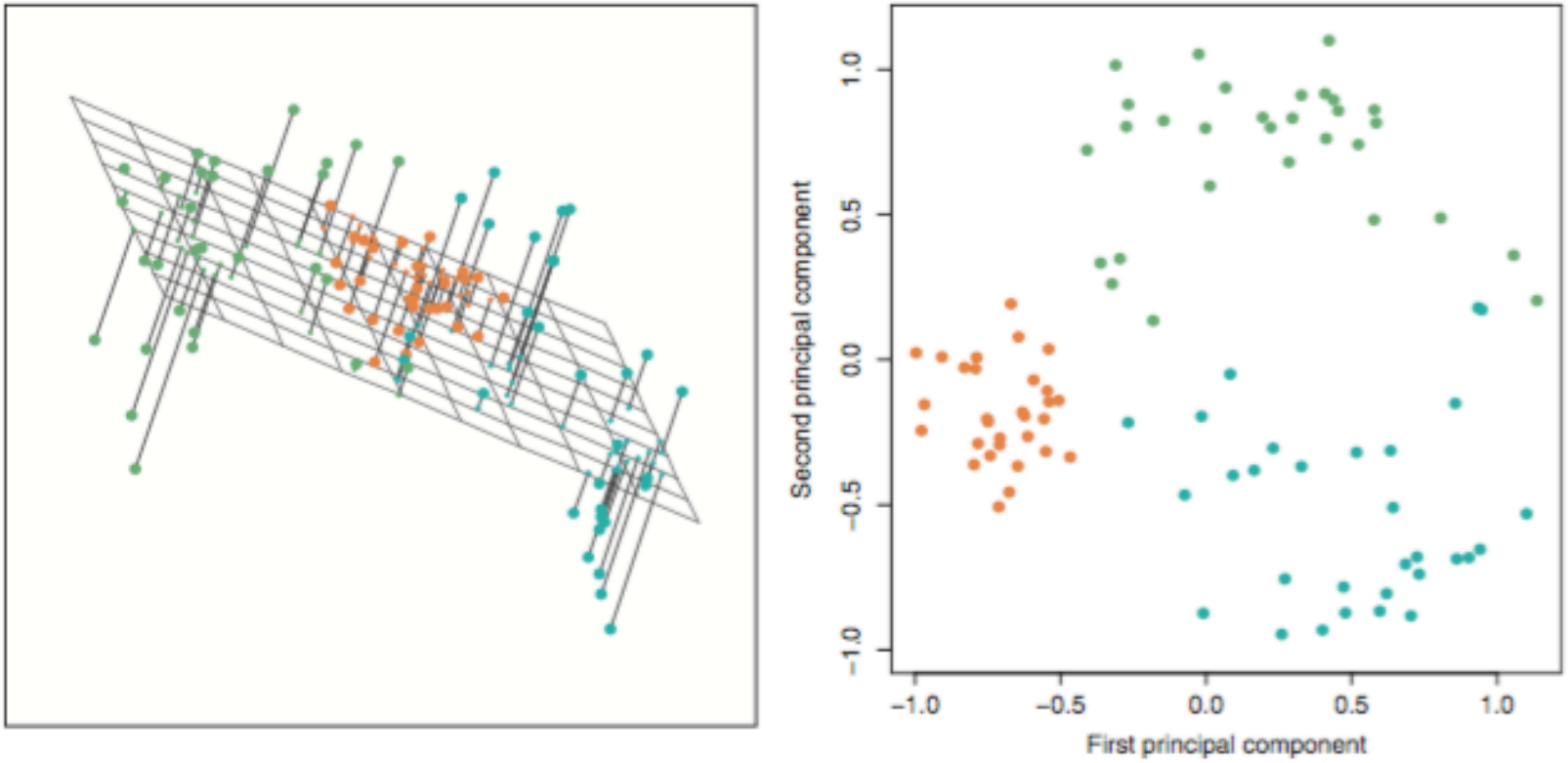


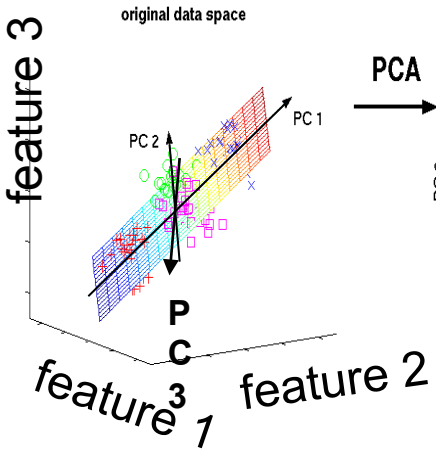
FIGURE 10.2. *Ninety observations simulated in three dimensions.* Left: *the first two principal component directions span the plane that best fits the data. It minimizes the sum of squared distances from each point to the plane.* Right: *the first two principal component score vectors give the coordinates of the projection of the 90 observations onto the plane. The variance in the plane is maximized.*

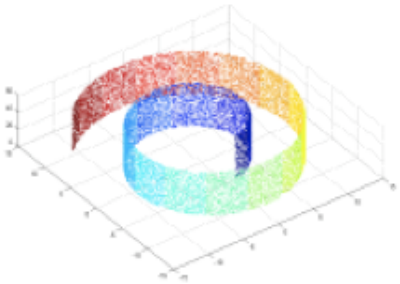
Beispiele für PCA

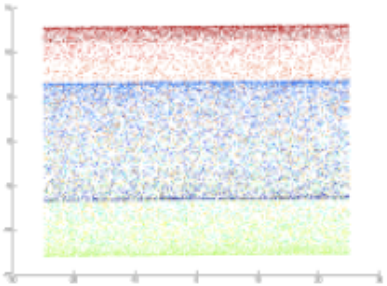
- **Ein paar Beispiele mit Ausreissern and die Tafel;**

Problems with PCA / metric MDS
The swiss roll

There is (almost) no reason, why the data should lie on a plan.





