university of groningen

# (B) Distance-based systems

**Michael Biehl**
Bernoulli Institute for Mathematics,
Computer Science and Artificial Intelligence
University of Groningen

www.cs.rug.nl/biehl

**Introduction:**

- supervised learning, clasification, regression
- machine learning “vs.” statistical modeling

**Early (important!) systems**

- linear threshold classifier, Rosenblatt’s *Perceptron*
- adaptive linear neuron, Widrow and Hoff’s *Adaline*

From **Perceptron** to **Support Vector Machine**

- large margin classification
- beyond linear separability

**Distance-based systems**

- prototypes: K-means and Vector Quantization
- from K-Nearest_Neighbors to Learning Vector Quantization
- adaptive distance measures and relevance learning

# overview

**Basic concepts of similarity / distance based classification**

- prototype based systems: Vector Quantization, K-means
- (K) Nearest Neighbor classifier
- Learning Vector Quantization (LVQ)

**Distance measures and Relevance Learning**

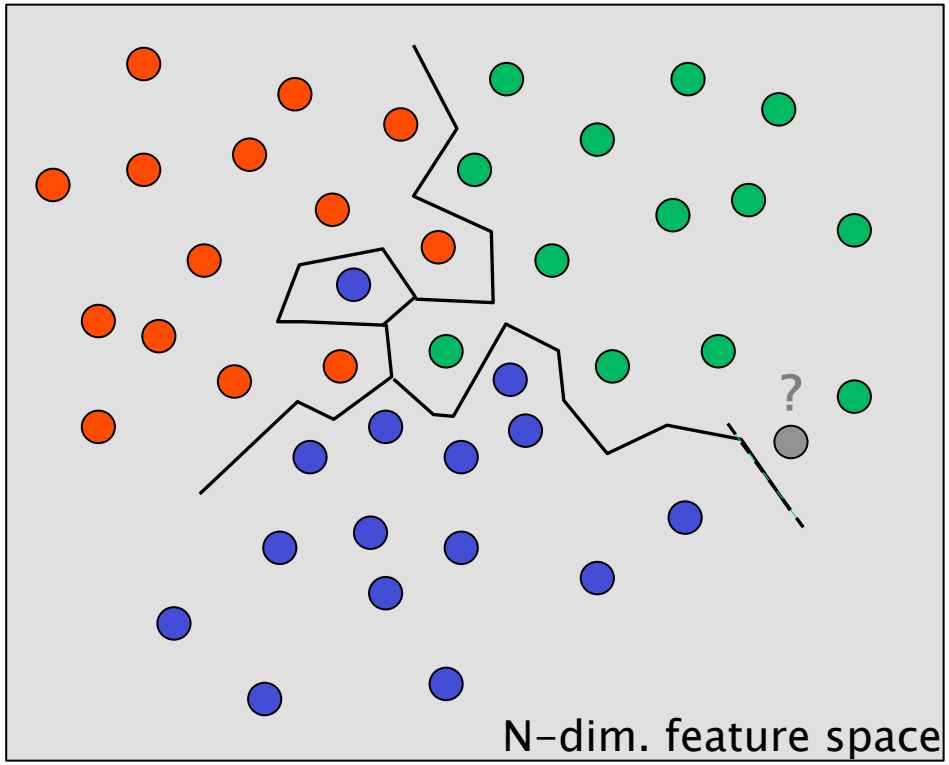
- predefined distances, e.g. divergence based LVQ
- adaptive distances, e.g. Matrix Relevance LVQ

