Speech Pattern Classification

A practical approach to feature extraction, machine learning and common tasks

Alberto Abad & Isabel Trancoso

IST/INESC-ID

alberto@l2f.inesc-id.pt

Speech Processing - **IST** Lisboa, April, 2016

PART II

PATTERN CLASSIFICATION FOR SPEECH

Introduction to ML

- Assume we have a training set D={(x(i),y(i))} drawn from the distribution p(x,y), x€X y€Y

- The goal of learning is to find a decision function f: X → Y that correctly predicts the output of future input from the same distribution:

$$f(x) = \mathit{argmax}_y \, d_y(x)$$

- Two fundamental elements in ML methods:
  - Type of "discriminant function" (the model)
  - Type of "loss function" (the training objective)

Classification (coarse) of ML methods

- Nature of the model and loss function:
  - Generative learning (descriptive)
    - Models the probability distribution of data p(x|y), ex: GMM
    - Loss function: Joint likelihood distribution → Maximum Likelihood estimation (MLE) training criteria

    Note: Bayes' rule makes them useful for classification p(y|x) = p(x|y)p(y)
  - Discriminative learning
    - Discriminative models maps directly x to y, ex: MLPs, SVMs, CRFs
    - Discriminative loss function, ex. MCE, MPE, MMI

    Note: Discriminative learning criteria can be used with Generative models

- How training data is used:
  - Supervised – all training samples are labeled
  - Semi-supervised – both labeled and unlabeled
  - Unsupervised – all training samples are unlabeled

Statistical models is speech pattern classification problems

- The most common model in speech pattern recognition problems is the Gaussian Mixture Model (GMM):
  - A GMM is a particular case of Hidden Markov models (HMM) → HMMs also model time

- Many other models have been also used in different speech classification tasks:
  - K-NN – K nearest neighbor
  - MLP – Multi-layer perceptron
  - SVM – Support Vector Machines
  - DNN – Deep neural networks
  - etc.

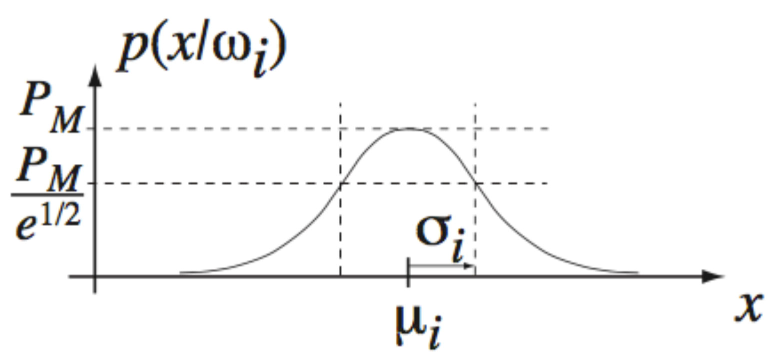
Gaussian mixture models (GMM)

Gaussian models

- Easiest way to model distributions is via parametric model
  - assume known form, estimate a few parameters
- Gaussian model is simple and useful. In 1D

$$p(x \mid \theta_i) = \frac{1}{\sigma_i \sqrt{2\pi}} \exp\left[-\frac{1}{2} \left(\frac{x - \mu_i}{\sigma_i} \right)^2 \right]$$

- Parameters mean μ_i and variance σ_i $\rightarrow$ fit



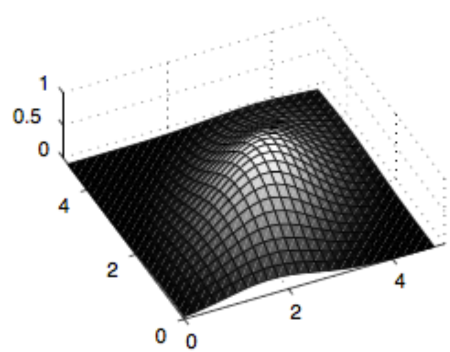
Slide after [1]

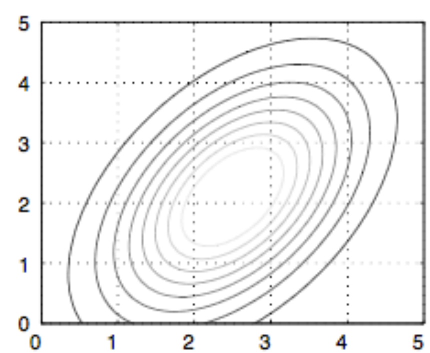
Gaussian mixture models (GMM)

Gaussians in d dimensions

$$p(\mathbf{x} \mid \theta_i) = \frac{1}{(2\pi)^{d/2}|\Sigma_i|^{1/2}} \exp\left[-\frac{1}{2}(\mathbf{x}-\mu_i)^T\Sigma_i^{-1}(\mathbf{x}-\mu_i)\right]$$

Described by a d-dimensional mean μ_i
and a $d \times d$ covariance matrix Σ_i

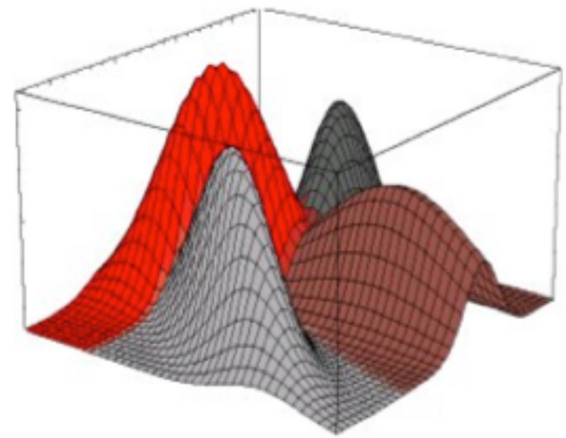




Slide after [1]

Gaussian mixture models (GMM)

Gaussian mixture models



- Single Gaussians cannot model
  - distributions with multiple modes
  - distributions with nonlinear correlations
- What about a weighted sum?

$$p(x) \approx \sum_k c_k p(x \mid \theta_k)$$

  - where $\{c_k\}$ is a set of weights and $\{p(x \mid \theta_k)\}$ is a set of Gaussian components
  - can fit anything given enough components
- Interpretation: each observation is generated by one of the Gaussians, chosen with probability $c_k = p(\theta_k)$

Slide after [1]

Gaussian mixture models (GMM)

In order to use GMMs we need:

1. A method to estimate GMM parameters
   - We use the **Expectation-maximization** (EM) algorithm:
     - General procedure for estimating model parameters
       - Similar for instance to **k-means** used in VQ
     - Iteratively updated model parameters leads to MLE:
       - Can lead to local optimum – depend on initialization
2. Compute the (log-)**likelihood** of a sequence of features given a GMM