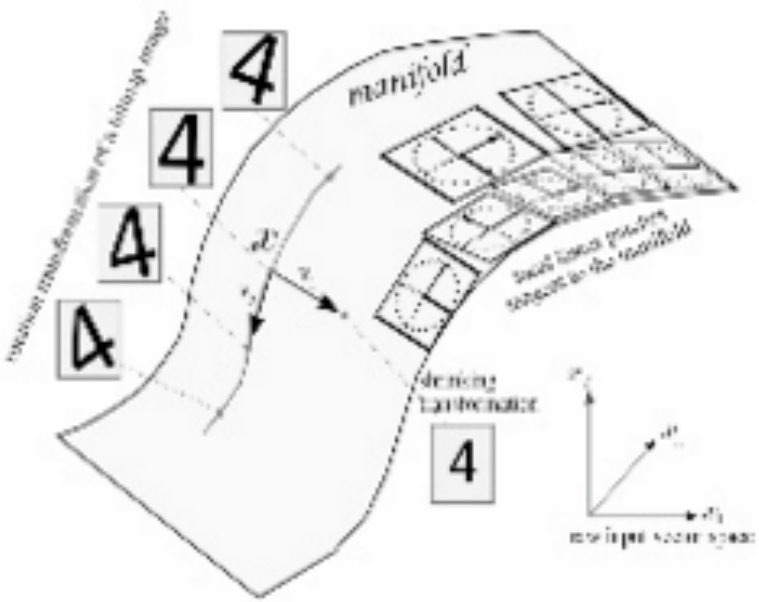


Goal: Preserve local structure. Keep local distances intact.

van der Maaten *et al.* 2007

Manifold hypothesis

- X high dimensional vector
- Data is concentrated around a low dimensional manifold



- Hope finding a representation Z of that manifold.

credit: http://www.deeplearningbook.org/

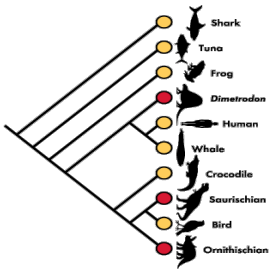
PCA: Variants (just for reference)

- A huge number of variants of PCA exists and is available in R packages, for example:
- **Robust PCA:** make PCA less sensitive to outliers, for example by using a robust estimate of the covariance matrix (PcaCov() in rrcov) or by other means like using Projection Pursuit (pcaPP)
- **Constrained PCA:** PCA-like transformation with some constraints on sparsity (constructing linear combinations from only a small number of original variables) and / or non-negativity of principal components (nsprcomp, elasticnet)
- **Kernel PCA:** By use of the so-called kernel trick, PCA can be extended by implicitly transforming the data to a high-dimensional space. Can also cope with non-numerical data like graphs, texts etc. R implementation e.g. as kpca() in kernlab.
- **Factor Analysis** is related to PCA. Focus is on interpretable transformations, often used in social sciences and psychology. Factors are often viewed as latent unobservable variables that influence outcomes of measurements
- For more variants implemented in R, see the CRAN task view „Multivariate“: https://cran.r-project.org/web/views/Multivariate.html

