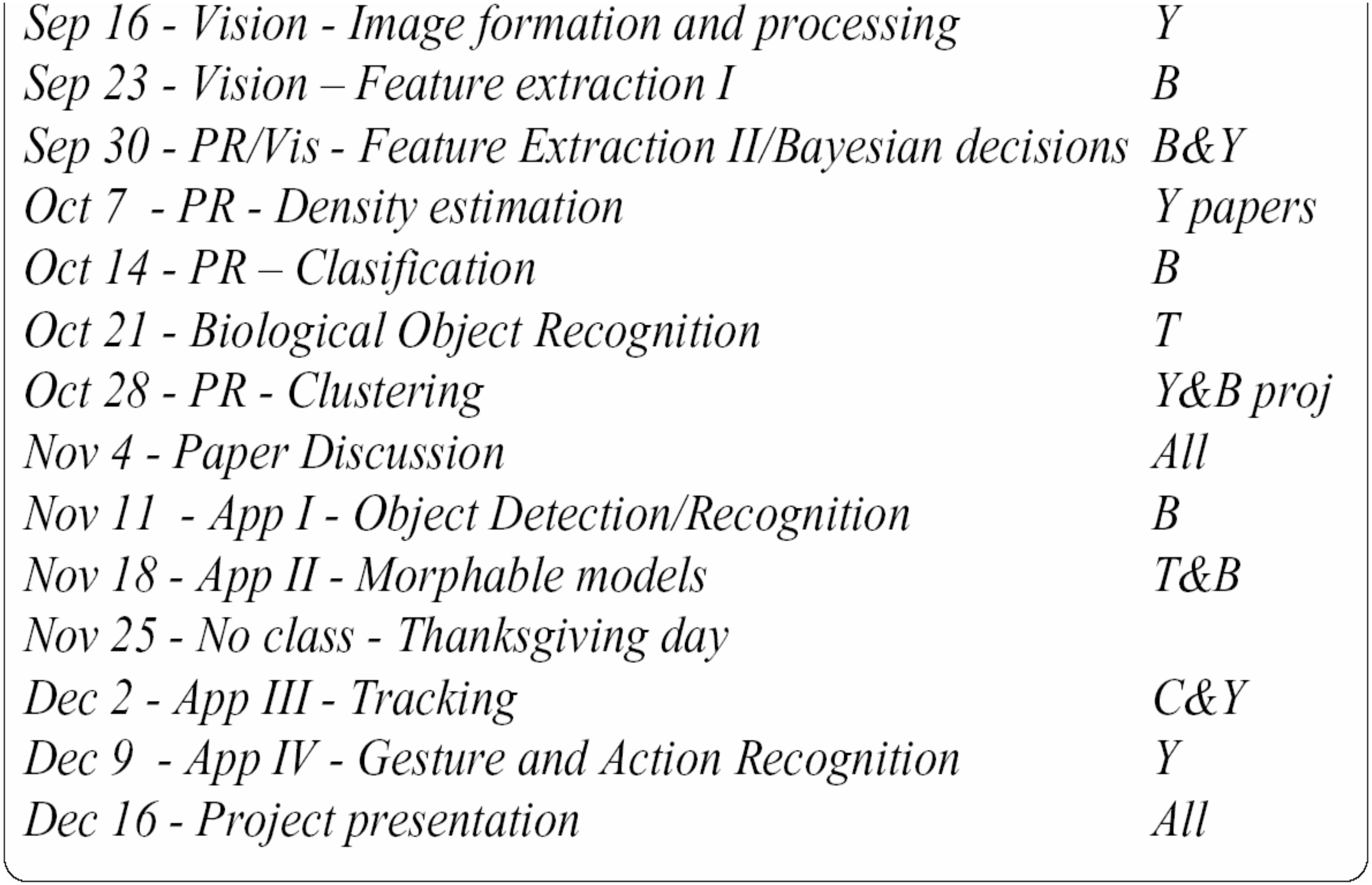
New Schedule



	Sep 16 - Vision - Image formation and processing	*Y*
	Sep 23 - Vision – Feature extraction I	*B*
	Sep 30 - PR/Vis - Feature Extraction II/Bayesian decisions	*B&Y*
	Oct 7 - PR - Density estimation	*Y papers*
	Oct 14 - PR – Clasification	*B*
Dec. 8	*Oct 21 - Biological Object Recognition*	*T*
same	*Oct 28 - PR - Clustering*	*Y&B proj*
same	*Nov 4 - Paper Discussion*	*All*
same	*Nov 11 - App I - Object Detection/Recognition*	*B*
Oct. 21	*Nov 18 - App II - Morphable models*	*T&B*
	Nov 25 - No class - Thanksgiving day	
^2 weeks	*Dec 2 - App III - Tracking*	*C&Y*
^1 week	*Dec 9 - App IV - Gesture and Action Recognition*	*Y*
^1 week	*Dec 16 - Project presentation*	*All*

9.913 Pattern Recognition for Vision

Classification

Bernd Heisele

Overview

Introduction

Linear Discriminant Analysis

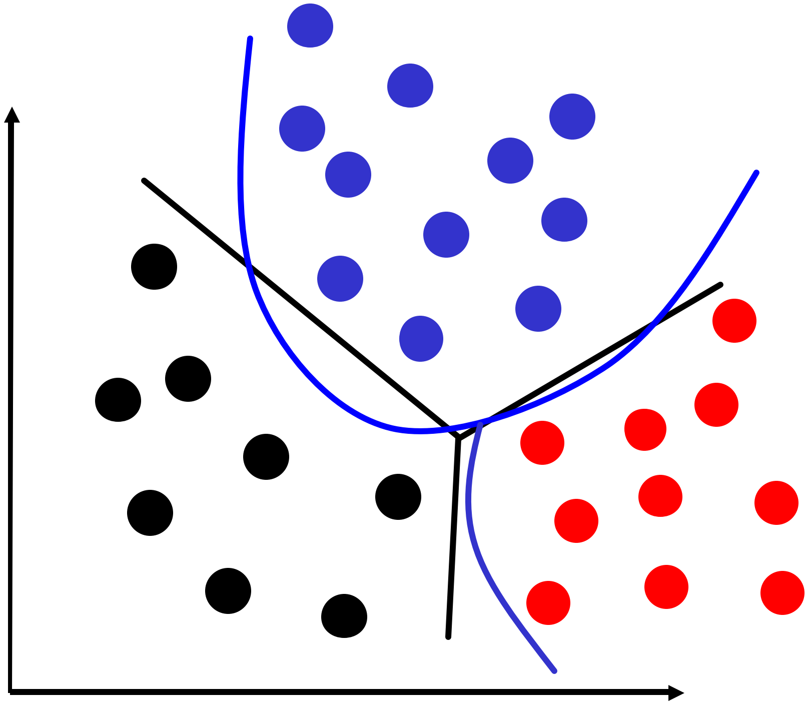
Support Vector Machines

Literature & Homework

Introduction

Classification

- Linear, non-linear separation
- Two class, multi-class problems



Two approaches:

- Density estimation, classify with Bayes decision: Linear Discr. Analysis (LDA), Quadratic Discr. Analysis (QDA)
- Without density estimation: Support Vector Machines (SVM)

LDA Bayes Rule

Bayes Rule

$$p(\mathbf{x},w) = p(\mathbf{x}\,|\,w)P(w) = P(w\,|\,\mathbf{x})\,p(\mathbf{x}) \quad \Rightarrow$$

$$\boxed{\underbrace{P(w\,|\,\mathbf{x})}_{\textit{posterior}} = \frac{\overbrace{p(\mathbf{x}\,|\,w)}^{\textit{likelihood}}\overbrace{P(w)}^{\textit{prior}}}{\underbrace{p(\mathbf{x})}_{\textit{evidence}}}}$$

$\mathbf{x}$: random vector

w : class

LDA— Bayes Decision Rule

$$\text{Decide } w_1 \text{ if } \frac{P(w_1 \mid \mathbf{x})}{P(w_2 \mid \mathbf{x})} > 1; \text{ otherwise } w_2$$

Likelihood Ratio

$$\frac{p(\mathbf{x} \mid w_1)P(w_1)}{p(\mathbf{x} \mid w_2)P(w_2)} > 1$$

Log Likelihood Ratio

$$\ln\left(\frac{p(\mathbf{x} \mid w_1)P(w_1)}{p(\mathbf{x} \mid w_2)P(w_2)}\right) > 0, \ln\left(\frac{p(\mathbf{x} \mid w_1)}{p(\mathbf{x} \mid w_2)}\right) + \ln\left(\frac{P(w_1)}{P(w_2)}\right) > 0$$