university of groningen

# Distance-based classification

# distance-based classifiers

## a simple distance-based system: (K) NN classifier

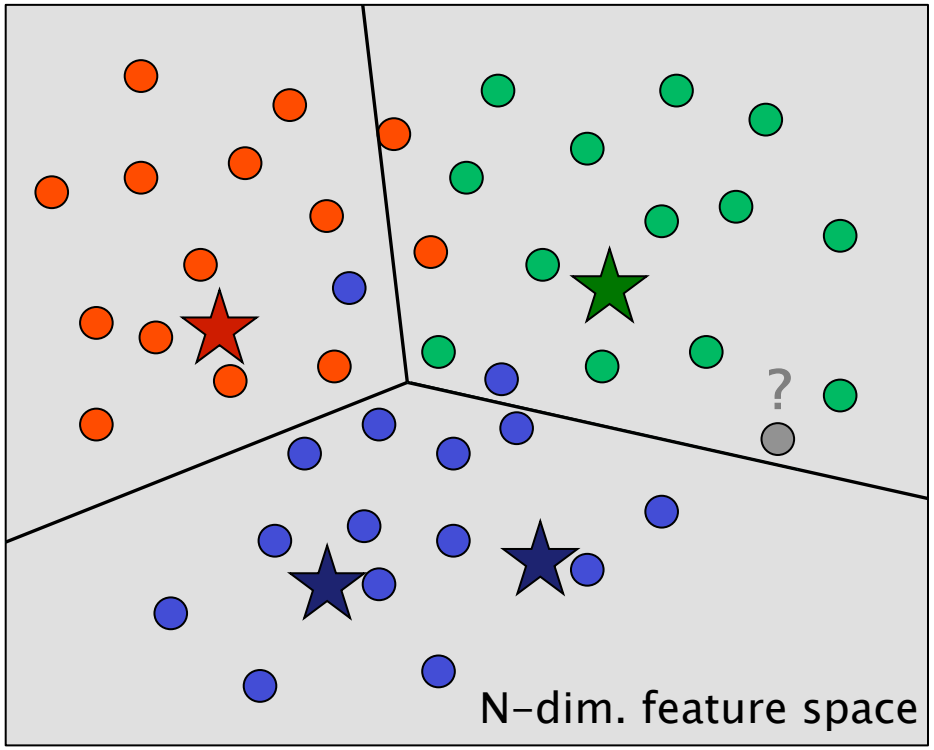



- store a set of labeled examples
- classify a query according to the label of the **Nearest Neighbor** (or the majority of **K NN**)
- piece-wise linear decision boundaries according to (e.g.) Euclidean **distance** from all examples

| | |
|---|---|
| + conceptually simple, | - expensive (storage, computation) |
| + no *training* phase | - sensitive to mislabeled data |
| + only one parameter (K) | - overly complex decision boundaries |

# prototype-based classification

## Learning Vector Quantization [Kohonen]




- represent the data by one or several **prototypes** per class
- classify a query according to the label of the **nearest prototype** (or alternative schemes)
- local decision boundaries acc. to (e.g.) Euclidean distances

\+ robust, low storage needs, little computational effort

\+ parameterization in feature space, interpretability

\− model selection: number of prototypes per class, etc.

\− requires training: placement of prototypes in feature space

# Nearest Prototype Classifier

set of **prototypes** $\mathbf{w}^1, \mathbf{w}^2, \ldots \mathbf{w}^K \qquad \mathbf{w}^k \in I\!R^N$

carrying class-labels $S^1, S^2, \ldots S^K \qquad S^k = S(\mathbf{w}^k) \in \{1, 2, \ldots C\}$

**nearest prototype classifier (NPC):**

based on dissimilarity/distance measure $d[\mathbf{w}, \mathbf{x}]$

given $\mathbf{x} \in I\!R^N$ : – determine the *winner* $\mathbf{w}^* = \mathrm{argmin}_j \left\{ d[\mathbf{w}^j, \mathbf{x}] \right\}$

– assign $\mathbf{x}$ to the class $S(\mathbf{w}^*) = S^*$

reasonable requirements: $d[\mathbf{x}, \mathbf{x}] = 0, \quad d[\mathbf{w}, \mathbf{x}] \geq 0 \text{ for } \mathbf{w} \neq \mathbf{x}$

most prominent example: (squared) **Euclidean distance**

$$d[\mathbf{w}, \mathbf{x}] = \sum_{j=1}^{N} (w_j - x_j)^2$$

university of groningen

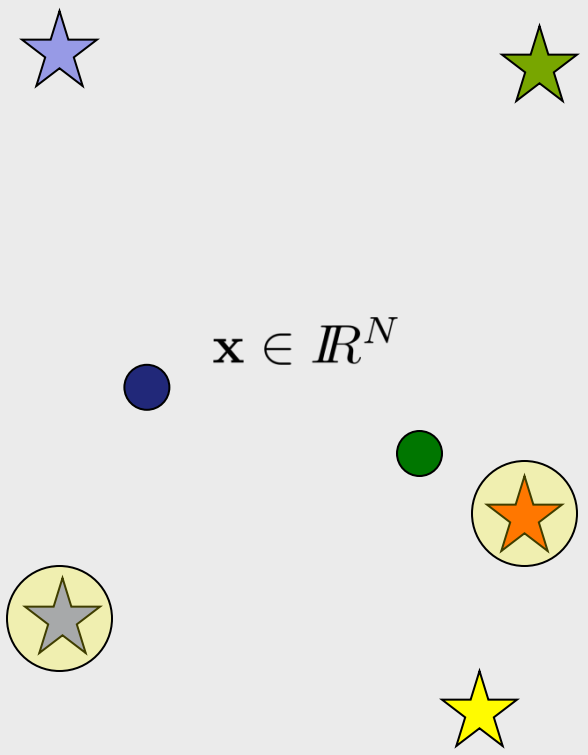
# Learning Vector Quantization

N-dimensional data, feature vectors

- identification of **prototype vectors** from labeled example data
- distance based **classification** (e.g. Euclidean)

**competitive learning**: **LVQ1** [Kohonen]

- **initialize** prototype vectors for different classes
- present a **single example**
- identify the **winner** (closest prototype)
- move the winner
  - closer **towards the data** (same class)
  - **away from the data** (different class)