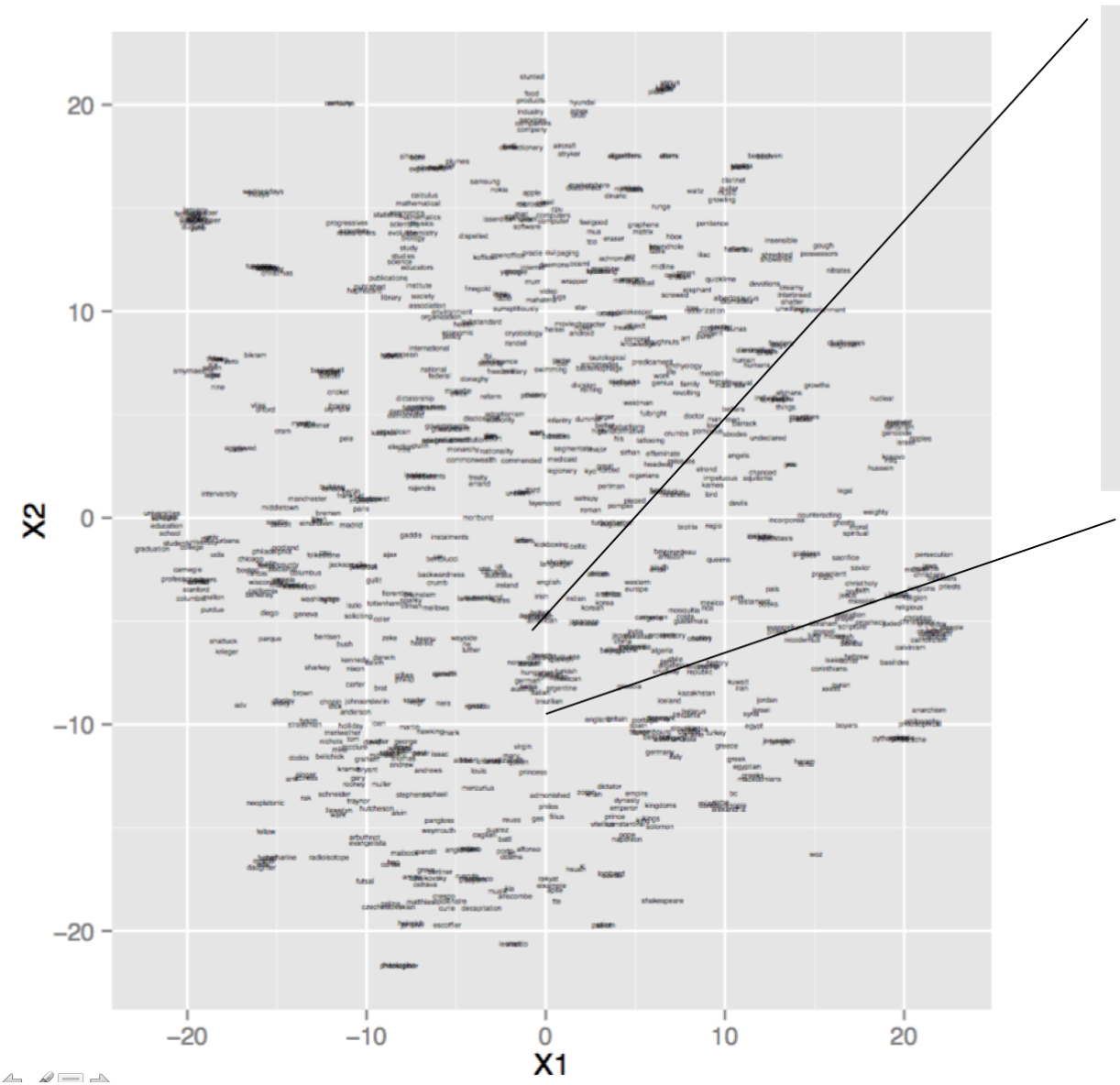
Visualizing Similarities

Overview: Unsupervised learning

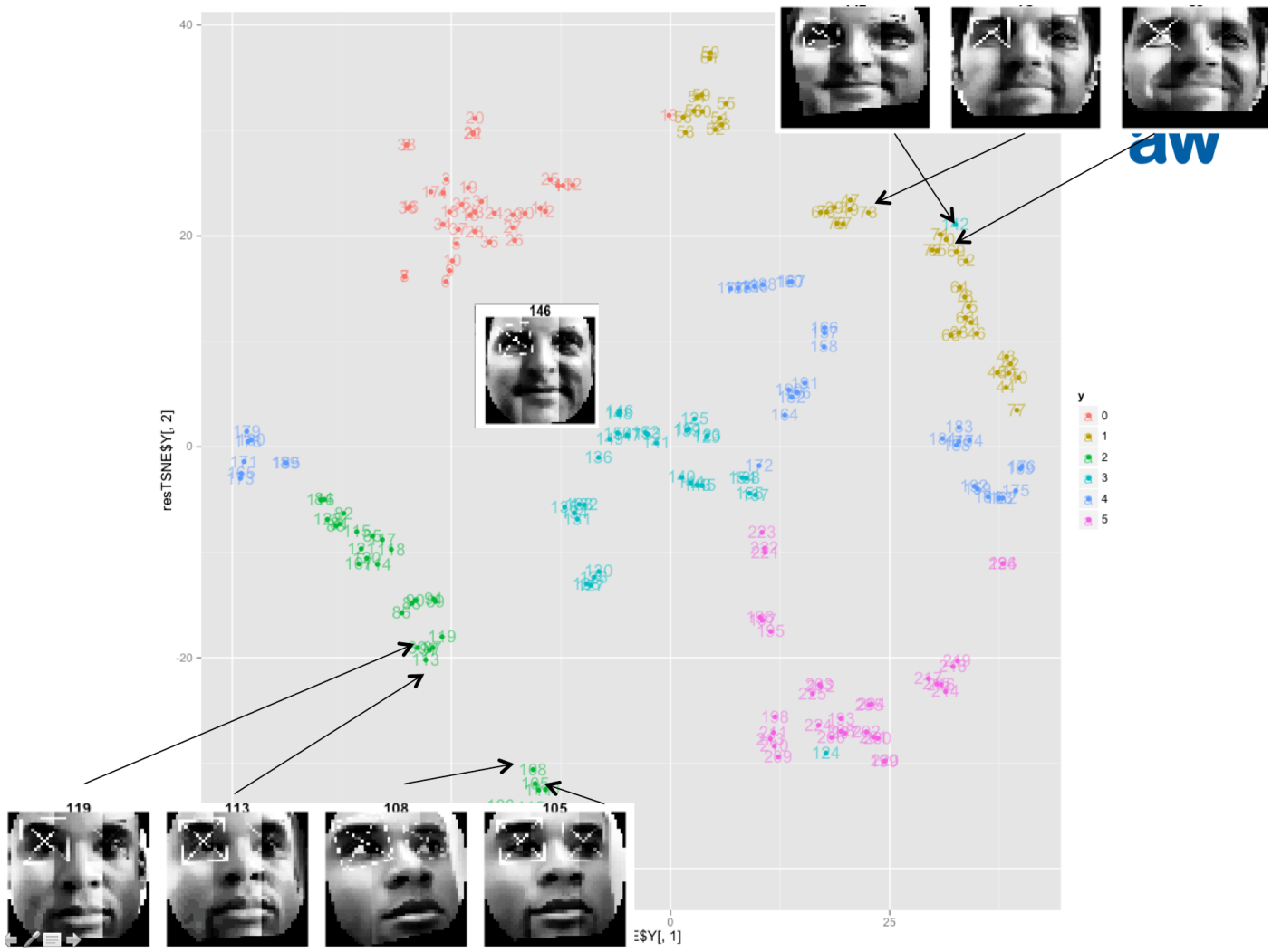
- Methods to visualize Data (dimension reduction in metric rooms)
  - PCA
- Distances
  - **Definition of distances**
  - **Euclidean and Minkowski Distance**
  - Binary Data
  - Categorical Data
  - Mixed data types
- Methods to visualize distance (in 2D)
  - Multidimensional Scaling (MDS)
  - Linear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