LDA—Two Classes, Identical Covariance

$$\text{Gaussian: } p(\mathbf{x}\,|\,w_i) = \frac{1}{(2p)^{d/2}\,|\,\Sigma_i\,|^{1/2}}\,e^{-\frac{1}{2}(\mathbf{x}-m_i)^T\Sigma_i^{-1}(\mathbf{x}-m_i)}$$

assume identical covariance matrices $\Sigma_1 = \Sigma_2$:

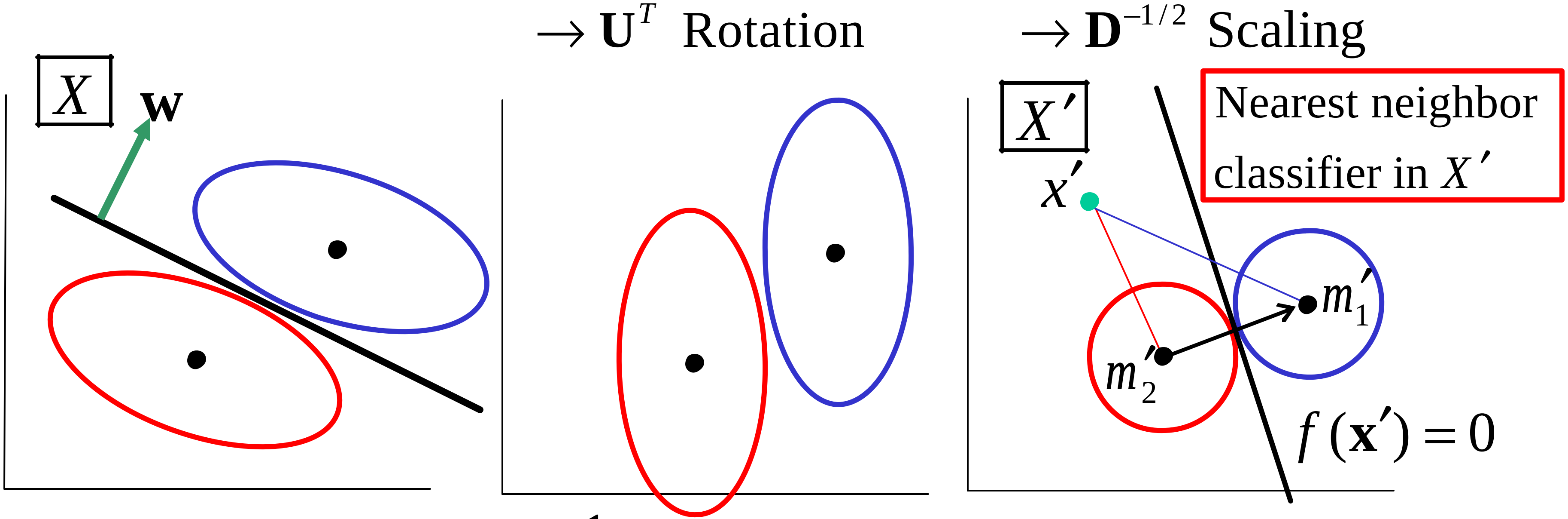
$$\ln\left(\frac{p(\mathbf{x}\,|\,w_1)}{p(\mathbf{x}\,|\,w_2)}\right) + \ln\left(\frac{P(w_1)}{P(w_2)}\right)$$

$$= \frac{1}{2}(\mathbf{x}-m_2)^T\,\mathbf{S}^{-1}(\mathbf{x}-m_2) - \frac{1}{2}(\mathbf{x}-m_1)^T\,\mathbf{S}^{-1}(\mathbf{x}-m_1) + \ln\left(\frac{P(w_1)}{P(w_2)}\right)$$

$$= \mathbf{x}^T\,\underbrace{\mathbf{S}^{-1}(m_1-m_2)}_{\mathbf{w}} + \underbrace{\frac{1}{2}(m_1+m_2)^T\,\mathbf{S}^{-1}(m_2-m_1) + \ln\left(\frac{P(w_1)}{P(w_2)}\right)}_{b}$$

$= \mathbf{x}^T\mathbf{w} + b$ linear decision function: w_1 if $\mathbf{x}^T\mathbf{w} + b > 0$

LDA—Two Classes, Identical Covariance



$\rightarrow \mathbf{U}^T$ Rotation

$\rightarrow \mathbf{D}^{-1/2}$ Scaling

Nearest neighbor classifier in X'

$$f(\mathbf{x}) = \mathbf{x}^T \mathbf{S}^{-1}(m_1 - m_2) + \frac{1}{2}(m_1 + m_2)^T \mathbf{S}^{-1}(m_2 - m_1) + \ln(P(w_1)/P(w_2))$$

$$= \mathbf{x}'^T(m_1' - m_2') + \underbrace{\frac{1}{2}(m_1' + m_2')^T(m_2' - m_1') + \ln(P(w_1)/P(w_2))}_{b},$$

$$\Gamma^T\Gamma = \mathbf{S}^{-1}, \mathbf{x}'^T = \mathbf{x}^T\Gamma^T, \mathbf{x}' = \Gamma\mathbf{x}, m_1' - m_2' = \Gamma(m_1 - m_2)$$

$$\mathbf{S} = \mathbf{UDU}^T, \ \mathbf{S}^{-1} = \mathbf{UD}^{-1}\mathbf{U}^T = \Gamma^T\Gamma, \Gamma = \mathbf{D}^{-1/2}\mathbf{U}^T$$

LDA—Computation

$$\left.\begin{aligned} \hat{m}_i &= \frac{1}{N_i}\sum_{n=1}^{N_i}\mathbf{x}_{i,n} \\ \hat{\mathbf{S}} &= \frac{1}{N_1+N_2}\sum_{i=1}^{2}\sum_{n=1}^{N_i}(\mathbf{x}_{i,n}-\hat{m}_i)(\mathbf{x}_{i,n}-\hat{m}_i)^T \end{aligned}\right\}\text{Density estimation}$$

$$f(\mathbf{x}) = sign(\mathbf{x}^T\mathbf{w}+b)$$

$$\mathbf{w} = \hat{\mathbf{S}}^{-1}(\hat{m}_1-\hat{m}_2)$$

$$b = \frac{1}{2}(\hat{m}_1+\hat{m}_2)^T\mathbf{S}^{-1}(\hat{m}_2-\hat{m}_1)+\ln\underbrace{\left(\frac{P(w_1)}{P(w_2)}\right)}_{\text{Approximate by}\frac{N_1}{N_2}}$$

QDA—Two classes, different covariance matrix

Quadratic Discriminant Analysis

decide w_1 if $f(\mathbf{x}) > 0$

$$f(\mathbf{x}) = \ln\left(p(\mathbf{x} \mid w_1)\right) + \ln(P(w_1)) - \ln\left(p(\mathbf{x} \mid w_2)\right) - \ln\left(P(w_2)\right)$$

$$\ln\left(p(\mathbf{x} \mid w_1)\right) = -\frac{1}{2}\ln\left|\Sigma_1\right| - \frac{1}{2}(\mathbf{x} - m_1)^T \Sigma_1^{-1}(\mathbf{x} - m_1) + \ln P(w_1)$$

$$f(\mathbf{x}) = \mathbf{x}^T \mathbf{A}\mathbf{x} + \mathbf{w}^T \mathbf{x} + w_0 \ - \mathit{quadratic}$$

where $\mathbf{A} = -\frac{1}{2}\left(\Sigma_1^{-1} - \Sigma_2^{-1}\right)$ - a matrix

$\mathbf{w} = \Sigma_1^{-1} m_1 - \Sigma_2^{-1} m_2$ - a vector

$w_o = \ldots$ well, the rest of it - a scalar

LDA Multiclass, Identical Covariance

Find the linear decision boundaries for k classes:

For two classes we have:

$f(\mathbf{x}) = \mathbf{x}^T \mathbf{w} + b$ decide w_1 if $f(\mathbf{x}) > 0$

In the multi-class case we have k -1 decision functions:

$$f_{1,2}(\mathbf{x}) = \mathbf{x}^T \mathbf{w}_{1,2} + b_{1,2},$$

$$f_{1,3}(\mathbf{x}) = \mathbf{x}^T \mathbf{w}_{1,3} + b_{1,3},$$

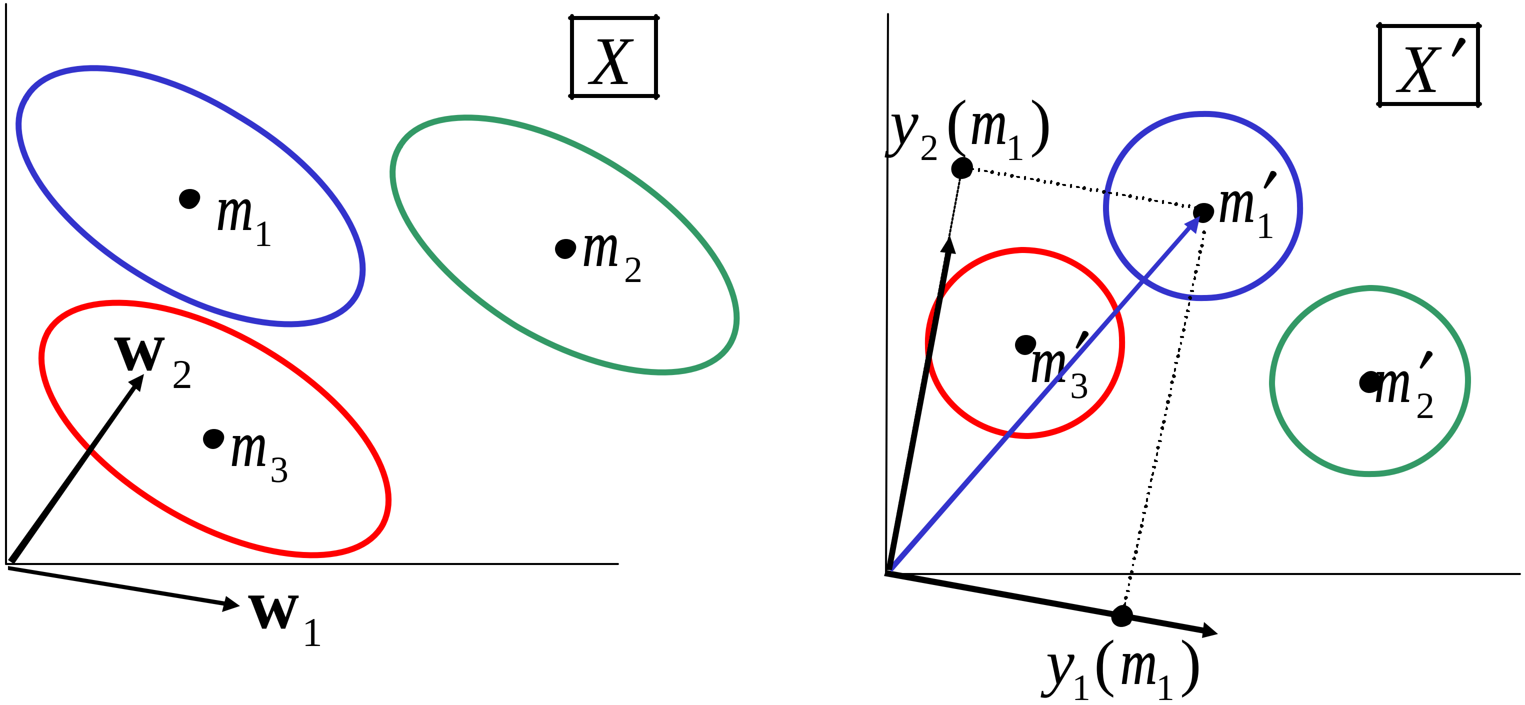
$$\vdots$$

$$f_{1,k}(\mathbf{x}) = \mathbf{x}^T \mathbf{w}_{1,k} + b_{1,k}$$

$\Rightarrow$ we have to determine $(k-1)(p+1)$ parameters,

p is dimension of $\mathbf{x}$

LDA Multiclass, Identical Covariance



Find the $n-$dimensional subspace that gives the best linear discrimination betw. the k classes.

$$\mathbf{y} = (\mathbf{w}_1 \mid \mathbf{w}_2 \mid \ldots \mid \mathbf{w}_n)^T \mathbf{x}$$

also known as Fisher Linear Discriminant

LDA—Multiclass, Identical Covariance

Computation

compute the $d \times k$ matrix $\mathbf{M} = (m_1 | m_2 | ... | m_k)$ and cov. matrix $\mathbf{S}$

compute the m': $\mathbf{M}' = \Gamma \mathbf{M}$, $\Sigma^{-1} = \Gamma^T \Gamma$

compute the cov. matrix $\mathbf{B}'$ of m'

compute the eigenvectors $\mathbf{v}'_i$ of $\mathbf{B}'$ ranked by eigenvalues

calculate $\mathbf{y}$ by projecting $\mathbf{x}$ into X' and then onto the eigenvector:

$$y_i = \mathbf{v}_i'^T \Gamma \mathbf{x} \Rightarrow \mathbf{w}_i = \Gamma^T \mathbf{v}'_i$$

LDA—Fisher's Approach

Find $\mathbf{w}$ such that the ratio of between-class and in-class variance is maximized if the data is projected onto $\mathbf{w}$:

$y = \mathbf{w}^T \mathbf{x}$

$\max \dfrac{\mathbf{w}^T \mathbf{B} \mathbf{w}}{\mathbf{w}^T \mathbf{S} \mathbf{w}}, \quad \mathbf{B} = \mathbf{M}\mathbf{M}^T$ the covariance of the m's

can be written as:

$\max \mathbf{w}^T \mathbf{B} \mathbf{w}$ subject to $\mathbf{w}^T \mathbf{S} \mathbf{w} = 1$

generalized eigenvalue problem,

solution are the ranked eigenvectors of $\mathbf{S}^{-1}\mathbf{B}$

...same is an previous derivation.

The Coffee Problem: LDA vs. PCA

Image removed due to copyright considerations. See: R. Gutierrez-Osuna
http://research.cs.tamu.edu/prism/lectures/pr/pr_l10.pdf

LDA/QDA—Summary

Advantages:

- LDA is the Bayes classifier for multivariate Gaussian distributions with common covariance.
- LDA creates linear boundaries which are simple to compute.
- LDA can be used for representing multi-class data in low dimensions.

- QDA is the Bayes classifier for multivariate Gaussian distributions.
- QDA creates quadratic boundaries.

Problems:

- LDA is based on a single prototype per class (class center) which is often insufficient in practice.

Variants of LDA

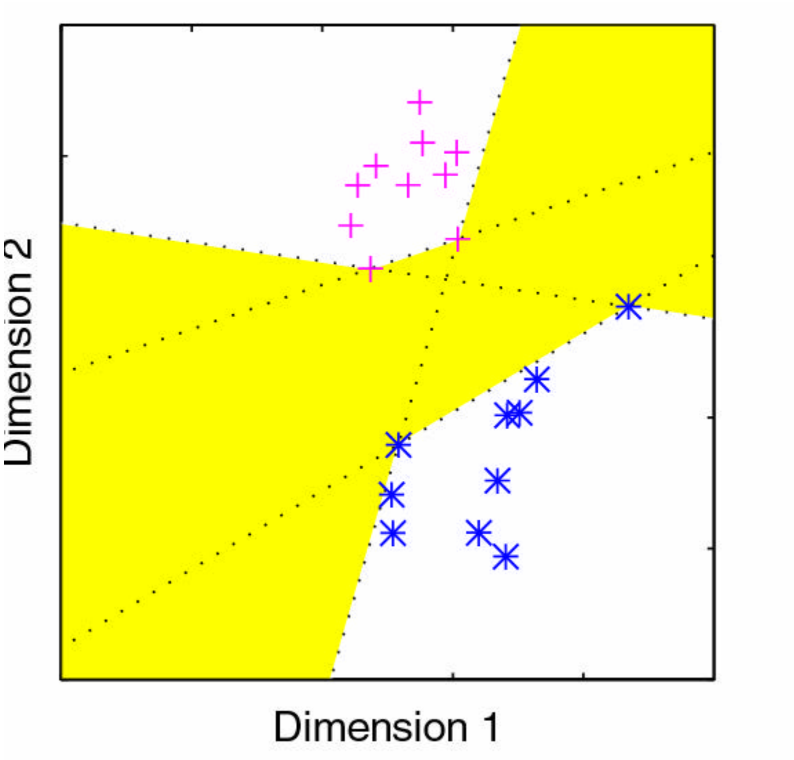
Nonparameteric LDA (Fukunaga)
removes the unimodal assumption by the scatter matrix using local information.
More than *k*-1 features can be extracted.