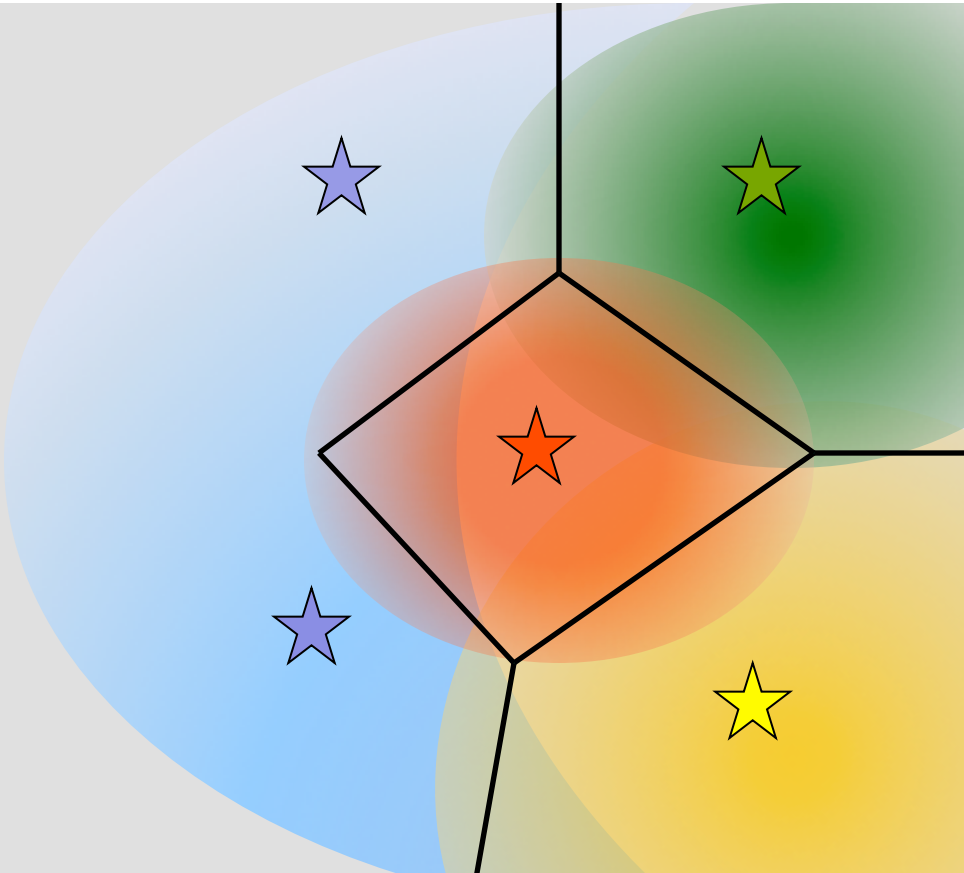
university of groningen

# Learning Vector Quantization

N-dimensional data, feature vectors

- identification of **prototype vectors** from labeled example data
- distance based **classification** (e.g. Euclidean)

- **distance-based classification**
  [here: Euclidean distances]
- **tesselation of feature space**
  [piece-wise linear]
- aim: **discrimination of classes**
  ( ≠ vector quantization
  or density estimation )
- **generalization ability**
  correct classification of *new* data



# LVQ1

**iterative training procedure**:

randomized initial $\{\mathbf{w}^k\}$ , e.g. close to the class-conditional means

sequential pres. of **labelled** examples $\{\mathbf{x}^m, \sigma^m\} \quad m = 1, 2, \ldots$

*… the winner takes it all:* $\mathbf{w}^* = \mathrm{argmin}_j \{d[\mathbf{w}^j, \mathbf{x}^m]\}$

**LVQ1 update step:**

$$\mathbf{w}^* \leftarrow \mathbf{w}^* + \eta_w \, \Psi(S^*, \sigma^m) \, (\mathbf{x}^m - \mathbf{w}^*)$$

$$\text{where } \Psi(S, \sigma) = \begin{cases} +1 & \text{if } S = \sigma \\ -1 & \text{else.} \end{cases}$$

$\eta_w$ : learning rate

many heuristic variants/modifications:

- learning rate schedules $\eta_w(t)$
- update more than one prototype per step

# LVQ1

**LVQ1 update step:**

$$\mathbf{w}^* \leftarrow \mathbf{w}^* - \eta_w \, \Psi(S^*, \sigma^m) \, (\mathbf{w}^* - \mathbf{x}^m)$$

$$\text{where } \Psi(S, \sigma) = \begin{cases} +1 & \text{if } S = \sigma \\ -1 & \text{else.} \end{cases}$$

**LVQ1–like update for generalized distance:**

$$\mathbf{w}^* \leftarrow \mathbf{w}^* - \eta_w \, \Psi(S^*, \sigma^m) \, \frac{1}{2} \, \boxed{\frac{\partial d(\mathbf{w}^*, \mathbf{x}^m)}{\partial \mathbf{w}^*}}$$

$$\text{where } \Psi(S, \sigma) = \begin{cases} +1 & \text{if } S = \sigma \\ -1 & \text{else.} \end{cases}$$

addtl. requirement: $d[\mathbf{w}, \mathbf{x}]$ differentiable w.r.t. $\mathbf{w}$

update decreases (increases) distance if classes coincide (are different)

# remark: the curse of dimension ?

**concentration of distances** for large *N*

„distance based methods are bound to fail in high dimensions“ ???

