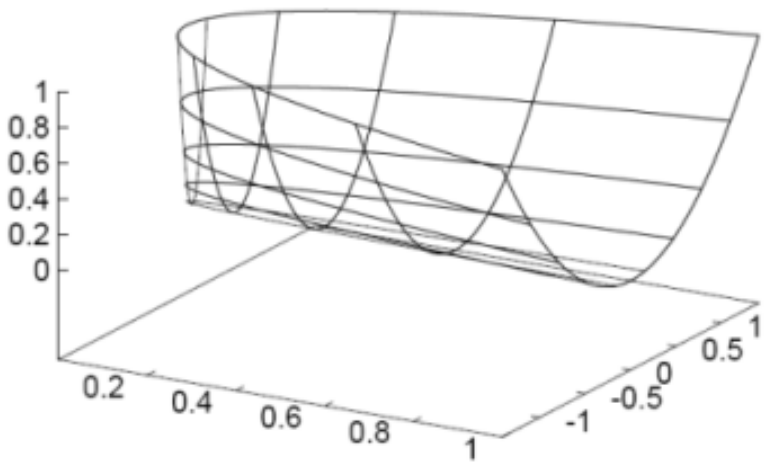
Introduction to Supervised Learning



Week 5:
Support Vector Machines, continued:
Soft Margins, Kernel Trick, VC dimension

