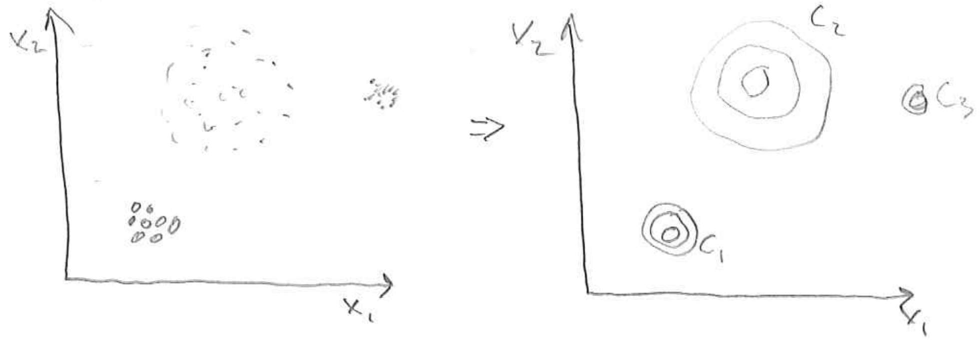
UNSUPERVISED LEARNING

So far in this course our problem:
find f such that

$$f : \mathcal{X} \to \mathcal{Y}$$

Given $\bar{x}_1 \to y_1, \bar{x}_2 \to y_2, \dots, \bar{x}_N \to y_N$.
How about if only $\bar{x}_i$ available?

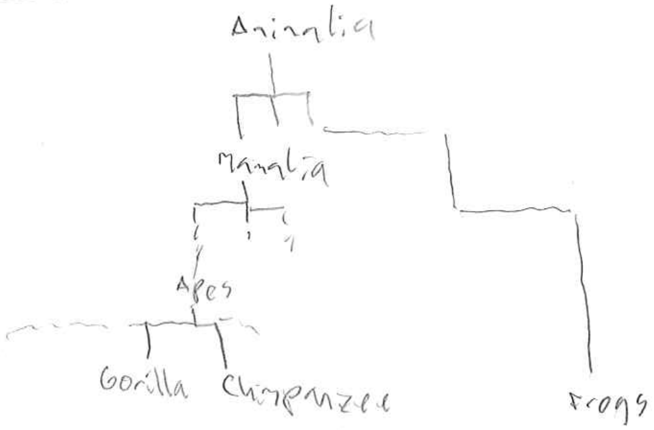
Example Plot if 2D



Approach: clustering

Example Data mining - name C_i if
x_1: age, x_2: income
$\bar{\mu}_1 = (15,200)$, $\bar{\mu}_2 = (30,2000)$, $\bar{\mu}_3 = (78,1500)$

Example



Sometimes data represents hierarchy rather than clusters

Unsupervised learning: <SLIDE>

HIERARCHICAL CLUSTERING

* Agglomerative (bottom-up)
* Divisive (top-down)

I. Agglomerative

To decide which to combine, we need

1° Distance metric

2° Linkage criteria (multiple points in a cluster)

1° Distances: $dist(\bar{a}, \bar{b})$ $\quad \bar{a} = (a_1, a_2, \ldots a_D), \bar{b} = (b_1, b_2, \ldots, b_D)$

L2 (Euclidean): $\sqrt{\sum_i (a_i - b_i)^2}$

L1 (Manhattan): $\sum_i |a_i - b_i|$

$L\infty$: $\max_i |a_i - b_i|$

Cosine : $\frac{a \cdot b}{\|a\| \|b\|}$ (angle of two vectors)

⋮

2° Linkage

Minimum (single-link)

Maximum

Mean

Centroid

etc.

Example <slide>

	1	2	(3,5)	4	6
1	0	4	12	24	8
2		0	10	22	10
(3,5)			0	6	8,5
4				0	18
6					0

⇒

	(1,2)	(3,5)	4	6
(1,2)	0	10	22	8
(3,5)		0	6	8,5
4			0	18
6				0

next is (3,4,5) . . .