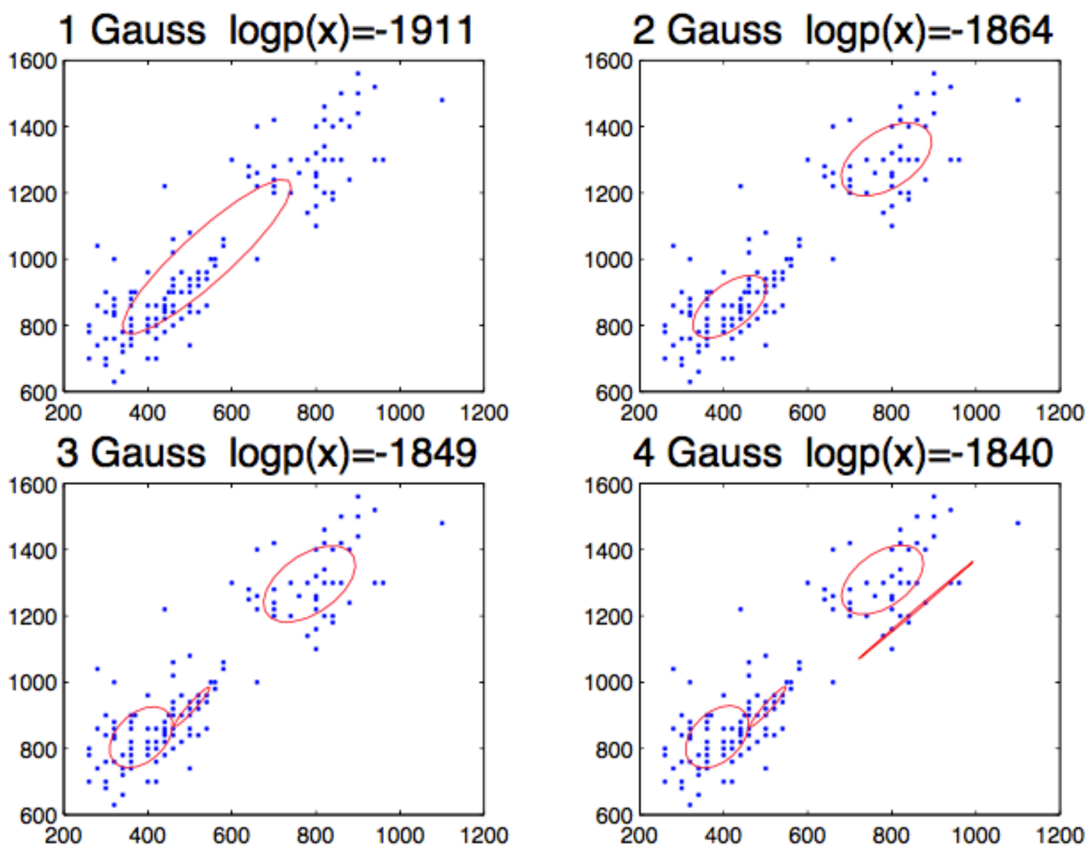
$$\log p(\vec{x}_1, \ldots, \vec{x}_N \mid \lambda) = \sum_{n=1}^{N} \log p(\vec{x}_n \mid \lambda)$$

$$= \sum_{n=1}^{N} \log \left(\sum_{i=1}^{M} p_i b_i(\vec{x}_n) \right)$$

Gaussian mixture models (GMM)

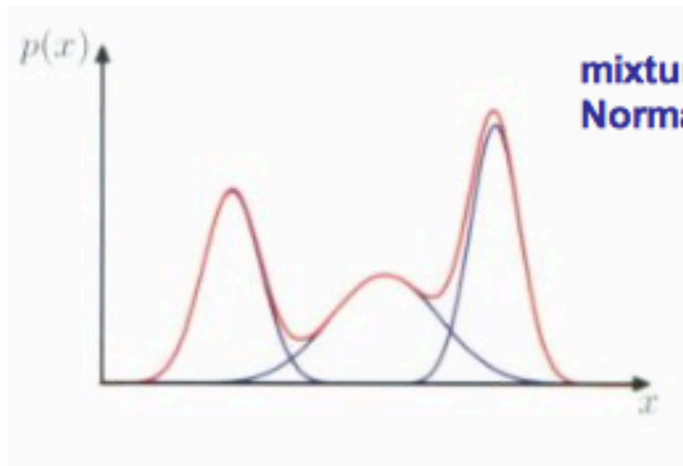
GMM examples

Vowel data fit with different mixture counts



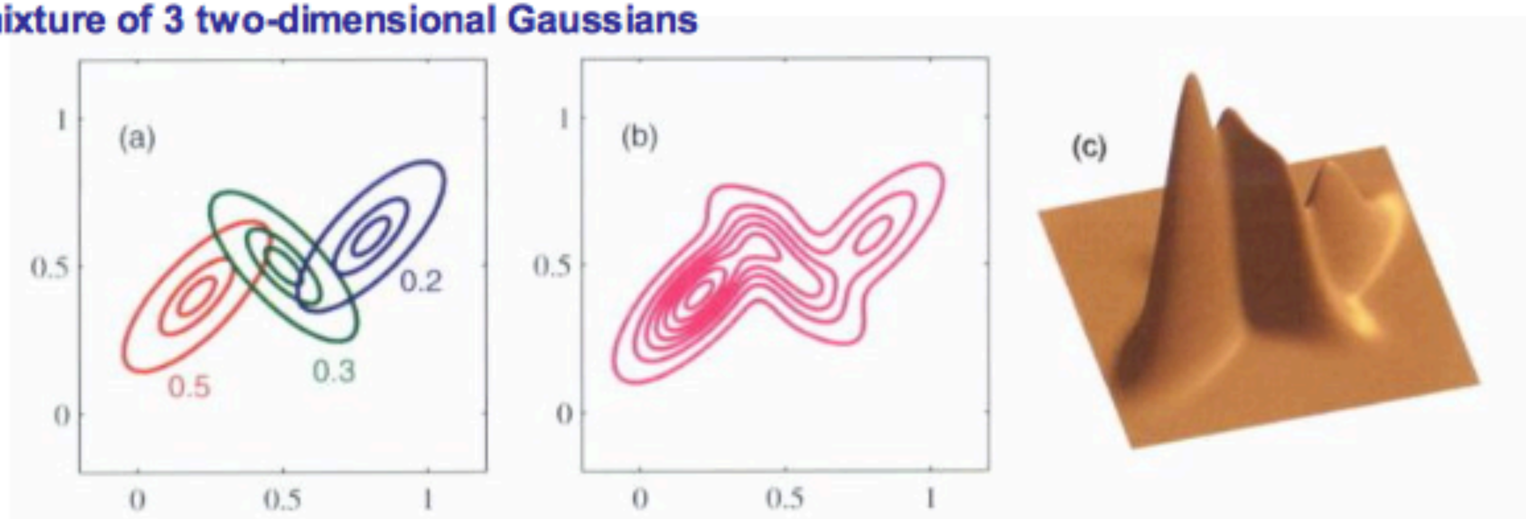
Slide after [1]