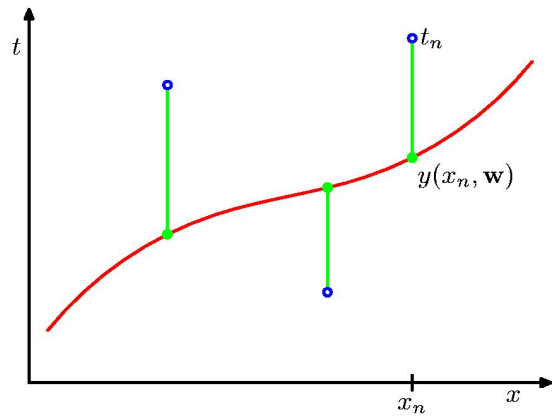
Iasonas Kokkinos

i.kokkinos@cs.ucl.ac.uk

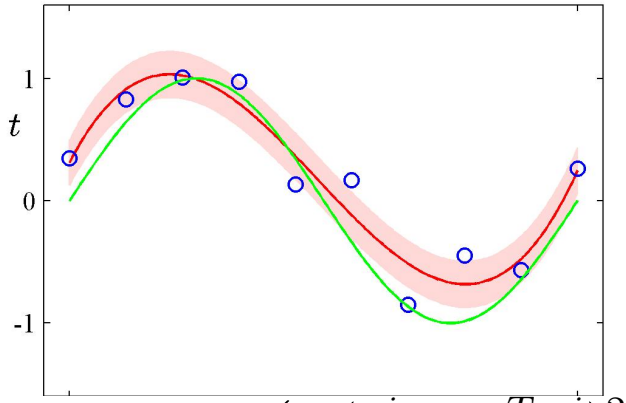
University College London

Weeks 1-4

Week 2 - regression: geometric

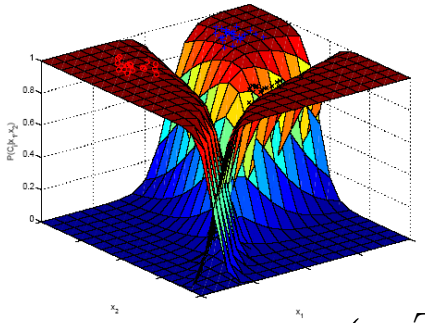


Week 3: probabilistic interpretation



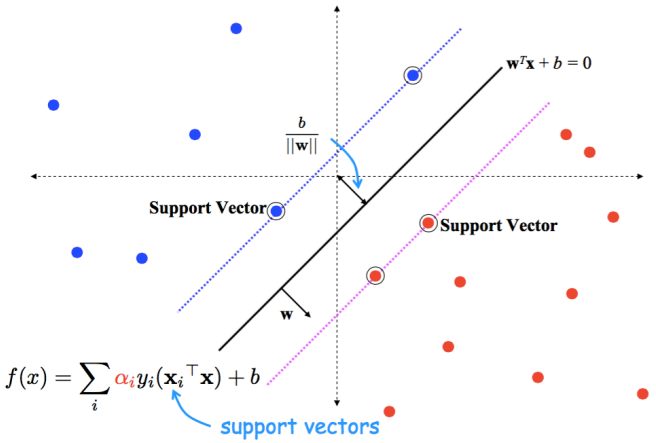
$$P(y^i|\mathbf{x}^i) = \frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{(y^i - \mathbf{w}^T\mathbf{x}^i)^2}{2\sigma^2}\right)$$

Week 3: switch to classification



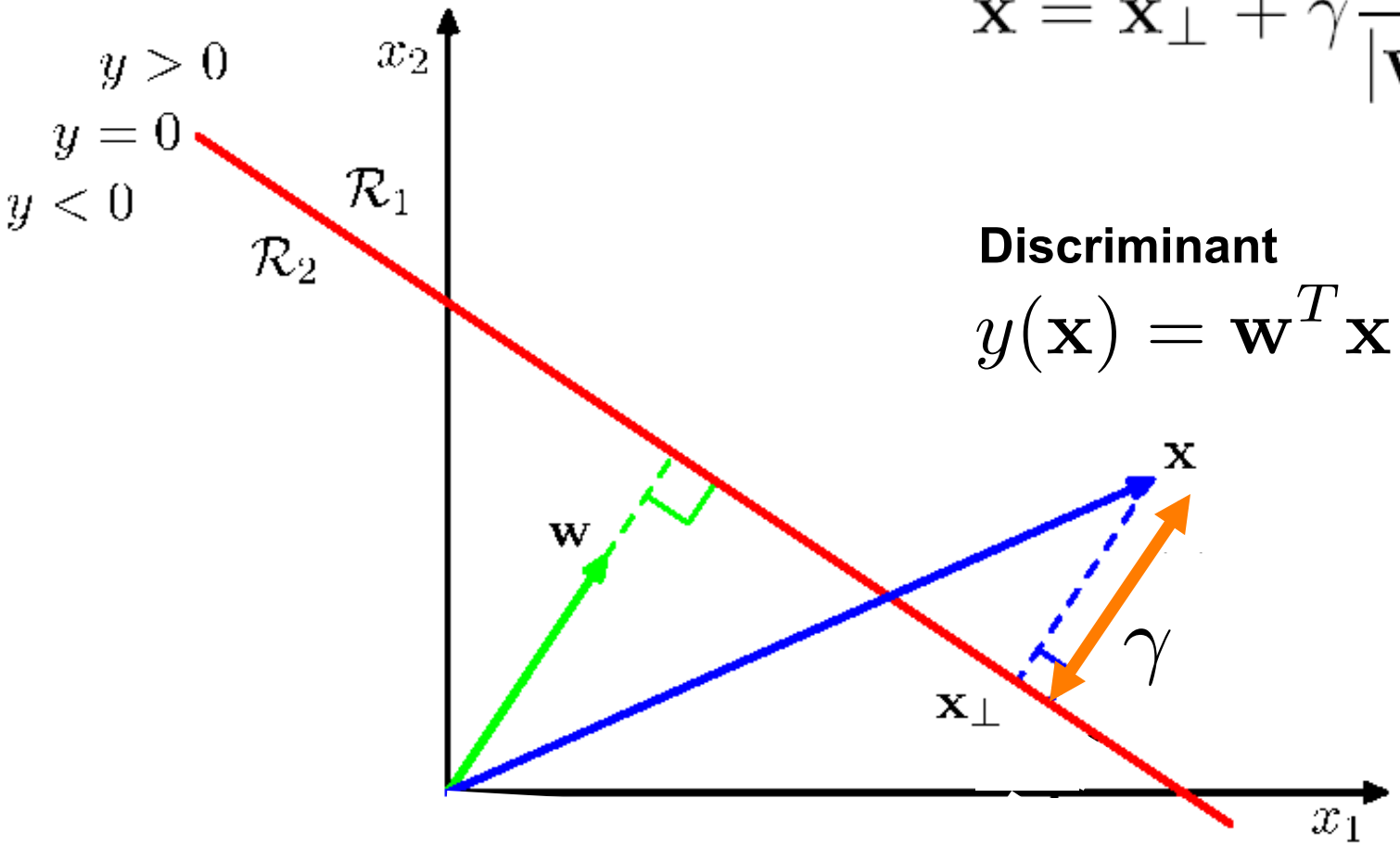
$$P(y^i|\mathbf{x}^i) = \frac{\exp(\mathbf{w}_c^T\mathbf{x})}{\sum_{c'=1}^{C}\exp(\mathbf{w}_{c'}^T\mathbf{x})}$$