LVQ:

- prototypes are not just random data points
- carefully selected *low-noise* representatives of the data
- distances of a given data point to prototypes are **compared**

$$d(\mathbf{w}_a, \mathbf{x}) - d(\mathbf{w}_b, \mathbf{x}) = \left[\mathbf{x}^2 - \mathbf{x}^2\right] + \left[\mathbf{w}_a^2 - \mathbf{w}_b^2\right] - 2\left[\mathbf{w}_a \cdot \mathbf{x} - \mathbf{w}_b \cdot \mathbf{x}\right]$$

**projection to non-trivial low-dimensional subspace!**

# cost function based LVQ

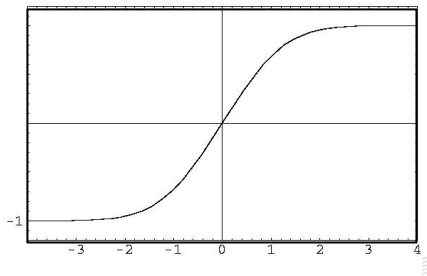
one example: **Generalized LVQ (GLVQ) cost function** [Sato&Yamada, 1995]

$$\text{minimize } E = \sum_m e(\mathbf{x}^m) \text{ with } e(\mathbf{x}^m) = \varphi\left[\frac{d(\mathbf{w}^J, \mathbf{x}^m) - d(\mathbf{w}^K, \mathbf{x}^m)}{d(\mathbf{w}^J, \mathbf{x}^m) + d(\mathbf{w}^K, \mathbf{x}^m)}\right]$$

two winning prototypes: $\begin{cases} \mathbf{w}^J & \text{closest correct} \\ \mathbf{w}^K & \text{closest wrong} \end{cases}$ w.r.t. $\mathbf{x}^m$

*E* favors

- small number of misclassifications, e.g. with $\varphi(x) = \frac{1}{1+e^{-\gamma x}}$
- *large margins* between classes
- small $d^J$, large $d^K$
- class-typical prototypes



# GLVQ

**training** = optimization with respect to prototype position,

e.g. single example presentation, stochastic sequence of examples,

update of two prototypes per step $\left\{\mathbf{w}^J, \mathbf{w}^K\right\}$

based on **non-negative, differentiable distance**

$$\Delta \mathbf{w}^J = -\eta_w \frac{1}{2} \frac{\partial e^m}{\partial \mathbf{w}^J}$$

$$\Delta \mathbf{w}^K = -\eta_w \frac{1}{2} \frac{\partial e^m}{\partial \mathbf{w}^K}$$

$$E = \sum_m e^m$$

# GLVQ

**training** = optimization with respect to prototype position,

e.g. single example presentation, stochastic sequence of examples,

update of two prototypes per step $\{\mathbf{w}^J, \mathbf{w}^K\}$

based on **non-negative, differentiable distance**

$$\Delta \mathbf{w}^J = -\eta_w \underbrace{\frac{\varphi'[\cdots]\; d^K}{(d^J + d^K)^2}}_{\xi_J} \frac{1}{2} \frac{\partial d(\mathbf{w}^J, \mathbf{x}^m)}{\partial \mathbf{w}^J}$$

$$d^L = d(\mathbf{w}^L, \mathbf{x})$$

$$\Delta \mathbf{w}^K = +\eta_w \underbrace{\frac{\varphi'[\cdots]\; d^J}{(d^J + d^K)^2}}_{\xi_K} \frac{1}{2} \frac{\partial d(\mathbf{w}^K, \mathbf{x}^m)}{\partial \mathbf{w}^K}$$

# GLVQ

**training** = optimization with respect to prototype position,

e.g. single example presentation, stochastic sequence of examples,

update of two prototypes per step $\left\{\mathbf{w}^J, \mathbf{w}^K\right\}$

based on **non-negative, differentiable distance**

$$\Delta\mathbf{w}^J = -\eta_w \, \xi_J \, \left(\mathbf{w}^J - \mathbf{x}^m\right)$$

$$\Delta\mathbf{w}^K = +\eta_w \, \xi_K \, \left(\mathbf{w}^K - \mathbf{x}^m\right)$$

moves prototypes towards / away from

sample with prefactors $\xi_K, \xi_J \geq 0$

# Alternative distance measures

**fixed, pre-defined distance measures**: heuristic LVQ1, GLVQ (or more general cost function based LVQ): can be based on general, differentiable distances,

e.g. Minkowski measures $$d_p(\mathbf{w}, \mathbf{x}) = \Big( \sum_j |w_j - x_j|^p \Big)^{1/p}$$