Drawbacks: Chaining

MIXTURE MODELS IN CLUSTERING

$$p(\bar{x}) = \sum_{i=1}^{g} \pi_i \, p(\bar{x}; \bar{\theta}_i)$$

=> estimate π_i (priori of each cluster) and $\bar{\theta}_i$
(for gaussians: $\bar{\mu}_i, \bar{\Sigma}_i$) and sometimes also g (num of clusters)

Effective, but may need much data (e.g. Gaussian mixture models GMM). Topic related to statistics and density estimation.

SUM-OF-SQUARES METHODS

Idea is to define a grouping criteria, e.g.,

$$\sum_{i=1}^{k} \sum_{\bar{x}_j \in S_i} |\bar{x}_j - \bar{\mu}_i|^2$$

sum of squared errors of each data point to its cluster centre (k clusters each $|S_i|$ points).

Many methods, but one of the most popular is k-means:

1. Select k random points as the cluster means $\bar{\mu}_{i=1..k}$
2. Assign each point to its closest mean
3. Compute means of each cluster and update the means
4. Repeat 2. and 3. until convergence*

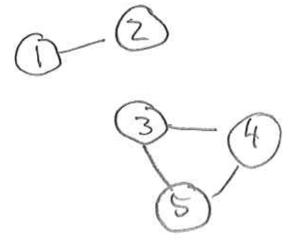
   No change in assignments
   small proportional change
   Max N iterations
   etc.

<matlab: kmeans_demo.m>
   ↳ same & diff variances

SPECTRAL CLUSTERING

Based on adjacency matrix A that summarises connections or similarities

Example



$$A = \begin{array}{c} \\ \\ \\ \\ \\ \\ \end{array}\begin{array}{c} 1\,2\,3\,4\,5 \\ \left(\begin{array}{ccccc} 0&1&0&0&0 \\ 1&0&0&0&0 \\ 0&0&0&1&1 \\ 0&0&1&0&1 \\ 0&0&1&1&0 \end{array}\right) \end{array} \begin{array}{c} \\ 1 \\ 2 \\ 3 \\ 4 \\ 5 \end{array}$$

We form a graph laplacian

$$L = D - A \qquad \left\{ D = \mathrm{diag}\left(\sum_{j=1}^{N} A_{ij}\right) \right\}$$

Example

$$L = \begin{pmatrix} 1&0&0&0&0 \\ 0&1&0&0&0 \\ 0&0&2&0&0 \\ 0&0&0&2&0 \\ 0&0&0&0&2 \end{pmatrix} - A = \begin{pmatrix} 1&-1&0&0&0 \\ -1&1&0&0&0 \\ 0&0&2&-1&-1 \\ 0&0&-1&2&-1 \\ 0&0&-1&-1&2 \end{pmatrix}$$

row sum to zero where diagonal represents # of links

If we do eigen value decomposition

$$A\bar{x} = \lambda \bar{x}$$

eig. vector — eig. value

the smallest eigen values represent the most "meaningful" graph component (cf. covariance matrix) => our clusters

Example

$$\lambda_4 = \lambda_5 = 0 \qquad \bar{x}_1 = \begin{pmatrix} a \\ a \\ 0 \\ 0 \\ 0 \end{pmatrix}, \bar{v}_2 = \begin{pmatrix} 0 \\ 0 \\ b \\ b \\ b \end{pmatrix}$$