Week 4: geometry + classification



Geometric Margins

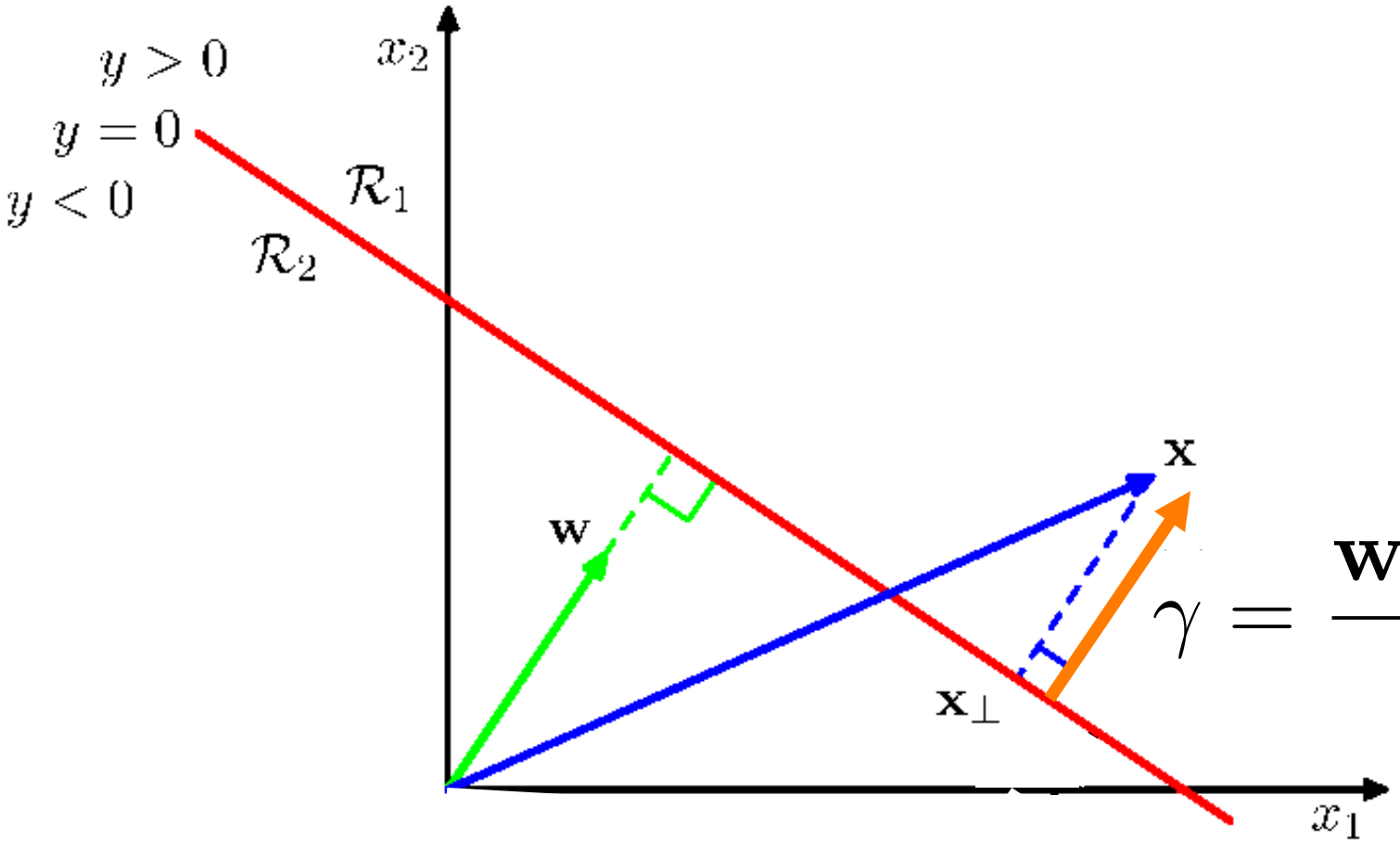
$$\mathbf{x} = \mathbf{x}_\perp + \gamma \frac{\mathbf{w}}{|\mathbf{w}|}$$