**possible work-flow**

- select several distance measures according to prior knowledge or a data-driven choice in a preprocessing step
- compare performance of various measures

examples: kernelized distances

divergences (statistics)

# Kernelized distances

rewrite squared Euclidean distance in terms of *dot-product*

$$d(\mathbf{w},\mathbf{x}) = \mathbf{w}^2 - 2\mathbf{w}\cdot\mathbf{x} + \mathbf{x}^2$$

analagous: distance measure associated with general inner product or kernel function

$$d_K(\mathbf{w},\mathbf{x}) = K(\mathbf{w},\mathbf{w}) - 2K(\mathbf{w},\mathbf{x}) + K(\mathbf{x},\mathbf{x})$$

e.g. Gaussian Kernel

$$K(\mathbf{x},\mathbf{y}) = \exp\left(-\frac{(\mathbf{x}-\mathbf{y})^2}{2\sigma^2}\right) \qquad \text{with kernel width } \sigma$$

implicit mapping to high-dimensional space for better seperability of classes, similar: *Support Vector Machine*

Biehl, Hammer, Villmann. 2014 Distance measures for prototype-based classification
2016 Prototype-based models in machine learning

# Relevance Learning

elegant approach: **Relevance Learning / adaptive distances**

- employ a parameterized distance measure
  with only the mathematical form fixed in advance
- optimize its parameters in the training process
- adaptive, data driven dissimilarity

example: **Matrix Relevance LVQ**

- **data-driven** optimization of prototypes
  and *relevance matrix*
- in the same training process (≠ pre-processing )

# Generalized Matrix Relevance LVQ: GMLVQ

[Schneider, Biehl, Hammer, 2009]

generalized **quadratic distance** in LVQ:

$$d(\mathbf{w},\mathbf{x}) = (\mathbf{w}-\mathbf{x})^{\top}\ \Lambda\ (\mathbf{w}-\mathbf{x}) = \left[\,\Omega\,(\mathbf{w}-\mathbf{x})\,\right]^2$$

$\Lambda$ non-negative, e.g. $\Lambda = \Omega^{\top}\Omega$ with $\Omega \in I\!R^{N\times N}$

# Generalized Matrix Relevance LVQ: GMLVQ

[Schneider, Biehl, Hammer, 2009]

generalized **quadratic distance** in LVQ:

$$d(\mathbf{w},\mathbf{x}) = (\mathbf{w}-\mathbf{x})^{\top}\ \Lambda\ (\mathbf{w}-\mathbf{x}) = \left[\,\Omega\,(\mathbf{w}-\mathbf{x})\,\right]^2$$

$$\Lambda \text{ non-negative, e.g. } \Lambda = \Omega^{\top}\Omega \quad \text{with} \quad \Omega \in I\!R^{N\times N}$$

**training**: adaptation of prototypes and distance measure guided by GLVQ cost function

**variants**:

one **global**, several **local**, class-wise relevance matrices

**diagonal** matrices: single feature weights [Hammer et al., 2002]

**rectangular** $\Omega \in I\!R^{N\times M}$ : low-dim. representation / visualization [Bunte et al., 2012]

# interpretation

after training:

**prototypes** represent typical class properties or subtypes

**Relevance Matrix**

$\Lambda_{ij}$ quantifies the contribution of the pair of features (i,j) to the distance

$\Lambda_{jj} = \sum_i \Omega_{ij}^2$ summarizes

- the contribution of a single dimension $j$
- **the relevance** of original features in the classifier

Note: interpretation assumes implicitly that features have equal order of magnitude

e.g. after z-score-transformation $\rightarrow \langle x_j \rangle = 0, \quad \langle x_j^2 \rangle = 1$

(averages over data set)

university of groningen

# But this is just Mahalonobis distance...

No.

$$d_M(\mathbf{x}, \mathbf{y}) = \sqrt{(\mathbf{x} - \mathbf{y})^\top S^{-1} (\mathbf{x} - \mathbf{y})}$$

[Mahalonobis, 1936]

('two point version')

$S$ **covariance matrix** of random vectors $\mathbf{x} \in \mathbb{R}^N$
(calculated once from the data, fixed definition, not adaptive)

*So it is a generalized Mahalonobis distance ?*

if you insist...

a generalized broccoli

$$E = \hbar\omega$$

a generalization of Ohm's Law

# Relevance Matrix LVQ

optimization of **prototypes** and **distance measure**

$$\frac{1}{2}\frac{\partial d_\Omega(\mathbf{w},\mathbf{x})}{\partial \mathbf{w}} = \Omega^\top \Omega(\mathbf{w}-\mathbf{x}) \qquad \frac{1}{2}\frac{\partial d_\Omega(\mathbf{w},\mathbf{x})}{\partial \Omega} = \Omega\,(\mathbf{w}-\mathbf{x})\,(\mathbf{w}-\mathbf{x})^\top$$