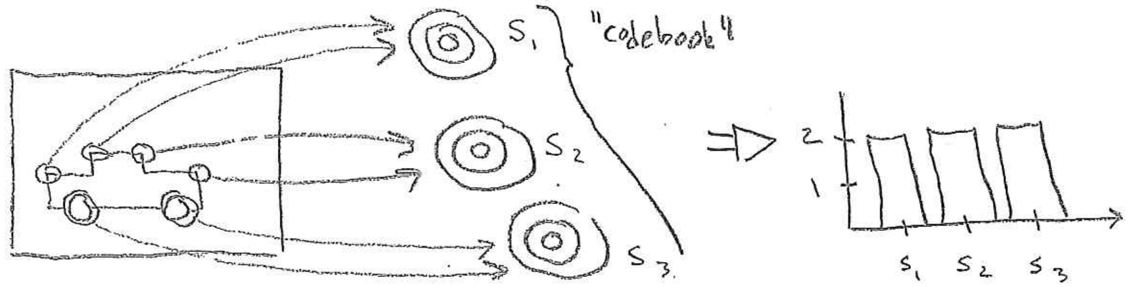
Note: A could be similarity instead of hard links, e.g.,) $e^{-\frac{(a-b)^2}{a}} \Rightarrow a=b \Rightarrow \mathrm{sim}(a,b)=1$

Example 3: Object categorisation using clustering

Let's assume we have a method for detecting "interesting points in image"

Slide: Interest point detection example

We may cluster these interest points to N clusters. Every image, whatever it contains, is represented by number of hits to the different clusters.



For any new image: 1) find interest points, 2) find clusters of each interest point, 3) compute histogram of cluster hits, 4) assign the image to the same category as the most similar histogram.

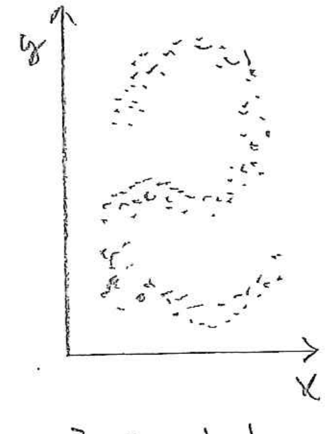
This state-of-the-art method automatically categorises images of any objects!

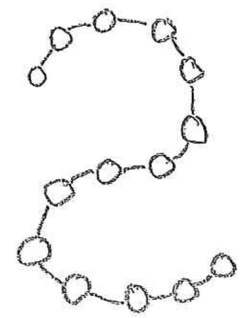
Apples
motorbikes
cars
humans
...
...

V. Self-organising map

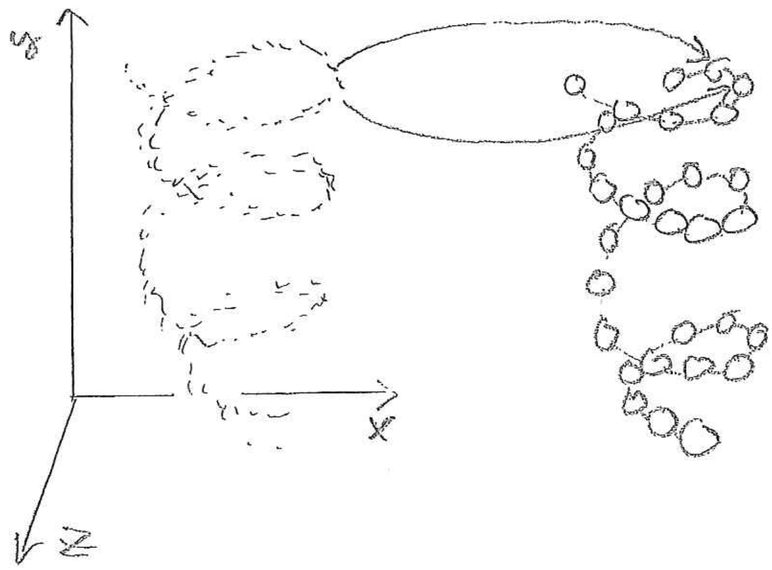
Example 4   SOM and topology preserving ordering