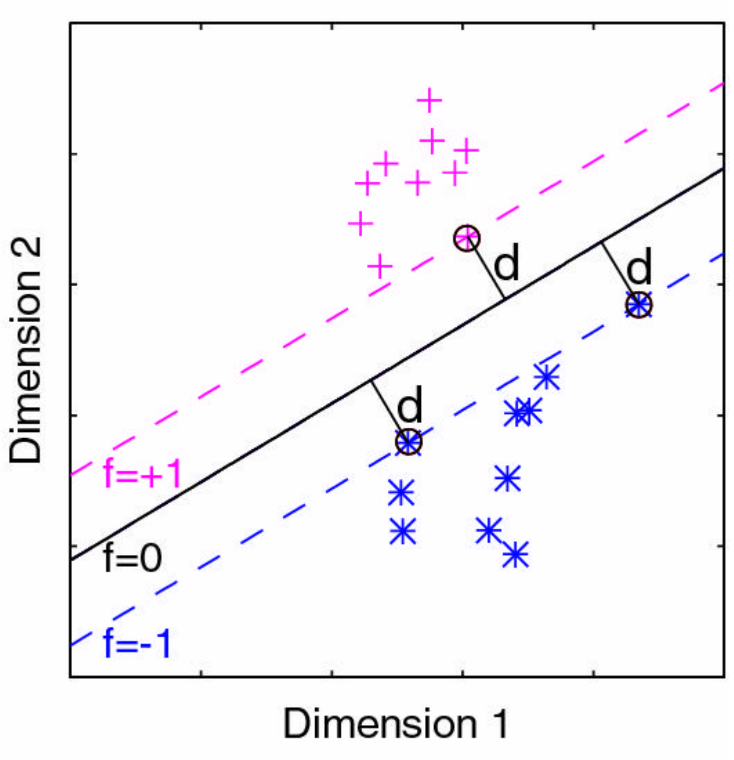
Orthonormal LDA (Okada&Tomita) computes projections that maximize separability and are pair-wise orthonormal.

Generalized LDA (Lowe)
Incorporates a cost function similar to Bayes Risk minimization.

….and many many more (see "Elements of Statistical Learning" Hastie, Tibshirani, Friedman)

SVM—Linear, Separable (LS)





Find the separating function $f(\mathbf{x}) = \operatorname{sign}(\mathbf{x}_i^T \mathbf{w} + b)$

which maximizes the margin $M = 2d$ on the training data.

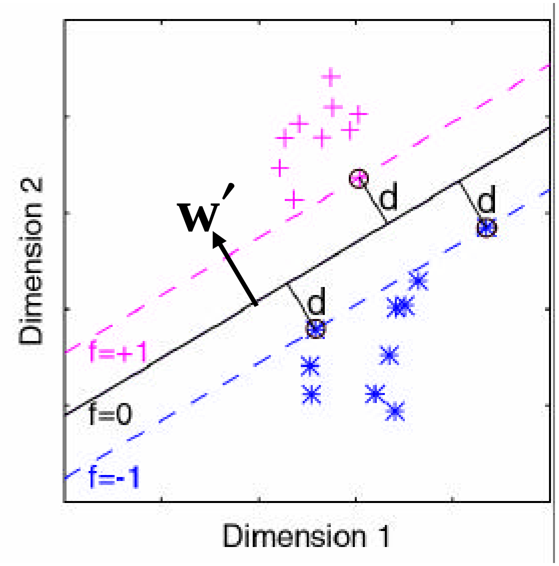
$\Rightarrow$ Maximum margin classifier.

SVM—Primal, (LS)

Training data consists of N pairs $\{(\mathbf{x}_1, y_1),\ldots,(\mathbf{x}_N, y_N)\}, y_i \in \{\text{-}1,1\}$.

The problem of maximizing the margin $2d$ can be formulated as:

$$\max_{\mathbf{w}',b} 2d \quad \text{subject to} \begin{cases} y_i(\mathbf{x}_i^T\mathbf{w}' + b') \geq d \\ \|\mathbf{w}'\|^2 = 1 \end{cases}$$



or alternatively: $\mathbf{w} = \mathbf{w}'/d,\, b = b'/d$

$$\min_{\mathbf{w},b} \frac{1}{2}\|\mathbf{w}\|^2 \quad \text{subject to } y_i(\mathbf{x}_i^T\mathbf{w} + b) \geq 1, \text{ where } d = \frac{1}{\|\mathbf{w}\|}$$

Convex optimization problem with quadratic objective function and linear constraints.

SVM—Dual, (LS)

Multiply constraint equations by positive Lagrange multipliers and subtract them from the objective function:

$$L_P = \frac{1}{2}\|\mathbf{w}\|^2 - \sum_{i=1}^{N} a_i \left[y_i (\mathbf{x}_i{}^T \mathbf{w} + b) - 1 \right]$$

Min. L_P w.r. t. $\mathbf{w}$ and b and max. w.r. t. a_i, subject to $a_i \geq 0$.

Set derivatives $d\,L_P / d\,\mathbf{w}$ and $d\,L_P / d\,b$ to zero and max. w.r. t. a_i:

$$\mathbf{w} = \sum_{i=1}^{N} a_i y_i \mathbf{x}_i , \quad \sum_{i=1}^{N} a_i y_i = 0.$$

substititung in L_p we get the so called Wolfe dual:

$$\left.\begin{array}{l} \text{max. } L_D = \sum_{i=1}^{N} a_i - \frac{1}{2} \sum_{k=1}^{N} \sum_{i=1}^{N} a_i a_k y_i y_k \mathbf{x}_i{}^T \mathbf{x}_k \\ \\ \text{subject to } a_i \geq 0, \quad \sum_{i=1}^{N} a_i y_i = 0 \end{array} \right\} \begin{array}{l} \text{solve for } a_i \text{ then} \\ \text{compute } \mathbf{w} = \sum a_i y_i \mathbf{x}_i \text{ and} \\ b \text{ from } a_i \left[y_i (\mathbf{x}_i{}^T \mathbf{w} + b) - 1 \right] = 0 \end{array}$$

SVM—Primal vs. Dual (LS)

Primal:

$$\min_{\mathbf{w},b} \frac{\|\mathbf{w}\|}{2} \quad \text{subject to: } y_i(\mathbf{x}_i^T\mathbf{w}+b)\geq 1$$

Dual:

$$\max_{a_i} \sum_{i=1}^{N} a_i - \frac{1}{2}\sum_{k=1}^{N}\sum_{i=1}^{N} a_i a_k y_i y_k \mathbf{x}_i^T \mathbf{x}_k \quad \text{subject to: } a_i \geq 0, \quad \sum_{i=1}^{N} a_i y_i = 0$$

The primal has a dense inequality constraint for every point in the training set. The dual has a single dense equality constraint and a set of box constraints which makes it easier to solve than the primal.

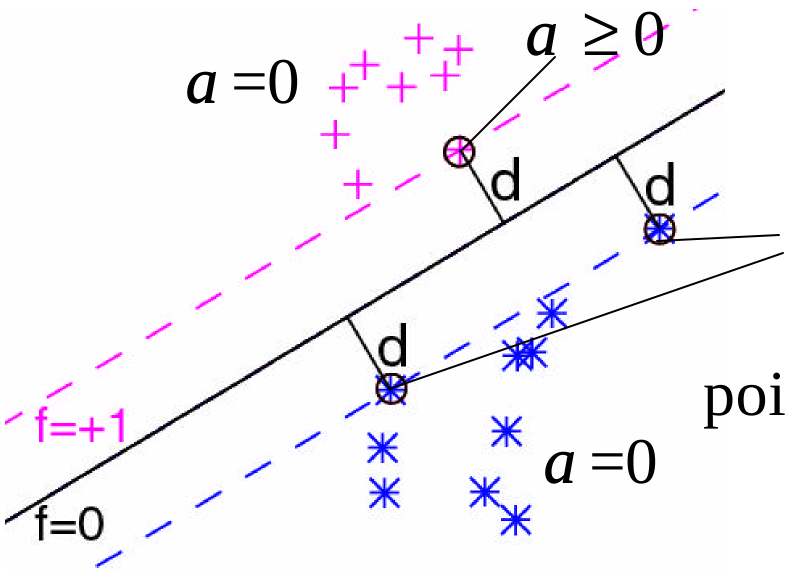
SVM—Optimality Conditions (LS)

Optimality conditions for the linearly separable data:

$$\sum_{i=1}^{N} a_i y_i = 0, \quad a_i \geq 0 \ \forall i, \quad y_i(\mathbf{x}_i^T \mathbf{w} + b) - 1 \geq 0 \ \ \forall i$$