Gaussian mixture models (GMM)



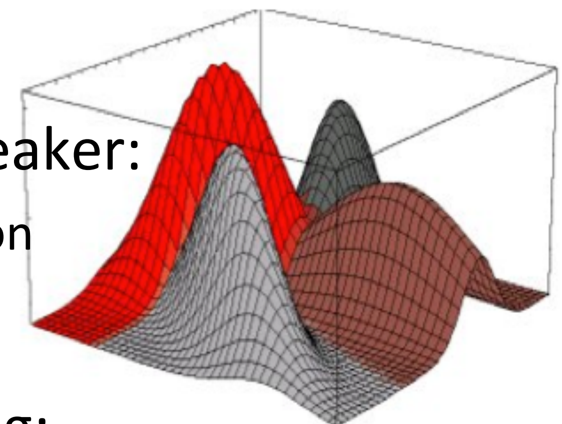
mixture of 3 one-dimensional Normal distributions

mixture of 3 two-dimensional Gaussians



Gaussian mixture models (GMM)
GMM-ML & Speaker Recognition

- Conventional **GMM-ML** approach:
  - In **train** phase:
    - Train a GMM model per target speaker:
      - Apply EM algorithm for ML estimation

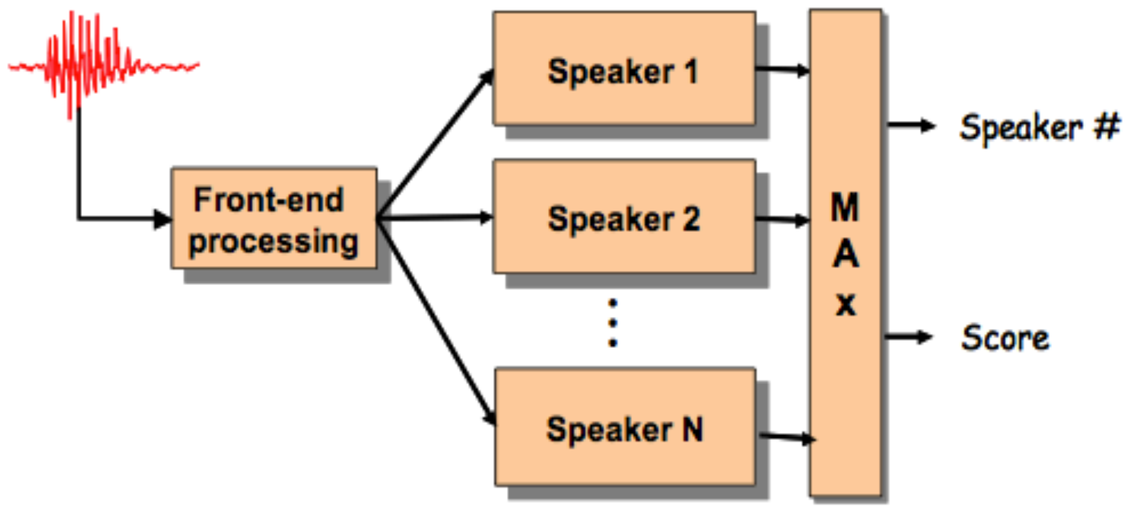


  - In **test** phase:
    - Compute log-likelihoods for scoring:
      - Speaker ID → MAX(LL)
      - Speaker Verification → log-likelihood compared to a threshold or impostor model

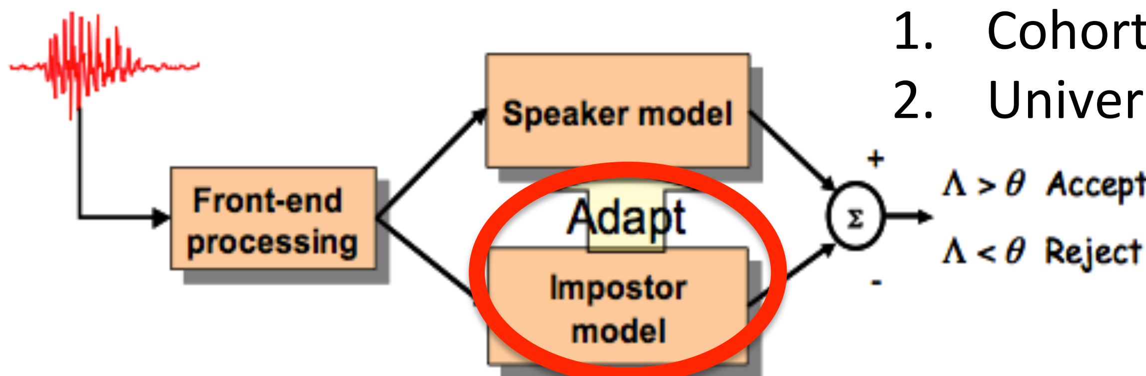
Gaussian mixture models (GMM)