2-D data   1-D SOM





3-D data   1-D SOM

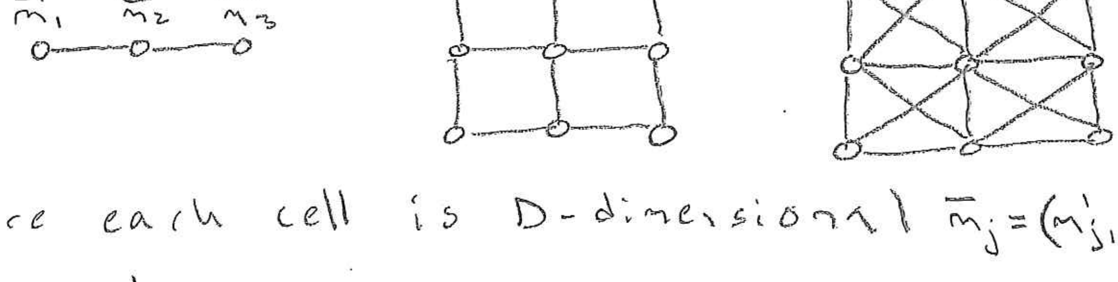


Are mapped to the neighbor cells, i.e. are similar in the original space

V.1. SOM algorithm

Input: N vectors $\bar{x}_i$ of D-dimensional data $\bar{x}_i = (x_i^1, x_i^2, \ldots, x_i^D)^T$

Connected network of cells $\bar{m}_j$, e.g.



where each cell is D-dimensional $\bar{m}_j = (m_j^1, m_j^2, \ldots, m_j^D)^T$

1. Repeat M times
2: Select a vector $\bar{x}_i$
3: Find the closest cell $\bar{m}_j$ (best matching unit, BMU) for $\bar{x}_i$

4: Adjust the $\bar{m}_{BMU}$ to more similar to $\bar{x}_i$

5: Adust neighbors of $\bar{m}_{BMU}$ to more similar to $\bar{x}_i$

6: End repeat

7: Return cells $\bar{m}_j$

How to find the "closest cell"?

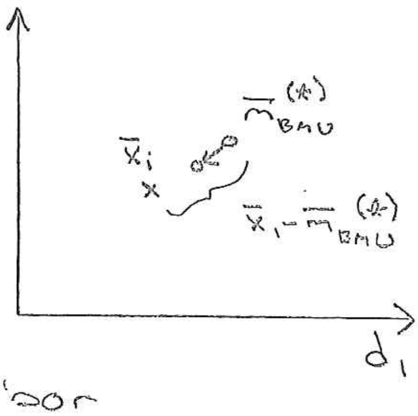
E.g. Euclidean distance

$$\bar{m}_{BMU} = \underset{\bar{m}_j}{\operatorname{argmin}} \ (\bar{x}_i - \bar{m}_j)^2 .$$

How to adjust cells to "more similar" to $\bar{x}_i$?

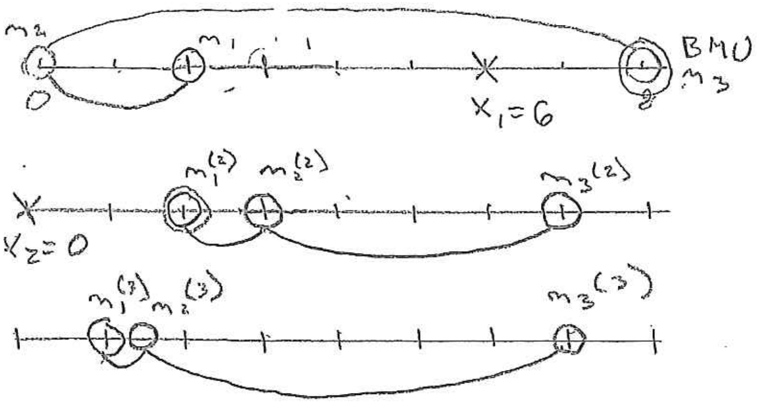
E.g. Add their difference to $\bar{m}_{BMU}$ with some training factor α

$$\bar{m}_{BMU}^{(t+1)} = \bar{m}_{BMU}^{(t)} + \alpha\left(\bar{x}_i - \bar{m}_{BMU}^{(t)}\right)$$