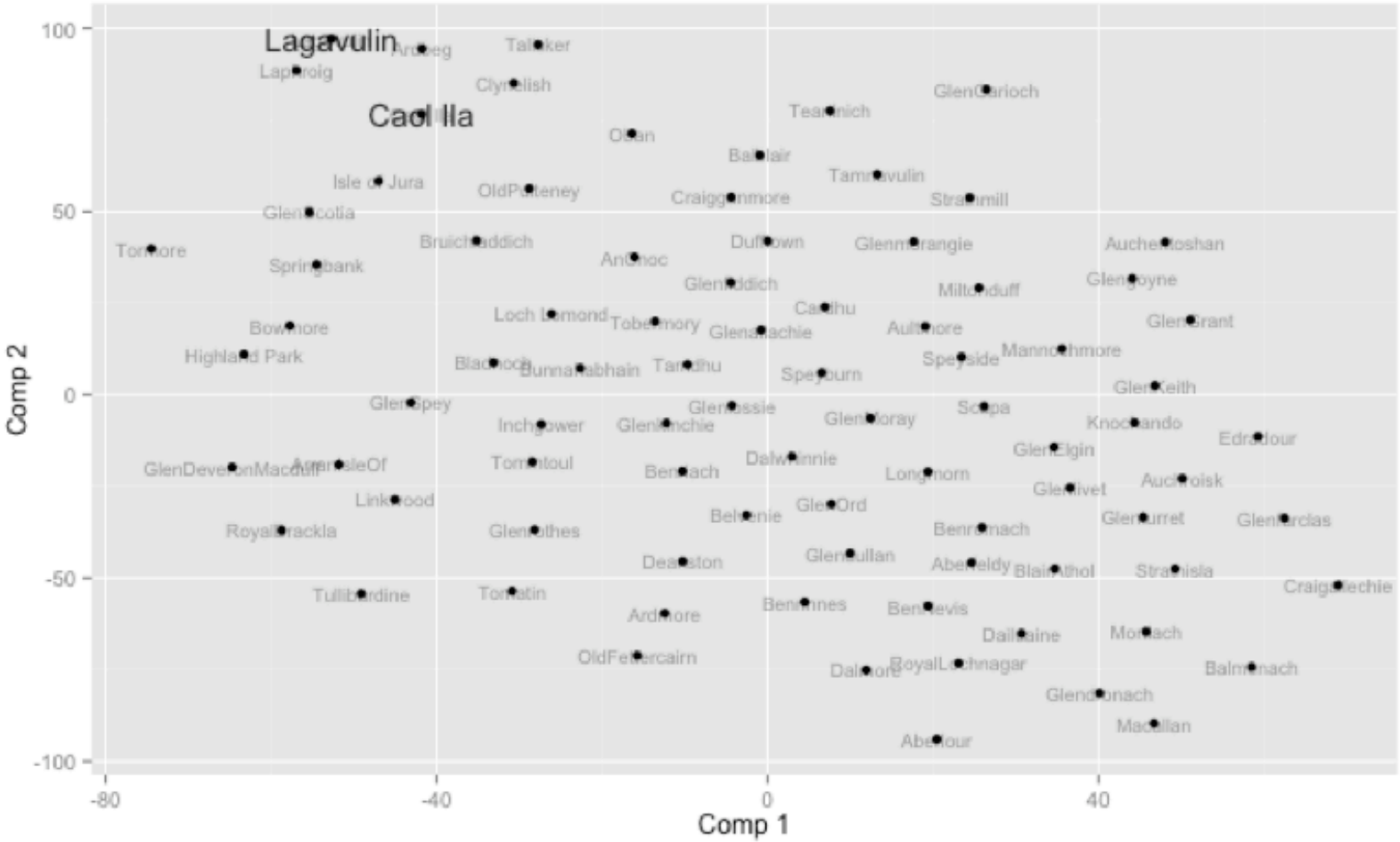
With distance you can



With distance you can



With distance you can



Similarities / Distances

What is Similarity?

The quality or state of being similar; likeness; resemblance; as, a similarity of features.