GMM-ML & Speaker Recognition

Identification



Verification



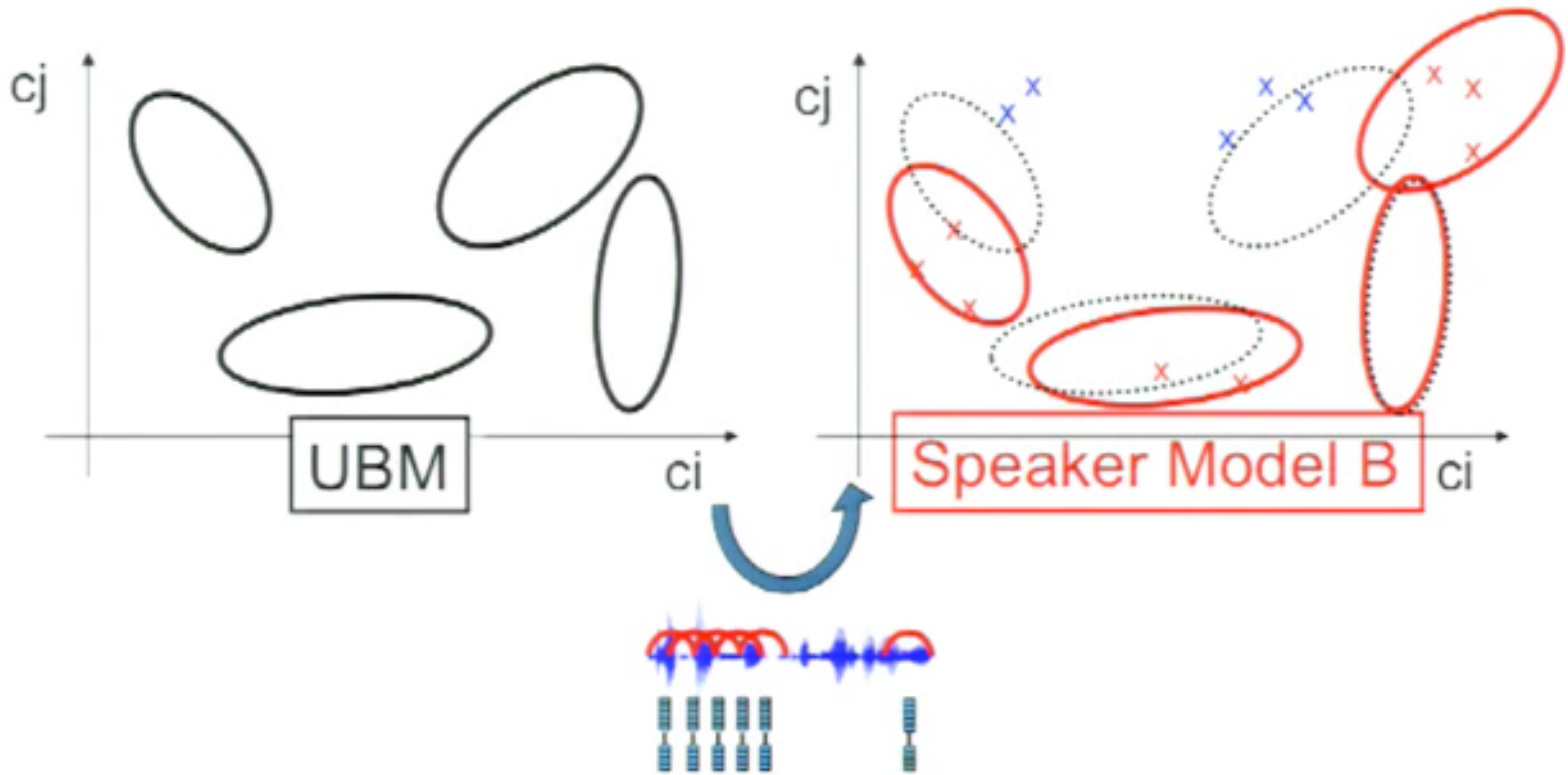
- Impostor model approaches:
  1. Cohort of impostors
  2. Universal model

Slide after [2]

Gaussian mixture models (GMM)
GMM-UBM & Speaker Recognition

- **GMM-UBM** approach:
  - In **train** phase:
    - Estimate the parameters of an UBM (Universal Background Model) with data from different speakers, channels, noise conditions, etc…
    - Adapt the UBM to each one of the target speakers:
      - Use MAP adaptation (usually only-means)
      - MAP "updates" the parameters of the prior model with new "information" obtained from the adaptation data (instead of computing from-the-scratch new model parameters)
  - In **test** phase is like in previous GMM-ML approach.
  - **Advantages**
    - Needs less data,
    - permits updating only seen events,
    - keeps correspondence between means, allows fast scoring (top-M)

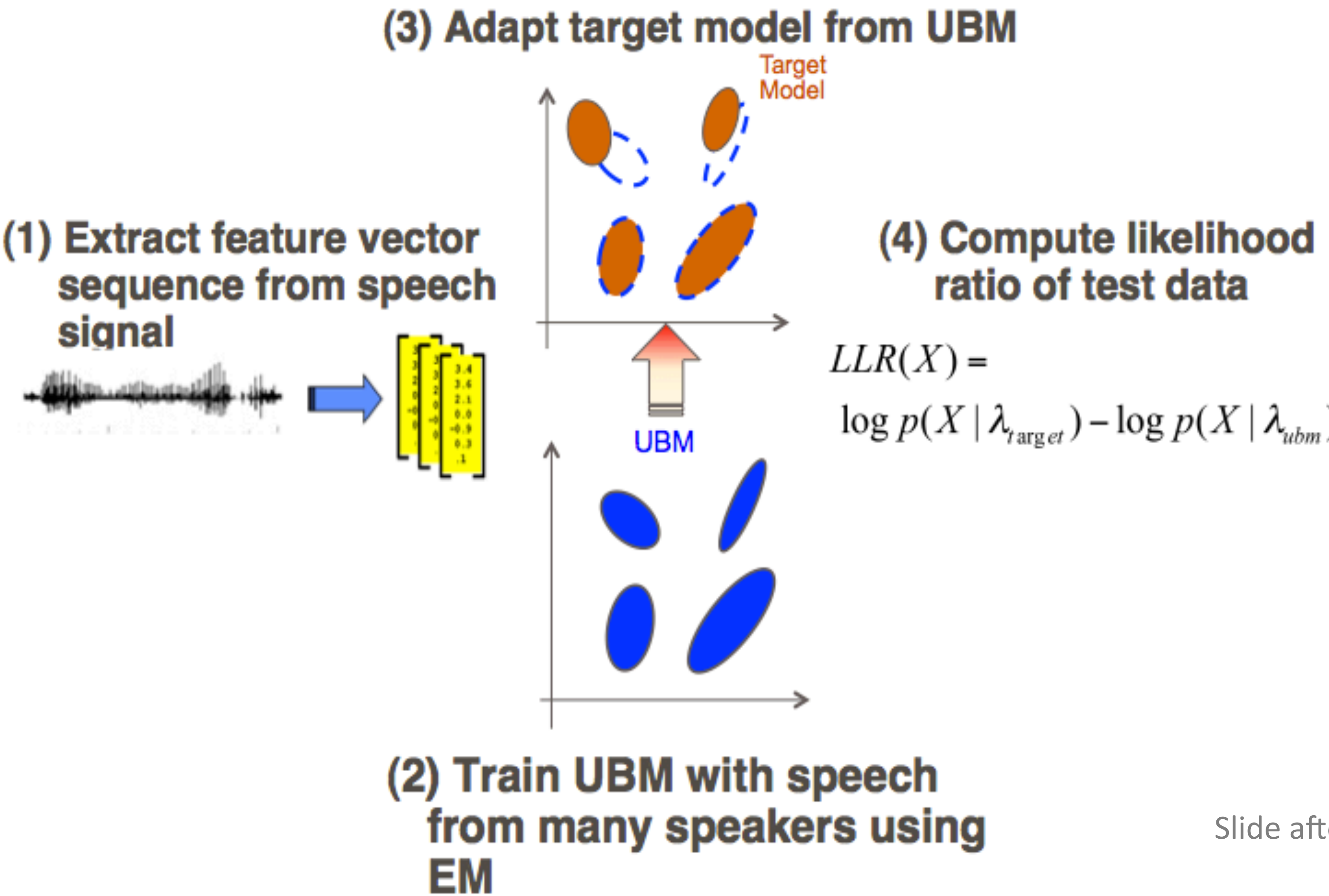
Gaussian mixture models (GMM)
GMM-UBM & Speaker Recognition