Example 5 1-D SOM, $\alpha = 0.5$, 1-neighbor

Init:



1:

2:

$(x_1 - m_3) = -2$

$(x_1 - m_2) = 6$

$(x_2 - m_1) = -2$

$(x_2 - m_2) = -3$

V.2. Considerations with SOM algorithm

- Datan normalisointi
- You may define different neighborhoods and it may change during the training (example slide)
- Value of α may depend on the neighbourhood distance and may also change during the training
- Topology of the cell structure may be changed (linear 1-D, ring 1-D, 2-D sheet, 2-D cylinder, 2-D torus)
- Most importantly, you may change the number of neurons

However, SOM is very robust and typically provides useful results even for amateurs

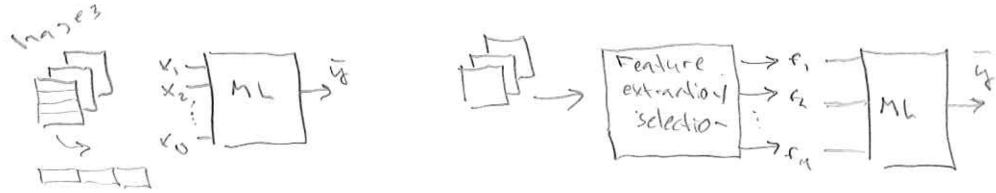
VI. Applications of SOM

<slides>

FEATURE SELECTION AND EXTRACTION

Idea:

instead of



Note: Cells in human visual and hearing system

Reasons:
Easier to learn by many methods if we normalise variables, emphasise relevant and surpress irrelevant information/noise and generally reduce dimensions (cf. overfitting)

1. Transformations

Plenty of different and depend on application. For example, translation variance

(e.g. mean val varies)

Absolute value of Fourier transformation is translation invariant

In 2D: location, rotation, scaling

2. Supervised selection

A combination of M (M<N) variables that perform well. You can select any set, train a classifier and evaluate.

- statistical measures (mutual information etc.)
- Random selection (combine with multiple classifiers)
- search (branch and bound)

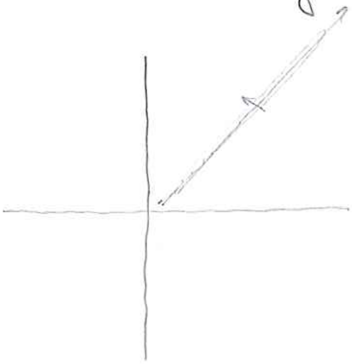
→ * Fisher kernel ←

3. Unsupervised selection

Most popular - generally unsuitable - ideas similar to compression (convey the same information with less bits).

3.1 Principal component analysis (PCA)

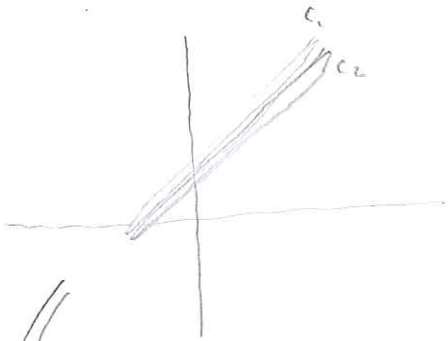
Based on properties of data covariance matrix:



data can be compressed upto 9X% using only the major axis

Idea: map data to the eigenvectors of the M largest eigenvalues

Problem:



Supervised alternative: Linear discriminant analysis (LDA)

=> finds directions that best separate two or more (single gaussian) classes

+ state-of-the-art: Fisher kernel mapping