**Generalized Matrix LVQ (GMLVQ)**

gradient terms for single example $\mathrm{x}^m$

$$\Delta \mathbf{w}^J = -\eta_w\ \xi_J\ \frac{1}{2}\ \frac{\partial d_\Omega(\mathbf{w}^J,\mathbf{x}^m)}{\partial \mathbf{w}^J}$$

$$\Delta \mathbf{w}^K = +\eta_w\ \xi_K\ \frac{1}{2}\ \frac{\partial d_\Omega(\mathbf{w}^K,\mathbf{x}^m)}{\partial \mathbf{w}^K}$$

$$\Delta \Omega = +\eta_\Omega \left[\ \xi_J\ \frac{1}{2}\frac{\partial d_\Omega(\mathbf{w}^J,\mathbf{x}^m)}{\partial \Omega}\ -\ \xi_K\ \frac{1}{2}\frac{\partial d_\Omega(\mathbf{w}^K,\mathbf{x}^m)}{\partial \Omega}\right]$$

# Relevance Matrix LVQ

optimization of { **prototype positions** ; **distance measure(s)** } in one training process (≠ pre-processing)

**motivation:**

**improved performance**

- weighting of features and pairs of features

**simplified classification schemes**

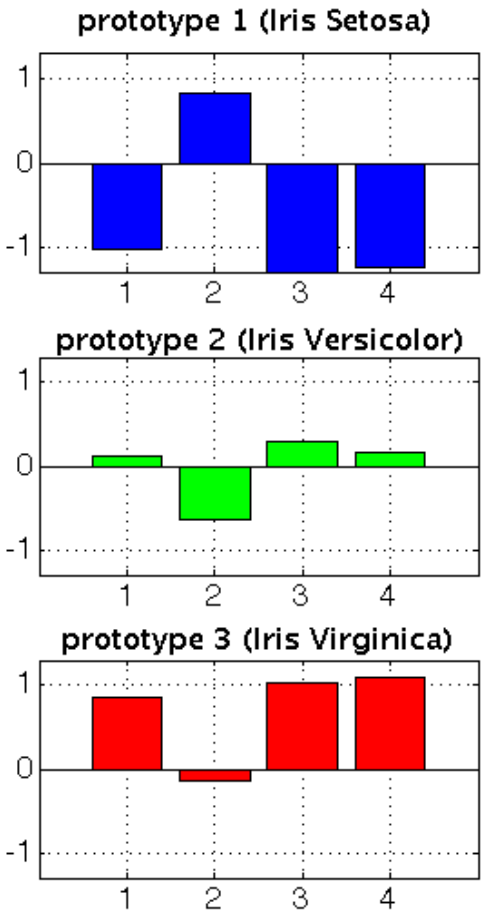
- elimination of non-informative, noisy features
- discriminative low-dimensional representation

**insight into the data / classification problem**

- identification of most discriminative features
- incorporation of prior knowledge (e.g. structure of $\Omega$)