Slide after [3]

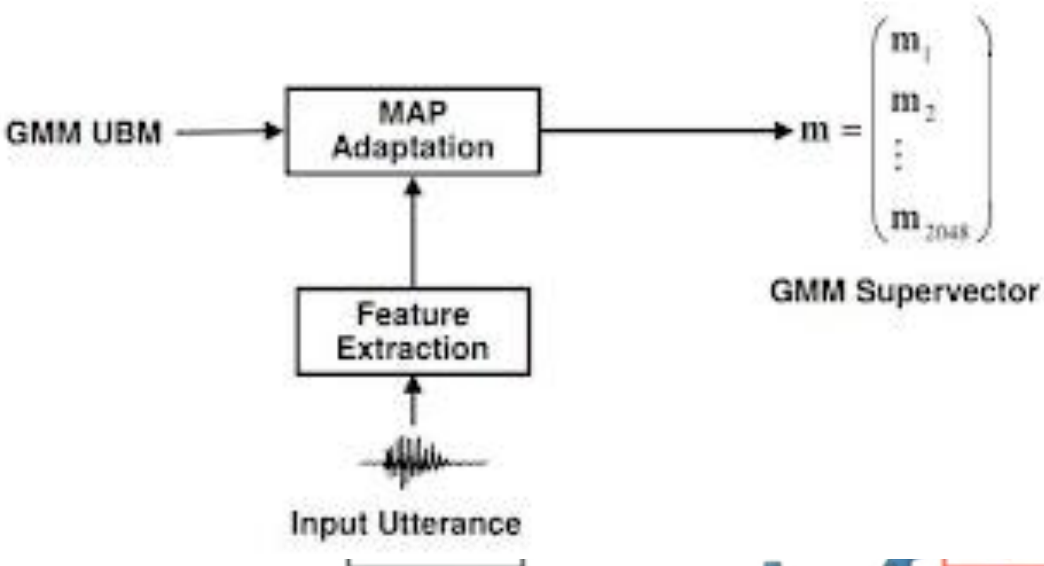
Gaussian mixture models (GMM)

GMM-UBM & Speaker Recognition



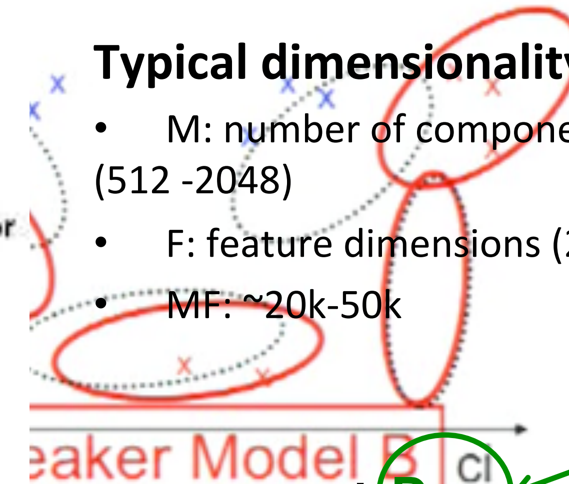
Slide after [2]

Gaussian mixture models (GMM)
GMM-UBM: The supervector concept



Typical dimensionality:

- M: number of components (512 -2048)
- F: feature dimensions (20-60)
- MF: ~20k-50k

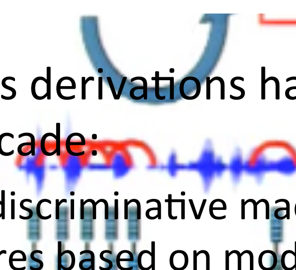


MAP

$$\mathbf{m} = \mathbf{m}_{\mathbf{UBM}} + \mathbf{D}\mathbf{z}_{\mathbf{sh}}$$

D = Full rank diagonal matrix (relevance MAP)

$\mathbf{z}_{\mathbf{sh}}$ = Full rank vector



- The supervector concept and its derivations has had a **huge impact** in in the last decade:

1. As a kind of feature extraction for discriminative machine learning methods → REMEMBER features based on models!?
2. As a tool for Factor Analysis derivation

Gaussian mixture models (GMM)
Factor Analysis approaches: The i-vector

Factor Analysis (FA) is a statistical method for investigating if a number of variables are linearly related to a small number of unobservable factors.

GMM-UBM (MAP) → $\mathbf{m} = \mathbf{m}_{\mathbf{UBM}} + \mathbf{D}\mathbf{z}_{\mathbf{sh}}$

- **D** diagonal full-rank
- $\mathbf{z}_{\mathbf{sh}}$: speaker (and more) component

i-vectors → $\mathbf{m} = \mathbf{m}_{\mathbf{UBM}} + \mathbf{T}\mathbf{w}$

- **T** total variability subspace (low-rank)
- **w** variability (loading) factors, a.k.a i-vectors
  - ~400-600 dimensions
  - They contain all speaker and channel variability
  - It is used as a low-dimensional representation (on top of them other models can be trained)

Example of discriminative model

Support Vector Machines (SVMs)

Slides after

Miguel Bugalho, “Support Vector Machines (SVMs) Classifiers: Introduction and Application. Case Study: VidiVideo Audio Event Detection”