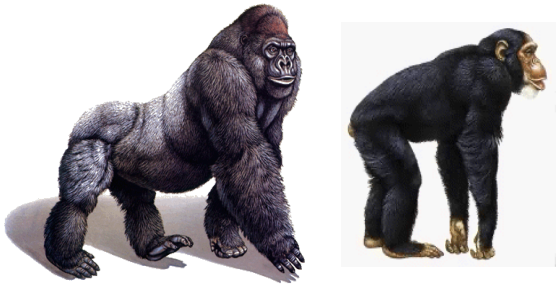
Webster's Dictionary

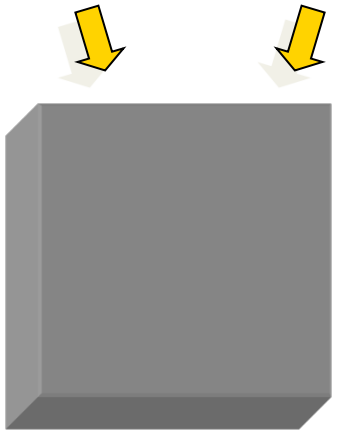
Similarity is hard to define, but… "*We know it when we see it*"

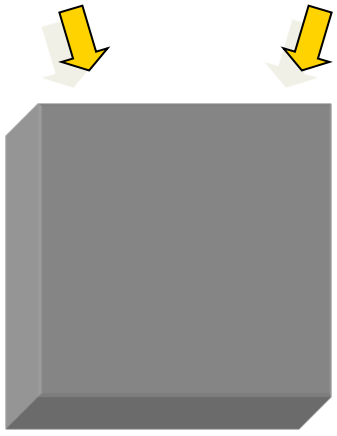
The real meaning of similarity is a philosophical question. We will take a more pragmatic approach.

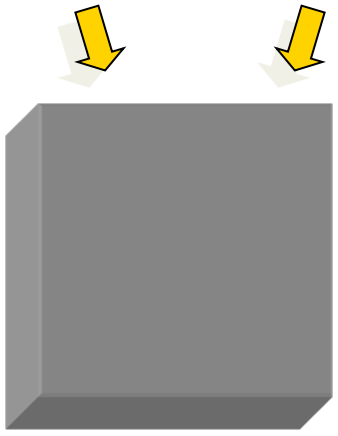
Defining Distance Measures (Recap)

Definition: Let O_1 and O_2 be two objects. The distance (dissimilarity) between O_1 and O_2 is a real number denoted by $d(O_1,O_2)$









0.23

3

342.7

(Dis-)similarities / Distance

Pairs of Objects:

Similarity	(large ⇒ similar), vague definition
Dissimilarity	(small ⇒ similar), Rules 1-3
Distance,Metric	(small ⇒ similar), Rule 4 in addition

Rules

1. $d(x, y) \geq 0$ (*non-negativity*, or separation axiom)
2. $d(x, y) = 0$ if and only if $x = y$ (*identity of indiscernibles*, or coincidence axiom)
3. $d(x, y) = d(y, x)$ (*symmetry*)
4. $d(x, z) \leq d(x, y) + d(y, z)$ (*subadditivity / triangle inequality*).

Examples of metrics (more follow with the examples)

- Euclidian and other Lp-Metrics
- Jaccard-Distance (1 - Jaccard Index)
- Graph Distance (shortest-path)

Example of a Metric

Task 1

- Draw 3 Objects and meassure their distances (e.g. by a ruler).
- Is this a proper distance? Are Axioms 1-4 fulfilled?

Task 2

- The 3 entities A,B,C have the dissimilarity:

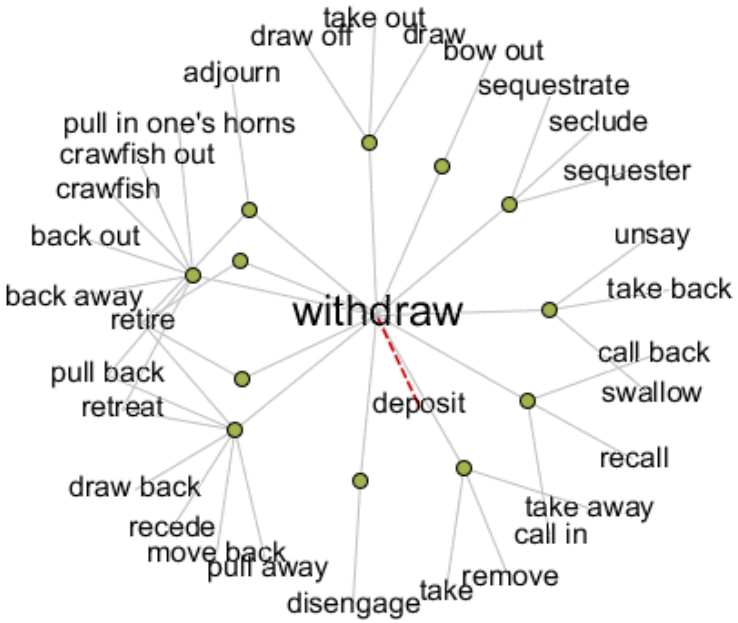
  $d(A,B) = 1$

  $d(B,C) = 1$

  $d(A,C) = 3$

- Is this dissimilarity a distance?
- Can you try to draw them on a piece of paper?