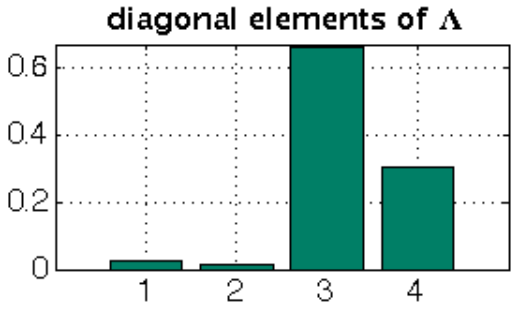
# Relevance Matrix LVQ

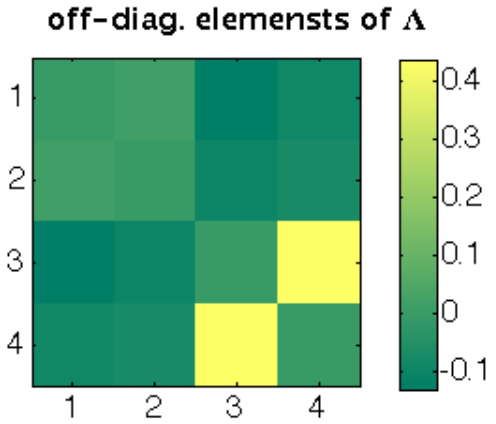
## *Iris flower* data




GMLVQ prototypes

relevance matrix



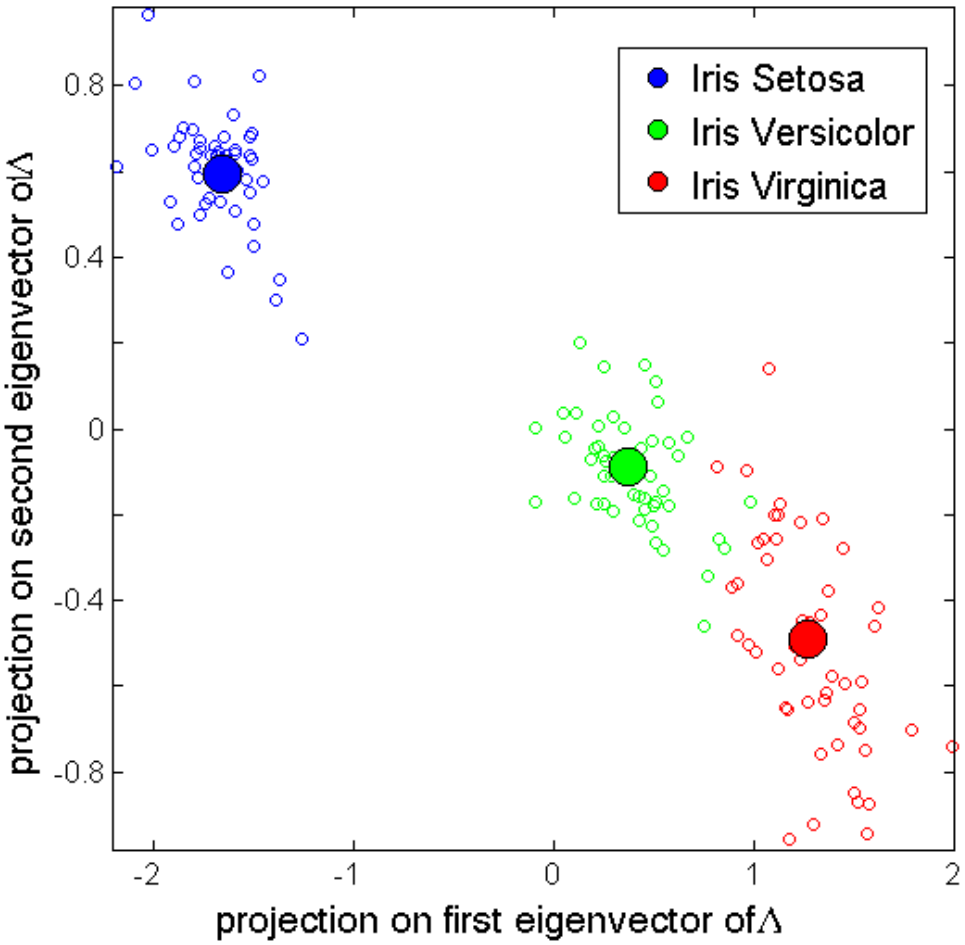
# Relevance Matrix LVQ

empirical observation / theory:
relevance matrix becomes
singular, dominated by
very few eigenvectors

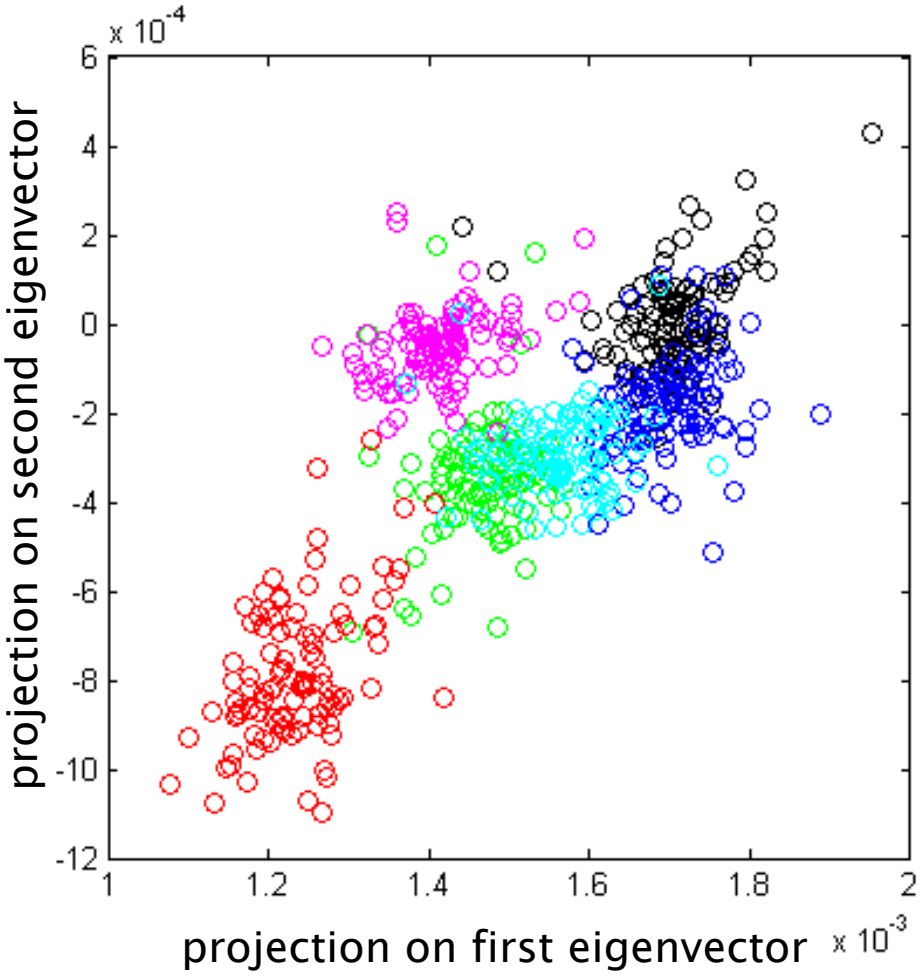
prevents over-fitting in
high-dim. feature spaces