SVM – Basic formulation

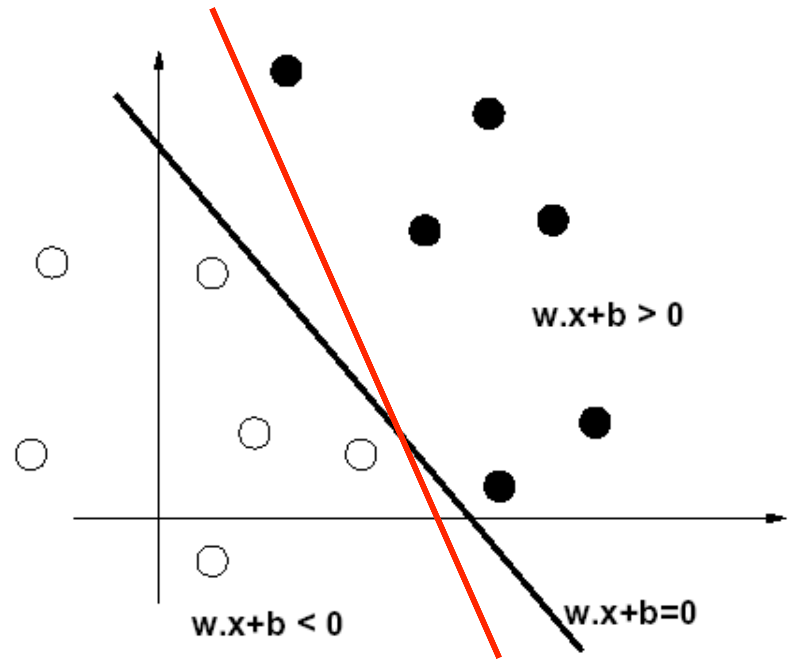
- Linear classifier (linear combination of features)
- Hyperplane equation

$$\vec{w}.\vec{x}+b=0$$

- Class is determined by the sign of

$$\vec{w}.\vec{x}+b$$

- Non-probabilistic binary classifier

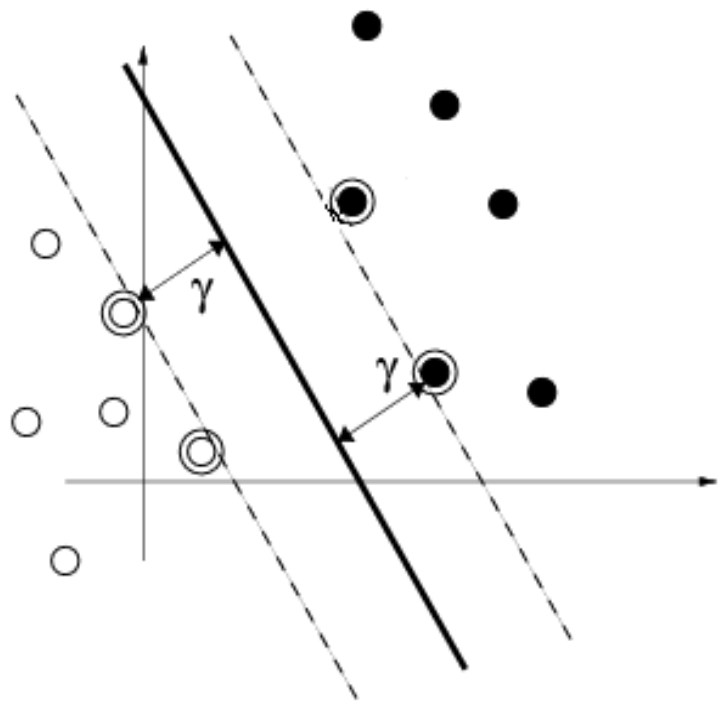


SVM - maximum-margin hyperplane

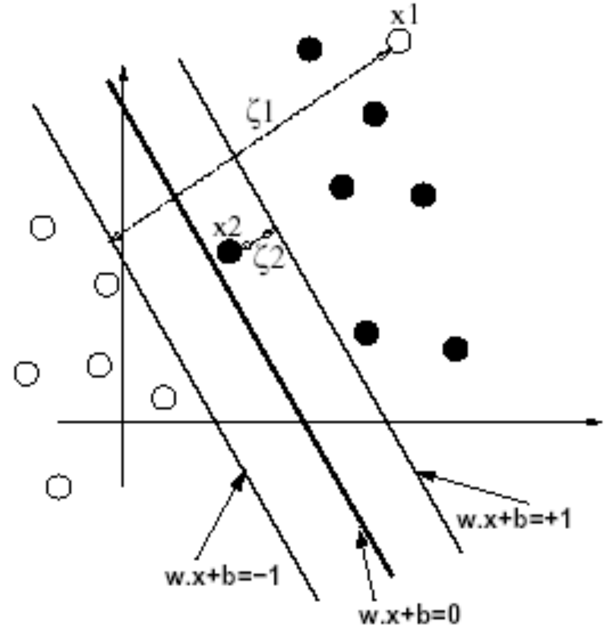
- Margin between both hyperplanes

$$\left.\begin{array}{l} \vec{w}.\vec{x}_i + b = 1 \\ \vec{w}.\vec{x}_i + b = -1 \end{array}\right\} y_i(\vec{w}.\vec{x}_i + b) \geq 1$$

- The max margin hyperplane is determined by those x_i which lie nearest to it →Support Vectors



SVM - Minimization



- Minimize

$$\| \vec{w} \|^2 + C \sum_{i=1}^{N} \varsigma_i(\vec{w}, b)$$

$$\varsigma_i(\vec{w}, b) = \begin{cases} 0, & \text{if } y_i(\vec{w}.\vec{x} + b) \geq 1 \\ 1 - y_i(\vec{w}.\vec{x} + b), & \text{if } y_i(\vec{w}.\vec{x} + b) < 1 \end{cases}$$

SVM – Support Vectors

- The hyperplane can be calculated using only a linear combination of the support vectors

$$\vec{w}^{*} = \sum_{x_i \in VS} \lambda_i^{*} y_i \vec{x}_i$$

- The parameter λ_i^{*} has to be estimated by the minimization procedure
- The parameter b also needs to be estimated

SVM - Classifying

- A new observation can be classified using the dot product of the support vectors and the new example:

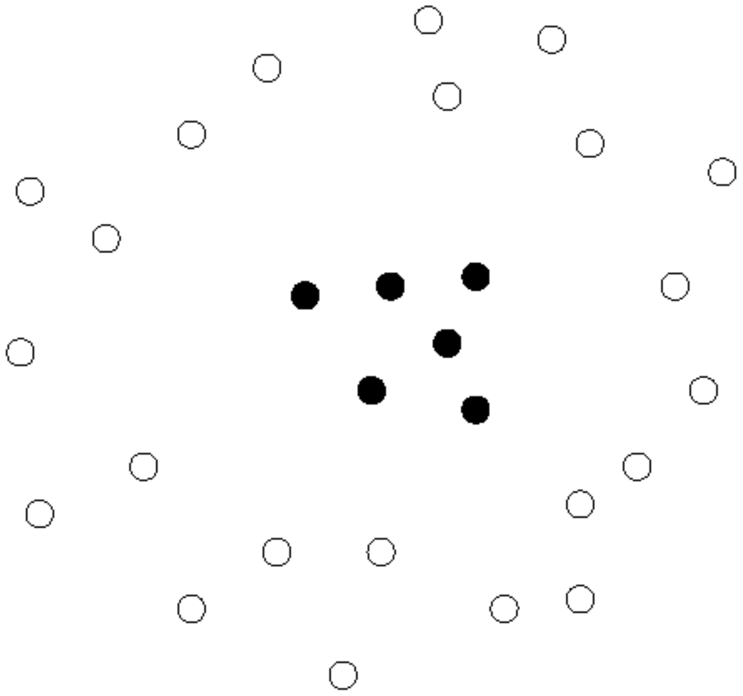
$$\vec{w}^{*} . \vec{x} + b = \sum_{x_i \in VS} \lambda_i^{*} y_i \vec{x}_i . \vec{x} + b^{*}$$