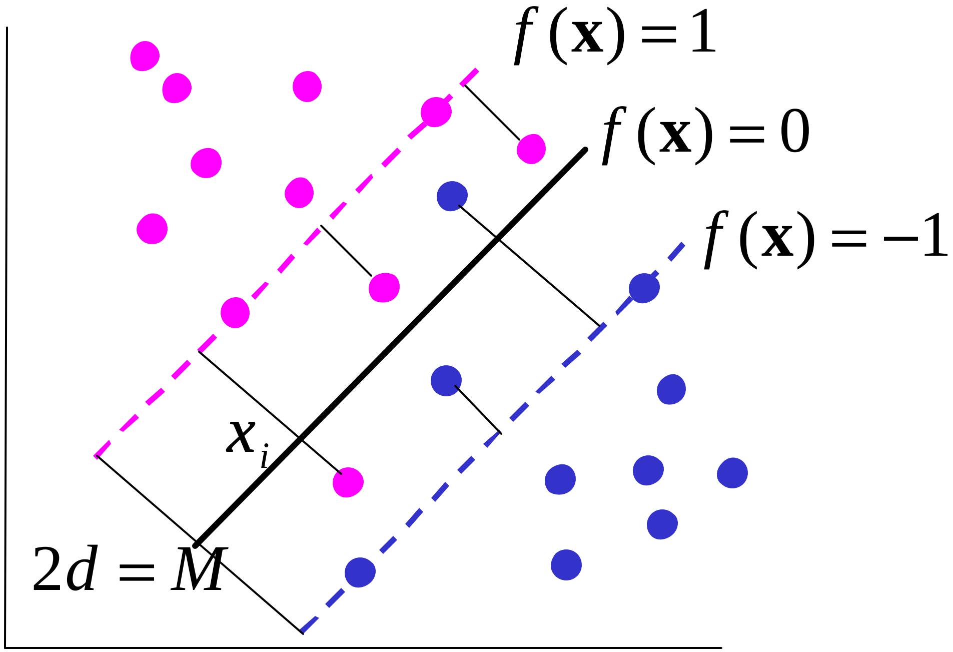
$$\mathbf{w} = \sum_{i=1}^{N} a_i y_i \mathbf{x}_i \ \Rightarrow \ f(\mathbf{x}) = \sum_{i=1}^{N} a_i y_i \mathbf{x}_i^T \mathbf{x} + b$$

$a_i(y_i(\mathbf{x}_i^T \mathbf{w} + b) - 1) = 0 \ \forall i \ \Rightarrow \ a_i = 0$ for points which are not on the boundary of the margin.



points with $a_i > 0$ are support vectors.

SVM—Linear, non-separable (LNS)



$$\min_{\mathbf{w},b}\frac{1}{2}\left\|\mathbf{w}\right\|^2+C\sum_{\mathrm{i=1}}^{\mathrm{N}}x_i \text{ subject to:}$$

$$y_i(\mathbf{x}_i^T\mathbf{w}+b)\geq 1-x_i\ \forall i,\quad x_i>0\ \forall i$$

x_i are called slack variables

C constant, penalizes errors

SVM—Dual (LNS)

Same procedure as in separable case

$$\text{max. } L_D = \sum_{i=1}^{N} a_i - \frac{1}{2} \sum_{k=1}^{N} \sum_{i=1}^{N} a_i a_k y_i y_k \mathbf{x}_i^T \mathbf{x}_k$$

$$\text{subject to } 0 \le a_i \le C, \quad \sum_{i=1}^{N} a_i y_i = 0$$

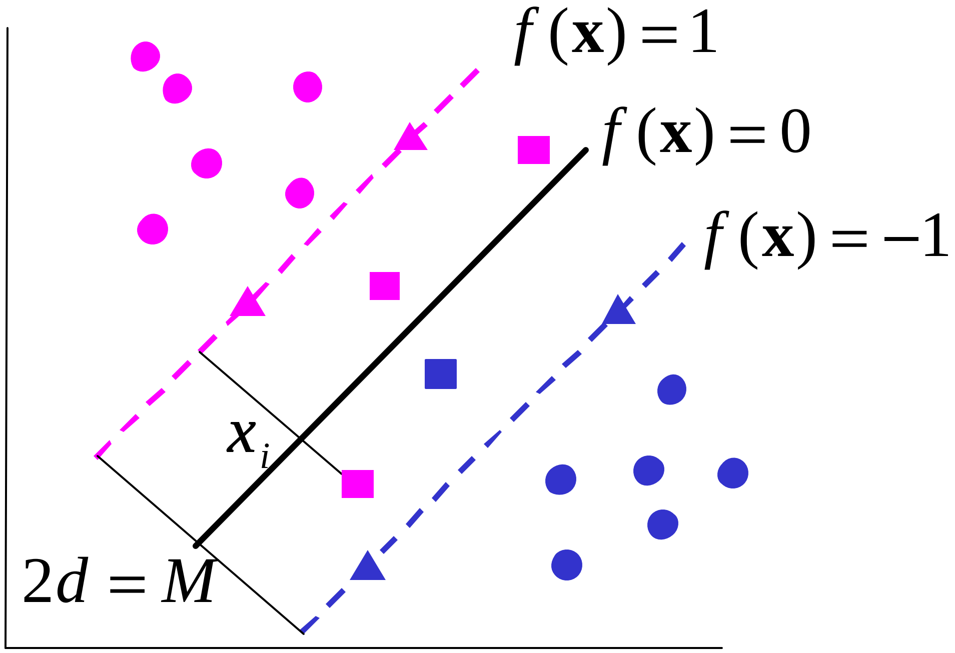
solve for a_i then

compute $\mathbf{w} = \sum a_i y_i \mathbf{x}_i$ and

b from $y_i(\mathbf{x}_i^T \mathbf{w} + b) - 1 = 0$

for any sample $\mathbf{x}_i$ for which $0 < a_i < C$

SVM—Optimality Conditions (LNS)



$$f(\mathbf{x}) = \sum_{i=1}^{N} a_i y_i \mathbf{x}_i^T \mathbf{x} + b$$

$a_i = 0 \Rightarrow x_i = 0, \;\; y_i f(\mathbf{x}_i) > 1$ ●

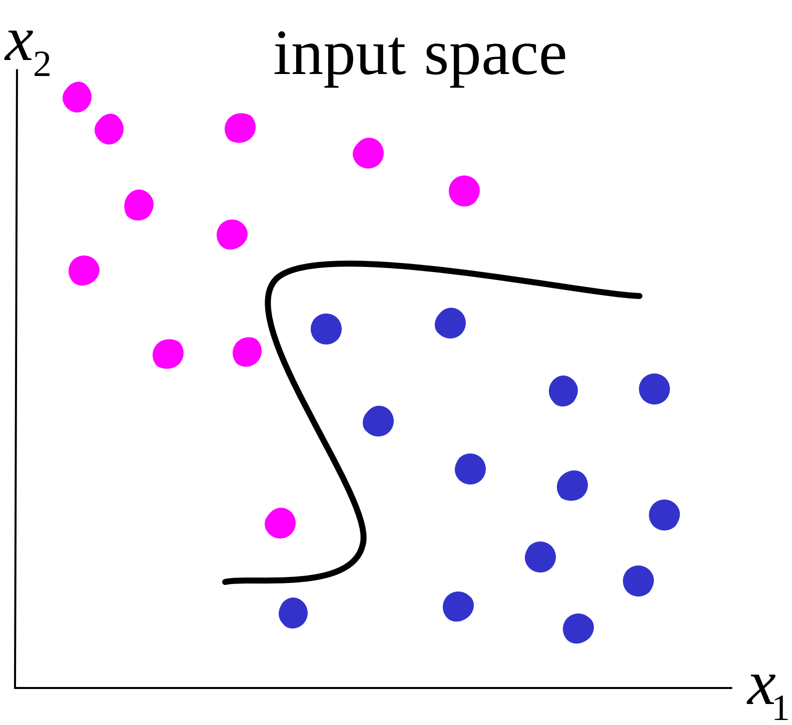
$0 < a_i < C \Rightarrow x_i = 0 \;\; y_i f(\mathbf{x}_i) = 1$ ▲ unbounded support vectors

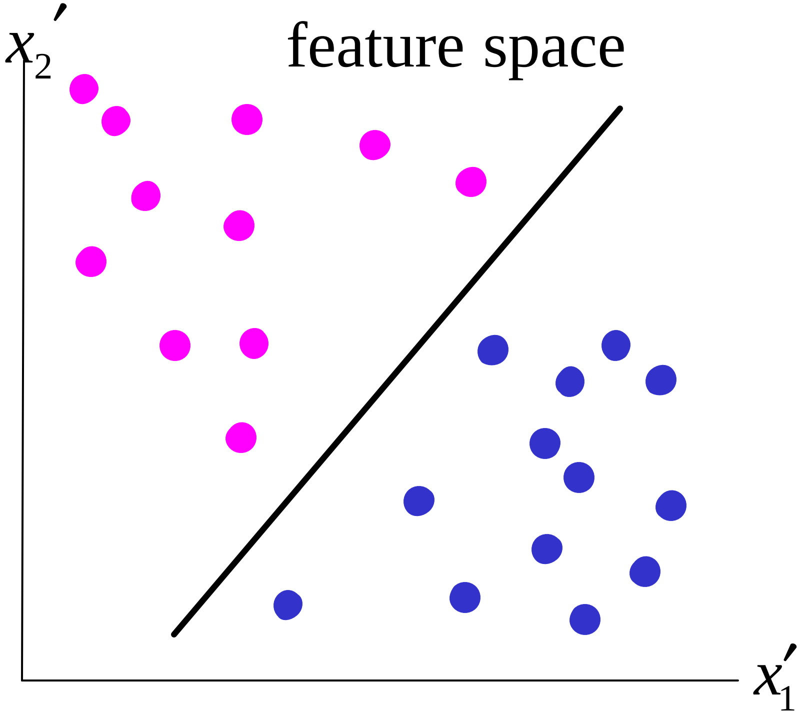
$a_i = C \Rightarrow x_i \geq 0, \;\; y_i f(\mathbf{x}_i) \leq 1$ ■▲ bounded support vectors