Discriminant

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

Geometric Margins



$$\gamma = \frac{\mathbf{w}^T\mathbf{x} + b}{|\mathbf{w}|}$$

Geometric Margin: $\gamma^i = y^i \left(\frac{\mathbf{w}^T}{|\mathbf{w}|}\mathbf{x}^i + \frac{b}{|\mathbf{w}|} \right)$

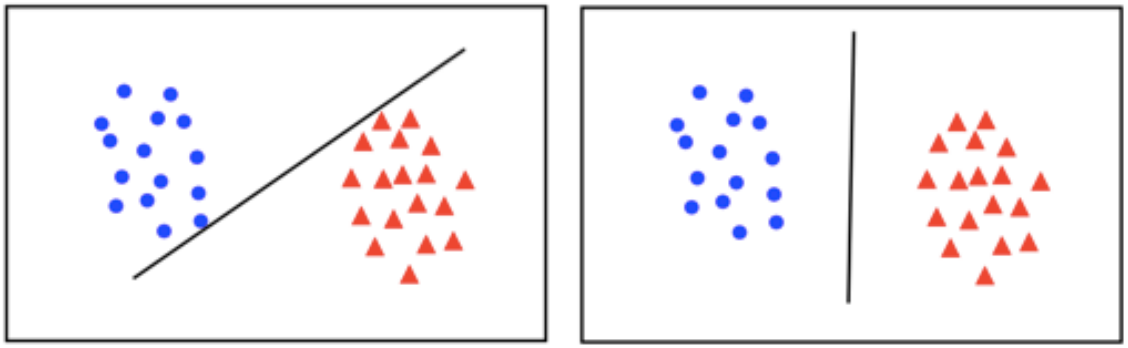
(positive if x is on the correct size of the decision boundary)

What should we be optimizing?