Problematic: Wordmaps



What about:

Bank
Finance
Sitting

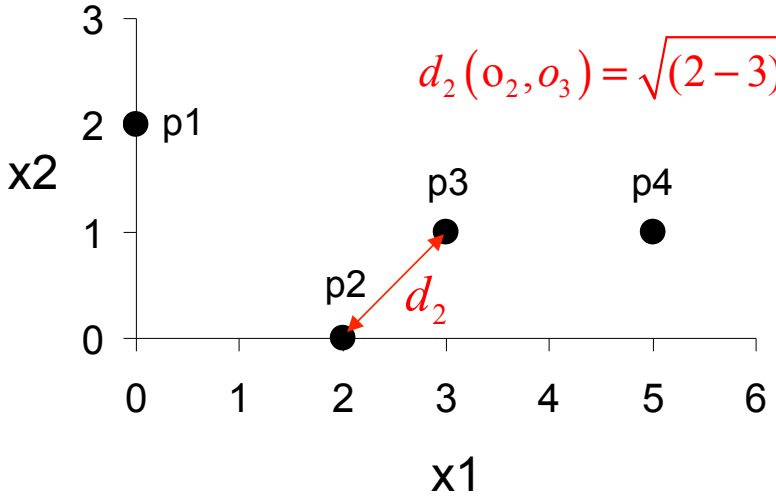
Triangular Inequality:
Not just a mathematical gimmick!

Triangle inequality would imply:
d(„sitting“, „finance“) ≤ d(„sitting“, „bank“) + d(„bank“, „finance“)

Euclidean Distance and its Generalization

2D example
(2 feature per observation)

obs	x1	x2
o1	0	2
o2	2	0
o3	3	1
o4	5	1



$$d_2(o_2, o_3) = \sqrt{(2-3)^2 + (0-1)^2} = \sqrt{2}$$

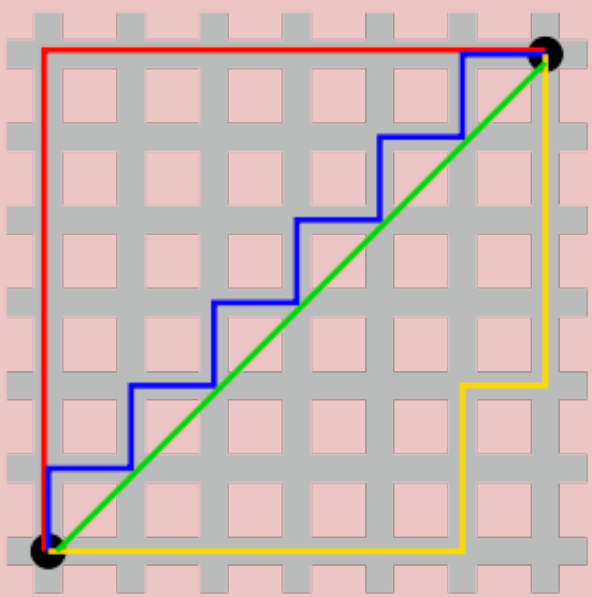
- Distance between observations $\mathbf{o}_i$, $\mathbf{o}_j$ p features describing each observation
- **Eucledian Distance** for 2 observations $\mathbf{o}_i$, $\mathbf{o}_j$, described by n numeric feature:

$$d_2(o_i, o_j) = \sqrt{\sum_{k=1}^{p} (o_{ik} - o_{jk})^2}$$

- **Minkowski Distance** as generalization

$$d_r(o_i, o_j) = \left(\sum_{k=1}^{p} |o_{ik} - o_{jk}|^r \right)^{\frac{1}{r}}$$

L1: Manhattan Distances



One block is one unit.

- How many Blocks you have to walk
- What is the L1 Distance from A to B
  - r=1
- What is the Euklidean Distance?

$$d_r\left(o_i, o_j\right) = \left(\sum_{k=1}^{p} | o_{ik} - o_{jk} |^r \right)^{\frac{1}{r}}$$

Image from Wikipedia

Minkowski Distances

Tafel:

- r = 1. City block (Manhattan, taxicab, L_1 norm) distance.

$$d_1(o_i, o_j) = \sum_{k=1}^{p} |o_{ik} - o_{jk}|$$

- r = 2. Euclidean distance (L_2 norm)

$$d_2(o_i, o_j) = \sqrt{\sum_{k=1}^{p} (o_{ik} - o_{jk})^2}$$

- r = ∞ "supremum" or maximum (L_{max} norm, L_∞ norm) distance.
  - This is the maximum difference between any component of the vectors

$$d_\infty(o_i, o_j) = \max_{k=1\ldots p} |o_{ik} - o_{jk}|$$

Distance matrix

As discussed on the last couple of slides there are different possibilities to determine the pair-wise distance between two observations o_i and o_j.

$$d\left(o_i, o_j\right) = d_{ij}$$

We can collect all these pair-wise distance d_{ij} in a distance matrix:

$$\mathbf{D} = \begin{bmatrix} d_{11} & d_{12} & . & . & d_{1n} \\ d_{21} & d_{22} & . & . & d_{2n} \\ . & & . & & . \\ . & & & . & . \\ d_{n1} & d_{n2} & . & . & d_{nn} \end{bmatrix}$$

Symmetry:

$$d_{ij} = d\left(o_i, o_j\right) = d\left(o_j, o_i\right) = d_{ji}$$

All diagonal elements are 0!

$$d\left(o_k, o_k\right) = d_{kk} = 0$$

General considerations on metrics

- N-1 entities with **any metric*** between them can be drawn in the N-dimensional (Euklidian) space preserving all of their mutual distances.