SVM—Non-linear (NL)

Non-linear mapping:

$$\mathbf{x}' = \Phi(\mathbf{x})$$





Project into feature space, apply SVM procedure

SVM—Kernel Trick

$$\text{max.}\ \sum_{i=1}^{N} a_i - \frac{1}{2}\sum_{k=1}^{N}\sum_{i=1}^{N} a_i a_k y_i y_k \mathbf{x}_i'^T \mathbf{x}_k'$$

$$\text{subject to } 0 \le a_i \le C, \quad \sum_{i=1}^{N} a_i y_i = 0$$

Only the inner product of the samples appears in the objective function. If we can write: $K(\mathbf{x}_i, \mathbf{x}_k) = \mathbf{x}_i'^T \mathbf{x}_k'$

we can avoid any computations in the feature space.

The solution $f(\mathbf{x}') = \mathbf{x}'^T \mathbf{w} + b$ can be written as:

$$f(\mathbf{x}) = \sum_{i=1}^{N} a_i y_i \Phi(\mathbf{x})^T \Phi(\mathbf{x}_i) + b = \sum_{i=1}^{N} a_i y_i K(\mathbf{x}, \mathbf{x}_i) + b$$

using $\mathbf{w} = \sum a_i y_i \mathbf{x}_i'$.

SVM—Kernels

When is a Kernel $K(\mathbf{u},\mathbf{v})$ an inner product in a Hilbert space?

$$K(\mathbf{u},\mathbf{v}) = \sum_n l_n \overline{f}_n(\mathbf{u}) f_n(\mathbf{v})$$

with positive coefficients l_n

if for any $g(\mathbf{u}) \in L_2$

$$\int K(\mathbf{u},\mathbf{v}) g(\mathbf{u}) \overline{g}(\mathbf{v}) d\mathbf{u} d\mathbf{v} \geq 0 \quad \text{Mercer's condition}$$

Some examples of commonly used kernels:

Linear kernel:	$\mathbf{u}^T \mathbf{v}$
Polynomial kernel:	$(1+\mathbf{u}^T \mathbf{v})^d$
Gaussian kernel (RBF):	$\exp(-\left\|\mathbf{u}-\mathbf{v}\right\|^2)$, shift invar.
MLP:	$\tanh(\mathbf{u}^T \mathbf{v} - q)$

SVM—Polynomial Kernel

Polynomial second degree kernel

$$K(\mathbf{u},\mathbf{v}) = (1+\mathbf{u}^T\mathbf{v})^2, \quad \mathbf{u},\mathbf{v}\in\mathbb{R}^2$$

$$= 1+(u_1v_1)^2+(u_2v_2)^2+2u_1v_1u_2v_2+u_1v_1+u_2v_2$$

$$= (1, u_1{}^2, u_2{}^2, \sqrt{2}u_1u_2, u_1, u_2)(1, v_1{}^2, v_2{}^2, \sqrt{2}v_1v_2, v_1, v_2)^T$$

$$\Phi(\mathbf{x}) = (1, x_1{}^2, x_2{}^2, \sqrt{2}x_1x_2, x_1, x_2)^T$$

Shift invariant kernel $K(\mathbf{u},\mathbf{v}) = K(\mathbf{u}-\mathbf{v})$

defined on $L^2([0,T]^d)$ can be written

as the Fourier series of K:

$$f(t) = \frac{1}{T}\sum_{k=-\infty}^{\infty} l_k\, e^{\frac{j2p\,kt}{T}}, \; f(t-t_0) = \frac{1}{T}\sum_{k=-\infty}^{\infty} l_k\, e^{\frac{j2p\,kt}{T}} e^{-\frac{j2p\,kt_0}{T}}$$

$$K(\mathbf{u}-\mathbf{v}) = \sum_{k=0}^{\infty} l_k\, e^{j2p\mathbf{k}_k\mathbf{u}} e^{-j2p\mathbf{k}_k\mathbf{v}} \; \forall\mathbf{k}_k \in Z([-\infty,\infty]^d)$$

SVM—Uniqueness

Are the a_i in the solution unique? No

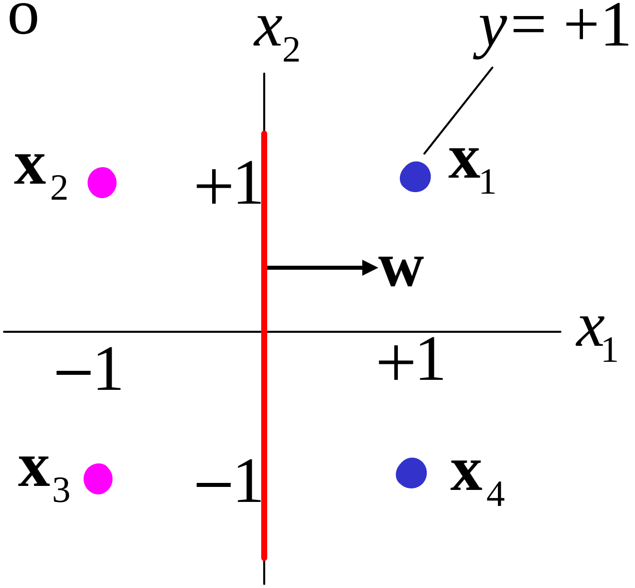
$$f(\mathbf{x}) = \sum_{i=1}^{N} a_i y_i \mathbf{x}_i^T \mathbf{x} + b$$

$$\mathbf{w} = (1,0)$$

two solutions:

$$a = (0.25, 0.25, 0.25, 0.25)$$

$$a = (0.5, 0.5, 0, 0)$$



constraints: $a_i \geq 0, \quad \sum_{i=1}^{N} a_i y_i = 0$ are satisfied

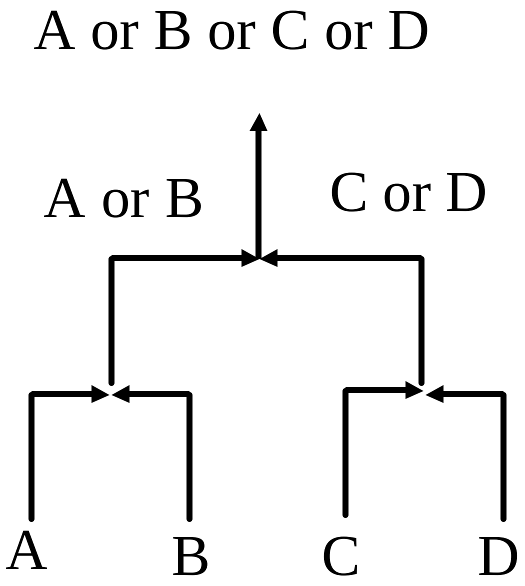
More important: is the solution $f(\mathbf{x})$ unique?

Yes, the solution is unique and global

if the objective function is strictly convex.

SVM—Multiclass

Bottom-Up 1vs1



A or B or C or D

A or B

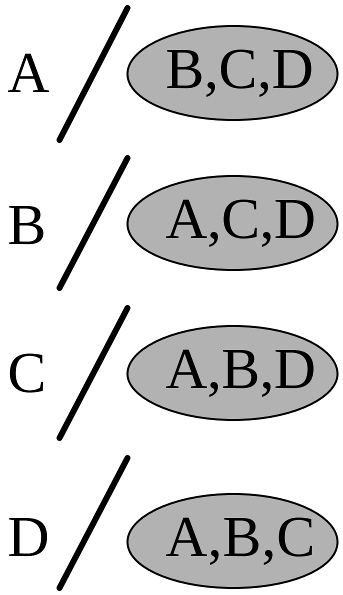
C or D

A B C D

Training: k $(k\text{-}1)$ / 2

Classification : k-1

1 vs. All



A / B,C,D

B / A,C,D

C / A,B,D

D / A,B,C

Training: k

Classification : k

SVM—Choosing the Kernel

How to choose the kernel?