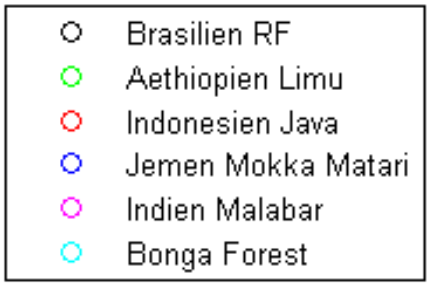
facilitates discriminative
visualization of datasets

confirms: Setosa well-separated
from Virginica / Versicolor



# a multi-class example



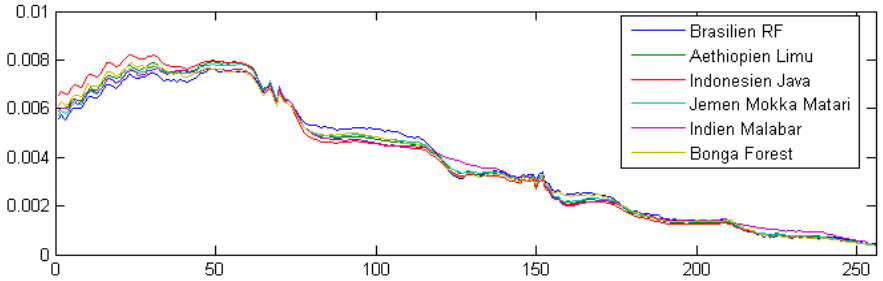




**classification of coffee samples**

based on hyperspectral data

(256-dim. feature vectors)

[U. Seiffert et al., IFF Magdeburg]




prototypes

# related schemes

**Linear Discriminant Analysis (LDA)**

~ one prototype per class + global matrix,

different objective function!

Relevance Learning related schemes in supervised learning ...

| | |
|---|---|
| **RBF Networks** | [Backhaus et al., 2012] |
| **Neighborhood Component Analysis** | [Goldberger et al., 2005] |
| **Large Margin Nearest Neighbor** | [Weinberger et al., 2006, 2010] |

and many more!

**Relevance LVQ variants**

local, rectangular, structured, restricted... relevance matrices

for visualization, functional data, texture recognition, etc.

relevance learning in Robust Soft LVQ, Supervised NG, etc.

combination of distances for *mixed data ...*

# links

**Matlab code:**

Relevance and Matrix adaptation in Learning Vector Quantization (GRLVQ, GMLVQ and LiRaM LVQ) [K. Bunte]

**http://matlabserver.cs.rug.nl/gmlvqweb/web/**

**A no-nonsense beginners' tool for GMLVQ:**
**http://www.cs.rug.nl/~biehl/gmlvq**

A **Scikit-Learn** compatible collection of **Python code** for LVQ and variants, including GMLVQ,
[B. Paaßen et al., CITEC Bielefeld]

**https://techfak.uni-bielefeld.de/~bpaassen/glvq.zip**
**https://github.com/MrNuggelz/sklearn-glvq**

**Related pre- and re-prints etc.:** **http://www.cs.rug.nl/~biehl/**

# References

**Overview articles which relate directly to the lectures and provide further refs.**

M. Biehl, B. Hammer, T. Villmann.
*Prototype-based models in machine learning.*
**WIREs Cognitive Scienes.** 2016. Advanced review, 20 pages

M. Biehl, B. Hammer, T. Villmann.
*Prototype-based models for the supervised learning of classification.*
In: **Proceedings IAU Symposium No. 325**, 2016
M. Brescia, S.G. Djogovski, E. Feigelson, G. Longo & S. Cavuoti (eds.),
10 pages, 2016.

M. Biehl, B. Hammer , T. Villmann.
*Distance measures for prototype based classification.*
In: Grandinetti L, Lippert T, Petkov N (eds.).
**Brain-Inspired Computing.** Lecture Notes in Computer Science.
Springer International Publishing, pp. 100-116, 2014.