- Examples:
  - In 1-d you can always draw 2 entities
  - In 2-d you can always draw 3 entities
  - In 3-d you can always draw 4 entities
  - …

- What if you want to draw 100 entities?
  - We need a 99-dimensional space
- What if you want to draw 100 entities, on a piece of paper 2-D.
  - You have to do compromise (dimensionality reduction)

*Still looking for a prove in d>2

Principle Idea (it's all about compromise)

- Have data in high dimensional space with distance, (e.g. 99 features)

or (➔)

- Have distances / dissimilarities d_{ij} between many objects (e.g. 100 Objects)

- Draw this in low dimensional space (2, 3)

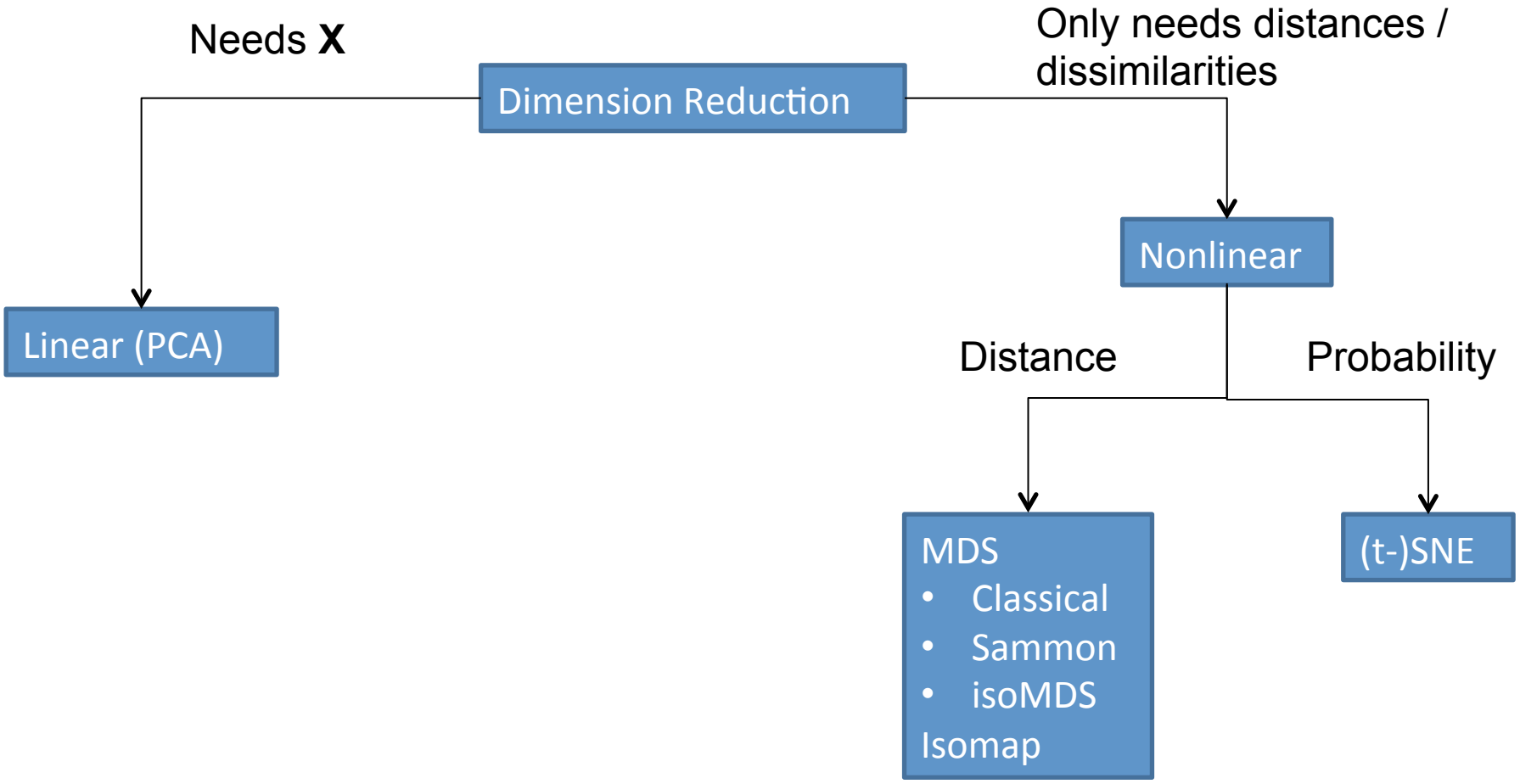
$$d_{ij} \rightarrow d^*_{ij} = \left\| \vec{y}_i - \vec{y}_j \right\|^2$$

High Dimensional Space

Low 2,3 Dimensional Space (Euklidean)

- The distances in (low-D) d^*_{ij} should match the original ones d_{ij} (high-D) as "**good as possible**"

Visualizing Distances: A Taxonomy of techniques



Other methods: e.g. autoencoder

For a fine grained map see (Maarten, http://www.iai.uni-bonn.de/~jz/dimensionality_reduction_a_comparative_review.pdf)

Classical Metric Scaling MDS

- Classical MDS. Formulation as minimisation of a cost function.
- In R: cmdscale()

$$\text{Cost} = \sum_{i<j} (d_{ij} - d^*_{ij})$$

$$d_{ij} = \| x_i - x_j \|^2$$

$$d^*_{ij} = \| y_i - y_j \|^2$$

Euclidean Distances also in high-D

Remarks

- Fast, “based on linear algebra”
- Only distances are needed as input (as all MDS methods)
- The formulation as a cost function is valid for Euclidian distances only (internally Eigenvalues are used)
- If other distances (besides Euclidian) are taken but nothing is guaranteed. Usually works if they are „mildly non-euclidean“, i.e. air-distances between cities on a Swiss map (small country, curvature of earth plays a minor role)
- Non-Euclidean distances -> negative Eigenvalues
- For Euclidean distances, classical MDS is equivalent to PCA (but conceptually different)