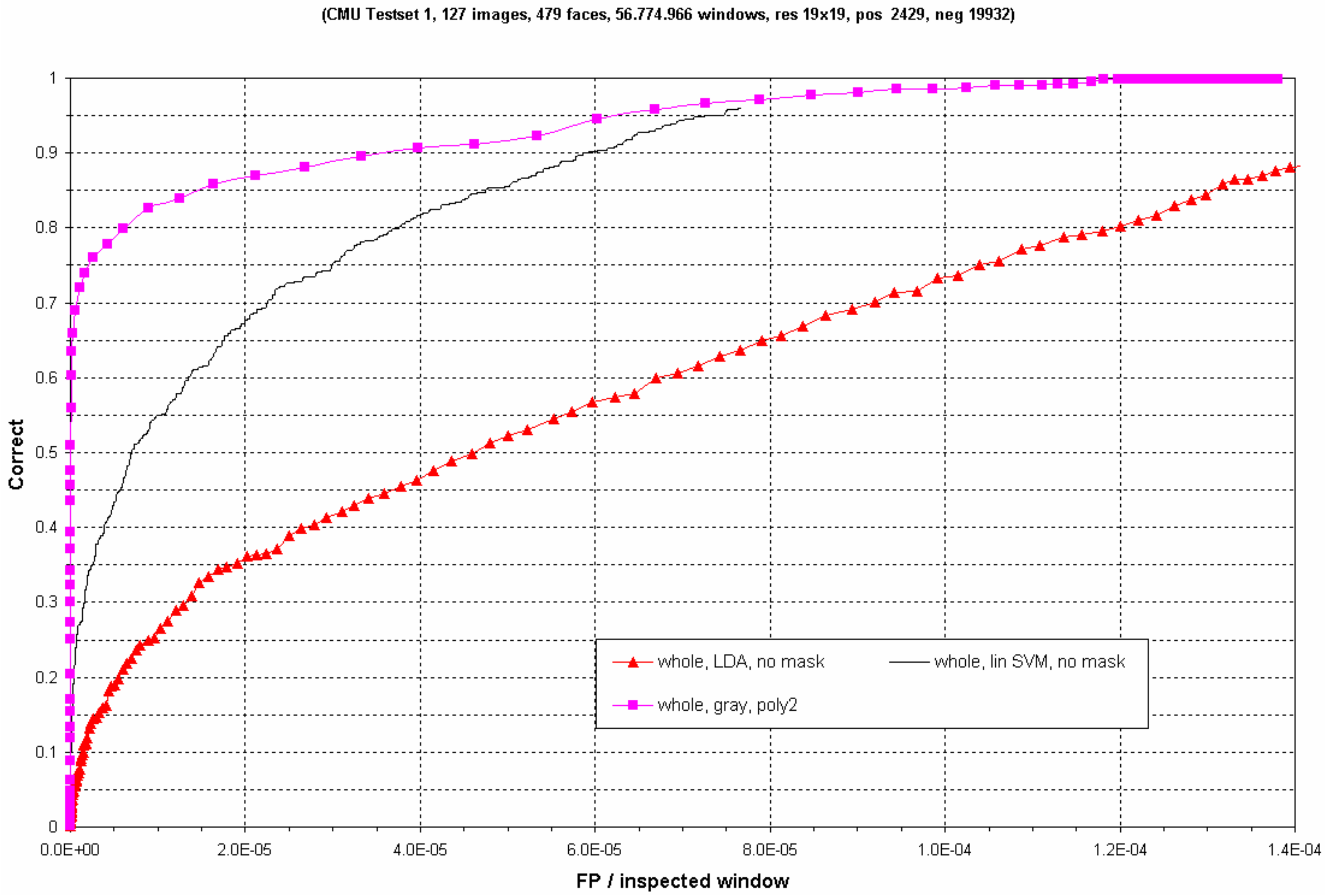
Linear SVMs are simple to compute, fast at runtime but often not sufficient for complex tasks.

SVM with Gaussian kernels showed excellent performance in many applications (after some tuning of sigma). Slow at run-time.

Polynomial with 2nd are commonly used in computer vision applications. Good trade off between classification performance computational complexity.

SVM—Example

Face detection with linear and 2nd degree polyn. SVM & LDA



SVM—Choosing *C*

How to choose the *C*-value?

$$\min_{\mathbf{w},b} \frac{1}{2}\left\|\mathbf{w}\right\|^2 + C\sum_{\mathrm{i=1}}^{\mathrm{N}} x_i$$

C-value penalizes points within the margin.

Large *C*-value can lead to poor generalization performance (over-fitting).
From own experience in **object detection** tasks:
Find a kernel and *C*-values which give you zero errors on the training set.

SVM—Computation during Classification

In computer vision applications fast classification is usually more important than fast training.

Two ways of computing the decision function $f(\mathbf{x})$:

a) $\mathbf{w}^T\Phi(\mathbf{x})+b$ b) $\sum_{i=1}^{N} a_i y_i K(\mathbf{x},\mathbf{x}_i)+b$ Which one is faster?

-For a linear kernel a)

-For a polynomial 2nd degree kernel:

Multiplications for a): $G_{\Phi,poly2}=(n+2)n$, where n is dim. of $\mathbf{x}$

Multiplications for b): $G_{K,poly2}=(n+2)s$, where s is nb. of sv's

-Gaussian kernel: only b) since dim. of $\Phi(\mathbf{x})$ is ∞.

Learning Theory—Problem Formulation

From a given set of training examples $\{\mathbf{x}_i, y_i\}$ learn the mapping $\mathbf{x} \rightarrow y$. The learning machine is defined by a set of possible mappings $\mathbf{x} \rightarrow f(\mathbf{x}, a)$ where a is the adjustable parameter of f.

The goal is to minimize the expected risk R:

$$R(a) = \int V(f(\mathbf{x}, a), y)\, dP(\mathbf{x}, y)$$

V is the loss function

P is the probability distribution function

We can't compute $R(a)$ since we don't know $P(\mathbf{x}, y)$

Learning Theory –Empirical Risk Minimization

To solve the problem minimize the "empirical risk" R_{emp} over the training set :

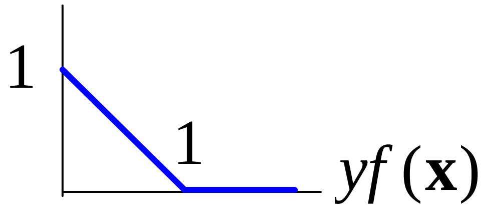
$$R_{emp}(a) = \frac{1}{N}\sum_{i=1}^{N} V(f(\mathbf{x}_i, a), y_i)$$

V is the loss function

Common loss functions:

$V(f(\mathbf{x}), y) = (y - f(\mathbf{x}))^2$ least squares

$V(f(\mathbf{x}), y) = (1 - yf(\mathbf{x}))_+$ hinge loss where $(x)_+ \equiv \max(x, 0)$



Learning Theory & SVM

Bound on the expected risk:

For a loss function with $0 \leq V(f(\mathbf{x}), y) \leq 1$ with probability $1-h$, $0 \leq h \leq 1$ the following bound holds:

$$R(a) \leq R_{emp}(a) + \sqrt{\frac{h\ln(2N/h) + h - \ln(h/4)}{N}}$$

$R_{emp}(a)$ empirical risk

N number of training examples

h Vapnik Chervonenkis (VC) dimension

Bound is independant of the probablility distribution $P(\mathbf{x}, y)$.

Keep all parameters in the bound fixed except one:

$(1-h)\uparrow$ bound $\uparrow$, $N\uparrow$ bound $\downarrow$, $h\uparrow$ bound $\uparrow$

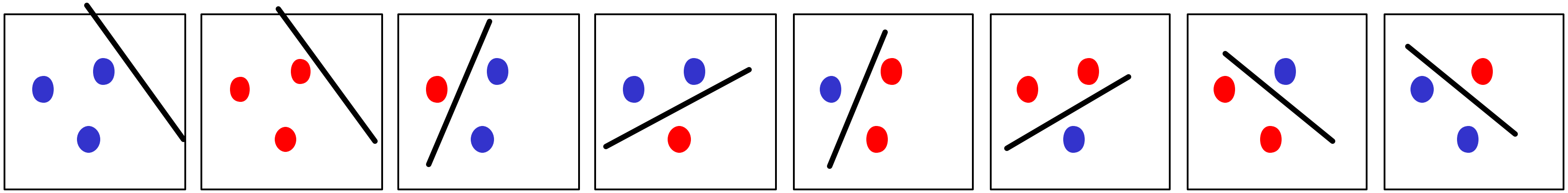
Learning Theory VC Dimension

The VC dimension is a property of the set of functions $\{f(a)\}$.