- The dot product can be replaced by kernels
- Kernels allow to transform the initial space to a new space where the examples are linearly separable

SVM – Non Linear Space

- When the examples are not linearly separable, a kernel may be used transform the initial space

$$K(\vec{x},\vec{x'}) = \phi(\vec{x}).\phi(\vec{x'})$$

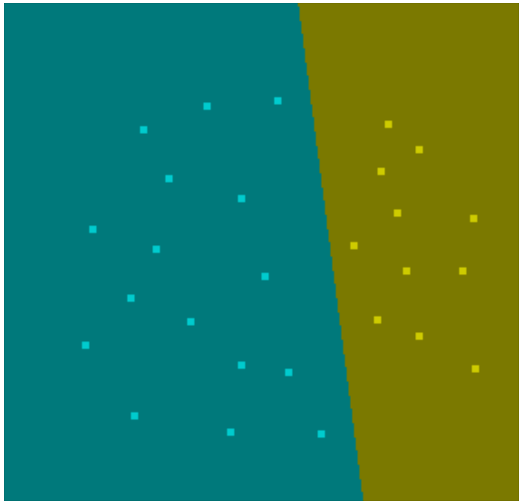


SVM – Basic Kernels

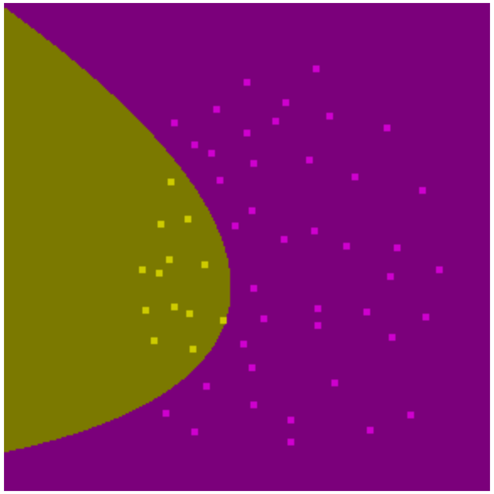
- Linear Kernel – Corresponds to the dot product in the previously presented expression
- Polynomial Kernel $K(\vec{x},\vec{x}') = (\gamma \vec{x}.\vec{x}' + c)^d$
  - Where d is the degree of the polynomial. c and γ are constants
- Radial Basis Kernel $K(\vec{x},\vec{x}') = \exp(-\gamma \|\vec{x} - \vec{x}'\|^2)$
  - Where γ defines the "size" of the radial basis function

SVM – Kernel Examples

- http://www.csie.ntu.edu.tw/~cjlin/libsvm/



Linear

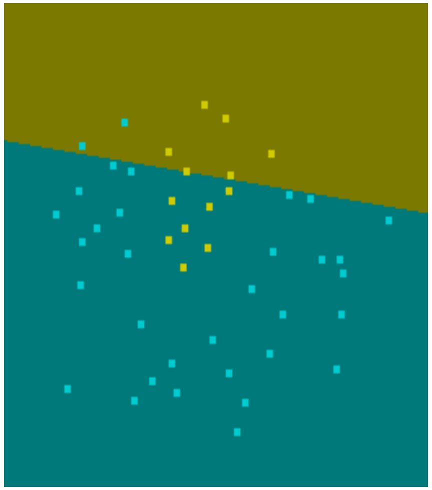


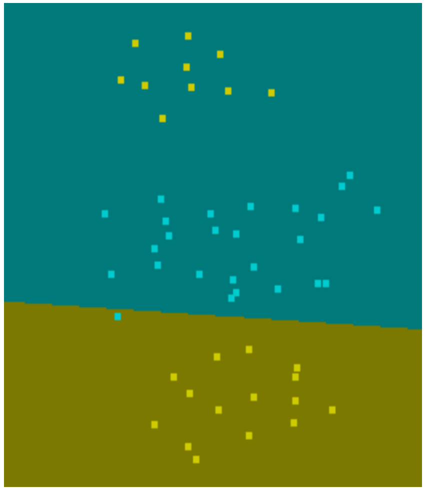
Polynomial
Degree=2

Radial

SVM – Kernel Advantages / Disadvantages (1/3)

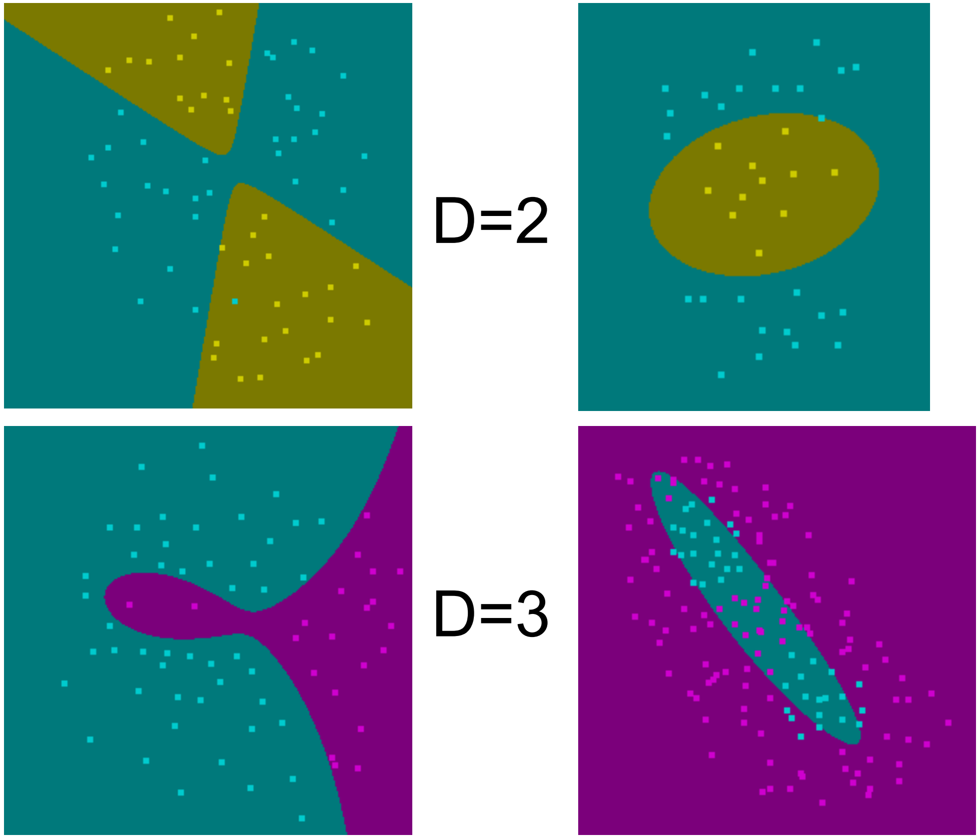
- Linear Kernel
- Advantage
  - is faster to calculate and less prune to overfitting
- Disadvantage
  - If the data is not linearly separable (can’t learn)
  - High dimension data is easier to separate
  - Complex data is harder