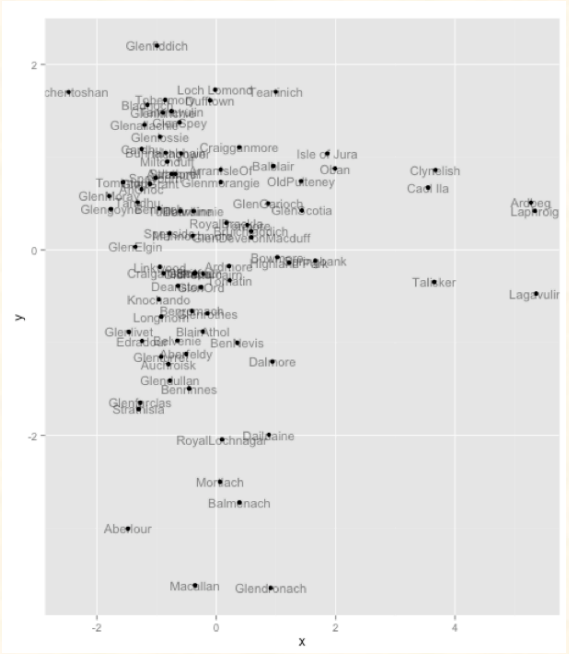
MDS in R

```
whiskies <- read.csv("../data/whiskies.txt", row.names = 2,
   stringsAsFactors = FALSE)
whiskies.f <- whiskies[,2:13]

d <- dist(whiskies.f, method = 'manhattan')
res <- cmdscale(d, eig = TRUE)
x <- res$points
```

Options (selection):

k is (maximum) number of
dimensions for representation




```
qplot(x,y,label = row.names(whiskies.f)) + geom_text(size=3, alpha=0.5)
```

How good is the reduction (Eigenvalues)

- If original distances are Euclidean, then Eigenvalues λ are positive
- If Eigenvalues are too negative other methods might be better (see below)
- Goodness of Fit using m-dimensions

$$P_m^{(1)} = \frac{\sum_{i=1}^{m} |\lambda_i|}{\sum_{i=1}^{n} |\lambda_i|}$$

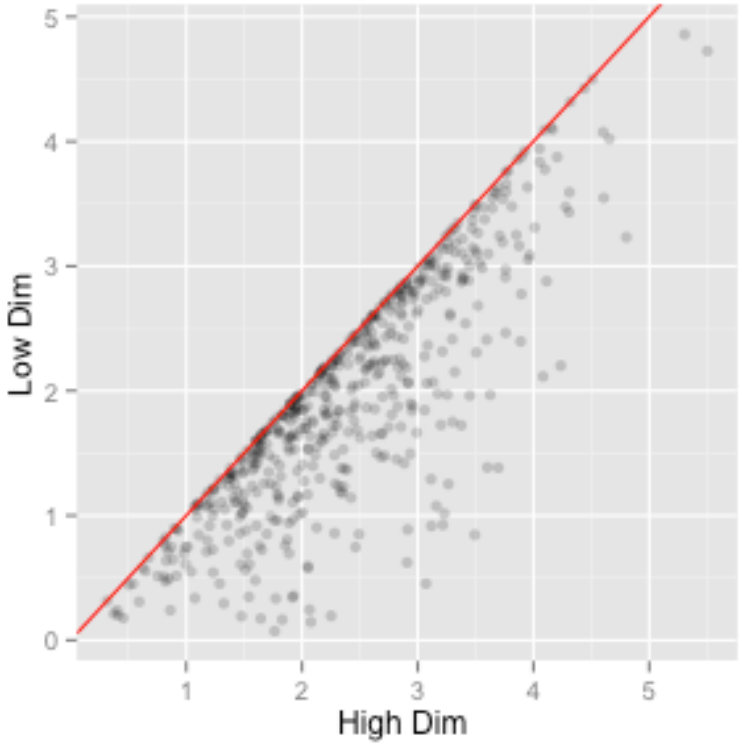
Similar to PCA (explained variance) but absolute values.
Values above 0.8 are good

```
•d <- dist(whiskies.f, method = 'euclidian') #Change to Euklidian
•r = cmdscale(d, eig = TRUE)
•min(r$eig) #-1.649809e-14, -164(euklidian, manhattan)
•p = (cumsum(abs(r$eig)) / sum(abs(r$eig)))
•qplot(1:length(p),p[1:length(p)]) +
•  xlab("Number of Eigenvectors") + ylab("P_m") +
geom_vline(xintercept=12)
```

Live!

How good is the fit (Shepard Diagram)

```
X <- matrix(rnorm(100), ncol = 3) #Play around change to 3
dd <- dist(X)
rr = cmdscale(dd, eig = TRUE)
shep <- Shepard(dd, rr$points)#MASS package
qplot(shep$x, shep$y, alpha=I(0.2)) +
  geom_abline(slope=1, color = 'red') +
  xlab("High Dim ") + ylab("Low Dim")
```



All pairwise distances are plotted

Ende HS 2016

Take home message from exercise

- PCA and metrical MDS are equivalent, if original distances are taken in Euclidean Space
- PCA and MDS reproduce the original data if original data is in 2 D.
- Metric MDS needs only distances
- Metric MDS OK (kind of) for non-Euclidean distances