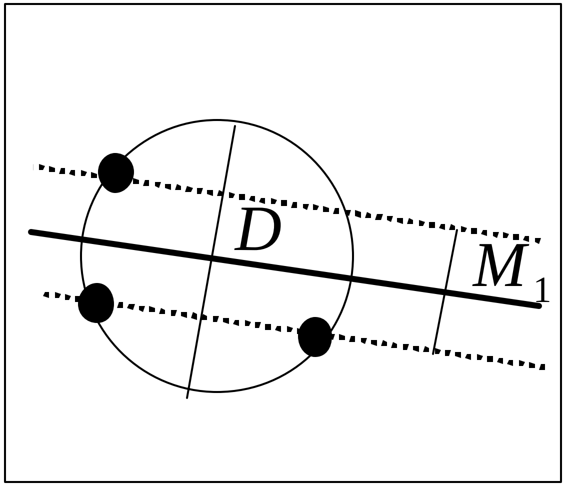
If for a set of N points labeled in all 2^N possible ways one can find an $f \in \{f(a)\}$ which separates the points correctly one says that the set of points is shattered by $\{f(a)\}$.

The VC dimension is the maximum number of points that can be shattered by $\{f(a)\}$.

The VC dimension of a functions $f: \mathbf{w}^T\mathbf{x} + b = 0$ in 2 dim:



Learning Theory—SVM



The expected risk $E(R)$ for the optimal hyperplanes:

$$E(R) \leq \frac{E(D^2 / M^2)}{N}$$

where the expectation is over all training sets of size N.

'Algorithms that maximize the margin have better generalization performance.'

Bounds

Most bounds on expected risk are very loose to compute instead:

Cross Validation Error
Error on a cross validation set which is different from the training set.

Leave-one-out Error
Leave one training example out of the training set, train classifier and test on the example which was left out. Do this for all examples.
For SVMs upper bounded by the # of support vectors.

Regularization Theory

Given N examples $(\mathbf{x}_i, y_i), \mathbf{x} \in \mathbf{R}^n,\ y \in \{0,1\}$ solve:

$$\min_{f \in \mathcal{H}} \frac{1}{N} \sum_{i=1}^{N} V(f(\mathbf{x}_i), y_i) + g \| f \|_K^2$$

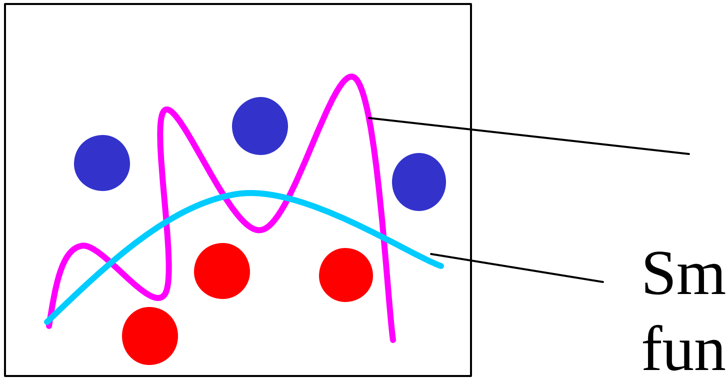
where $\| f \|_K^2$ is the norm in a Reproducing Kernel Hilbert Space (RKHS) $\mathcal{H}$,with the reproducing kernel K,

g is the regularization parameter.

$g \| f \|_K^2$ can be interpreted as a smoothness constraint.

Under rather general conditions the solution can be written as:

$$f(\mathbf{x}) = \sum_{i=1}^{N} c_i K(\mathbf{x}, \mathbf{x}_i)$$



Smooth function

Regularization—Reproducing Kernel Hilbert Space (RKHS)

Reproducing Kernel Hilbert Space (RKHS) $\mathcal{H}$

$$f(\mathbf{x}) = \left\langle \bar{K}(\mathbf{x},\mathbf{y}), f(\mathbf{y}) \right\rangle_{\mathcal{H}}$$

Positive numbers l_n and orthonormal set of

functions $f_n(\mathbf{x}),\quad \int \overline{f_n}(\mathbf{x}) f_m(\mathbf{x}) d\mathbf{x} = 0$ for $n \neq m,$ and 1 otherwise:

$K(\mathbf{x},\mathbf{y}) \equiv \sum_n l_n \overline{f_n}(\mathbf{x}) f_n(\mathbf{y}),\ l_n$ are nonnegative eigenvalues of K

$$f(\mathbf{x}) = \sum_n a_n f_n(\mathbf{x}), \quad a_n = \int f(\mathbf{x}) \overline{f_n}(\mathbf{x}) d\mathbf{x},$$

$$\left\langle f(\mathbf{x}), f(\mathbf{y}) \right\rangle_{\mathcal{H}} \equiv \sum_n \frac{a_n}{\sqrt{l_n}} \frac{b_n}{\sqrt{l_n}}$$

$$\left\| f(\mathbf{x}) \right\|_{\mathcal{H}} = \left\langle f(\mathbf{x}), f(\mathbf{x}) \right\rangle_{\mathcal{H}} = \sum_n a_n^2 / l_n$$

Regularization—Simple Example of RKHS

Kernel is a one dimensional Gaussian with $s = 1$:

$K(x, y) = \exp(-(x-y)^2)$, x, y in $[0,1]$

write $K(x, y)$ as Fourier expansion using

shift theorem:

$$K(x, y) = \sum_n l_n \exp(j2pnx)\exp(-j2pny) \quad \text{Period } T = 1$$

where l_n are the Fourier coeff. of $\exp(-x^2)$

$$l_n = A\exp(-n^2/2)$$

l_n decreases with higher frequencies (increasing *n*).

This is a property of most kernels. The regularization term:

$\| f(x) \|_{\mathcal{H}} = \sum_n a_n^2 / l_n$, where a_n are the Fourier coeff. of $f(x)$

penalizes high freq. more than low freq. → smoothness!

Regularization—SVM

For the hinge loss function $V(f(\mathbf{x}), y) = (1 - yf(\mathbf{x}))_+$ it can be shown that the regularization problem is equivalent to the SVM problem:

$$\min_{f \in \mathcal{H}} \frac{1}{N} \sum_{i=1}^{N} (1 - y_i \, f(\mathbf{x}_i))_+ + l \, \| f \|_K^2$$

introducing slack variables $x_i = 1 - y_i \, f(\mathbf{x}_i)$ we can rewrite:

$$\min_{f \in \mathcal{H}} \frac{1}{N} \sum_{i=1}^{N} x_i + l \, \| f \|_K^2 \, , \text{ subject to: } y_i \, f(\mathbf{x}_i) \geq 1 - x_i , \text{ and } x_i \geq 0 \; \forall i$$

It can be shown that this is equivalent to the SVM problem (up to b) :

$$\text{SVM:} \quad \min_{\mathbf{w}, b} \frac{1}{2} \|\mathbf{w}\|^2 + C \sum_{i=1}^{N} x_i \qquad C = 1/(2 \, l \, N)$$

subject to: $y_i (\mathbf{x}_i^T \mathbf{w} + b) \geq 1 - x_i , \quad x_i \geq 0 \; \forall i$

SVM—Summary

- SVMs are maximum margin classifiers.
- Only training points close to the boundary (support vectors) occur in the SVM solution.
- The SVM problem is convex, the solution is global and unique.
- SVMs can handle non-separable data.
- Non-linear separation in the input space is possible by projecting the data into a feature space.
- All calculations can be done in the input space (kernel trick).
- SVMs are known to perform well in high dimensional problems with few examples.
- Depending on the kernel, SVMs can be slow during classification
- SVMs are binary classifiers. Not efficient for problems with large number of classes.

Literature

T. Hastie, R. Tibshirani, J. Friedman: The Elements of Statistical Learning, Springer, 2001: *LDA, QDA, extensions to LDA, SVM & Regularization*.

C. Burges: A Tutorial on SVM for Pattern Recognition, 1999: *Learning Theory, SVM*.

R. Rifkin: Everything Old is New again: A fresh Look at Historical Approaches in Machine Learning, 2002: *SVM training, SVM multiclass*.

T. Evgeniou, M. Pontil, T. Poggio: Regularization Networks and SVMs, 1999: *SVM & Regularization*.

V. Vapnik: The Nature of Statistical Learning, 1995: *Statistical learning theory, SVM.*

Homework

Classification problem on the NIST handwritten digits data involving PCA, LDA and SVMs.

PCA code will be posted today