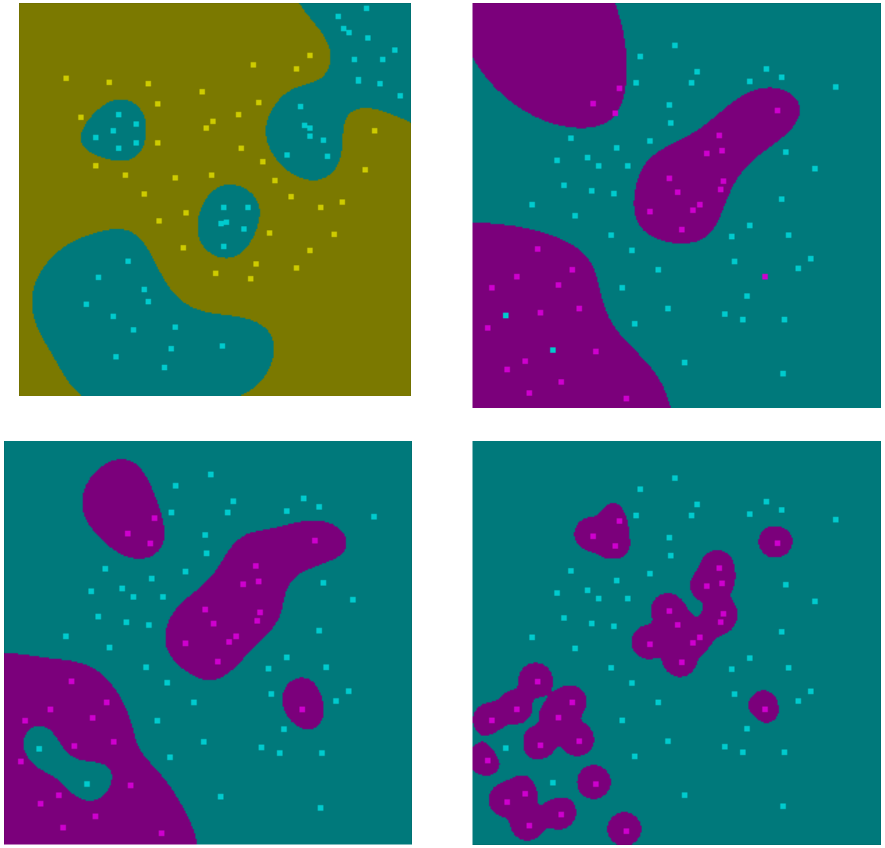
SVM – Kernel Advantages / Disadvantages (2/3)

- Polinomial Kernel
- Advantage
  - Higher power to separate data
- Disadvantage
  - Can have overfitting problems, specially with high degree polynomials
  - Still some data that can’t be separated



SVM – Kernel Advantages / Disadvantages (3/3)

- Radial Kernel
- Advantage
  - In the limit it can separate any data
- Disadvantage
  - Used without caution causes many overfitting problems



SVM - Advantages

- Easy to use
  - Few parameters to test.
    - The default parameters work for most problems, though testing some parameters with a simple cross validation can give extra precision
- Works with limited data
  - SVMs are used in applications with few data (ex: medical data)
    - Calculating the maximum margin is usually a good extrapolation
- It can separate any type of data
  - In the limit radial kernels separate any data (watch for overfitting)
- Is robust to overfitting if some precautions are taken
  - Optimize the parameters with a different data set or cross validation

Brief HOW-TO: Building a classifier

- Define task and classes
  - Need a labeled training data set
- Define feature space
  - Use meaningful features, disregard useless info
  - Prepare data (some ML methods are very sensible to scale, range, etc.)
- Define decision algorithm
  - Choose the right tool for the right job
    - The literature is full of examples
  - Avoid over-fitting (too complex model for few data):
    - Need a development data set
    - If no possible, cross-validation
- Measure performance
  - Use a separate evaluation data set

Tools for speech modeling

GMM

- SPEAR: A Speaker Recognition Toolkit based on Bob (Python)

https://pythonhosted.org/bob.bio.spear/

- MATLAB - Statistics and Machine Learning Toolbox

http://www.mathworks.com/help/stats/fitgmdist.html

SVM

- LIBSVM -- A Library for Support Vector Machines

https://www.csie.ntu.edu.tw/~cjlin/libsvm/

- Weka 3: Data Mining Software in Java (Collection of ML tools)

http://www.cs.waikato.ac.nz/ml/weka/

NEURAL NETWORKS

- Neural Network Toolbox

http://www.mathworks.com/help/nnet/index.html

- QuickNet

http://www1.icsi.berkeley.edu/Speech/qn.html

References

- These are some presentations that were used for this course:

[1] Michael Mandel, "Lecture 3: Machine learning, classification, and generative models"

http://www.ee.columbia.edu/~dpwe/e6820/lectures/L03-ml.pdf

[2] Douglas A. Reynolds, "Overview of Automatic Speaker Recognition" http://www.fit.vutbr.cz/study/courses/SRE/public/prednasky/2009-10/07_spkid_doug/sid_tutorial.pdf

[3] Javier González-Domínguez, "Session Variability Compensation in Speaker Recognition" http://tv.uvigo.es/matterhorn/20022

[4] Miguel Bugalho, "Support Vector Machines (SVMs) Classifiers: Introduction and Application. Case Study: VidiVideo Audio Event Detection"