Training set: $\{(\mathbf{x}^1, y^1), \ldots, (\mathbf{x}^N, y^N)\}$

Candidate parameter vector: $(\mathbf{w}, b)$

Related margins: $\gamma^i = y^i(\mathbf{w}^T\mathbf{x}^i + b)$



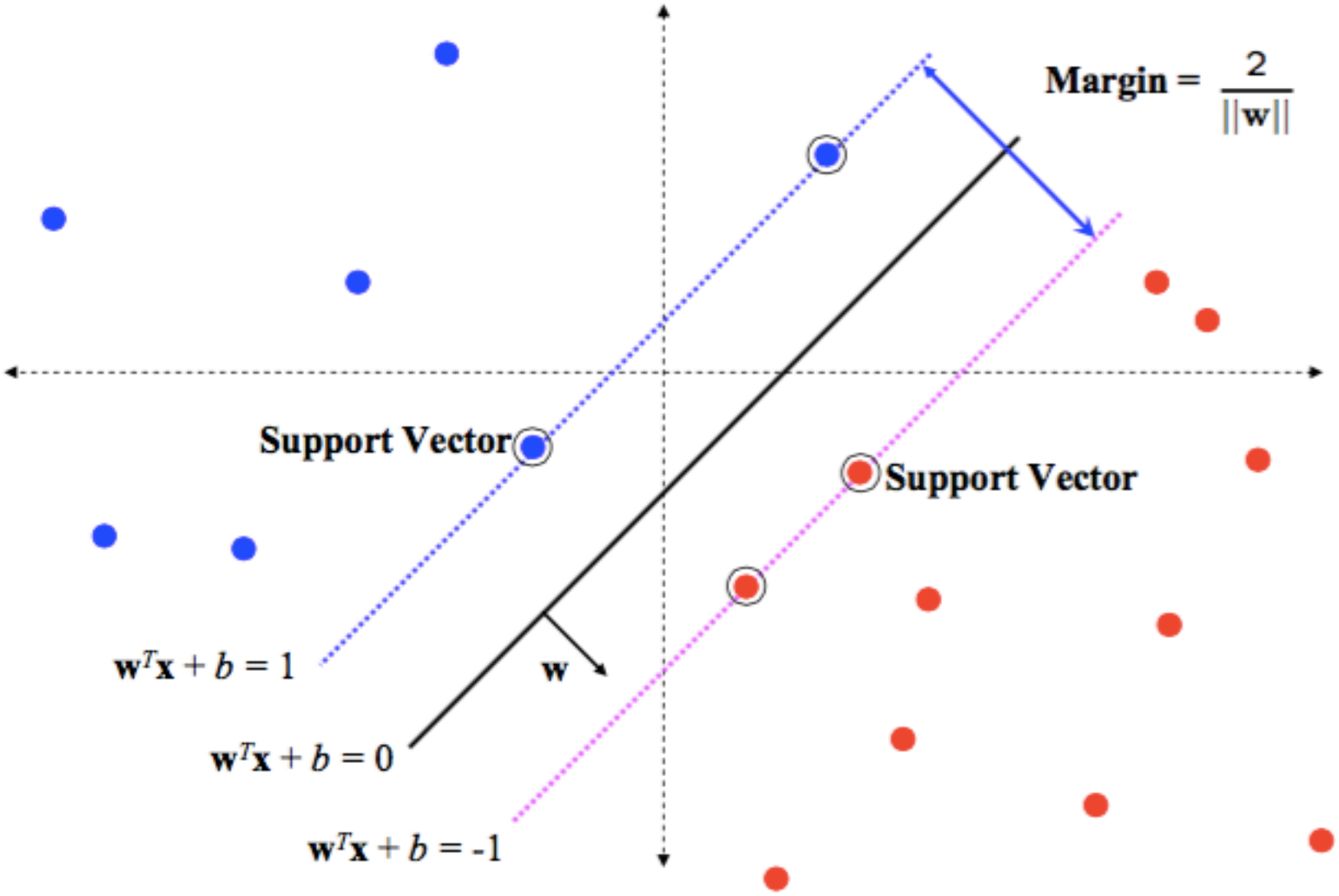
Should we be optimizing the mean, max, min margin?

All points should lie **clearly** on the correct side of the boundary

1) Take points that do not lie clearly on the correct side

2) Make sure they do

Support Vector Machine (SVM)



Representer theorem

Objective: find **w** that maximizes the margin subject to margin constraints

$$\min_{\mathbf{w}} \frac{2}{\|\mathbf{w}\|}$$

$$\text{s.t.} \quad y^i \left(\mathbf{w}^T \mathbf{x}^i + b\right) \geq 1 \quad \forall i$$

Equivalently:

$$\min_{\mathbf{w}} \|\mathbf{w}\|^2$$

$$\text{s.t.} \quad y^i \left(\mathbf{w}^T \mathbf{x}^i + b\right) \geq 1 \quad \forall i$$

Representer Theorem: $\mathbf{w}^* = \sum_{i=1}^{N} \alpha^i \left(y^i \mathbf{x}^i\right)$

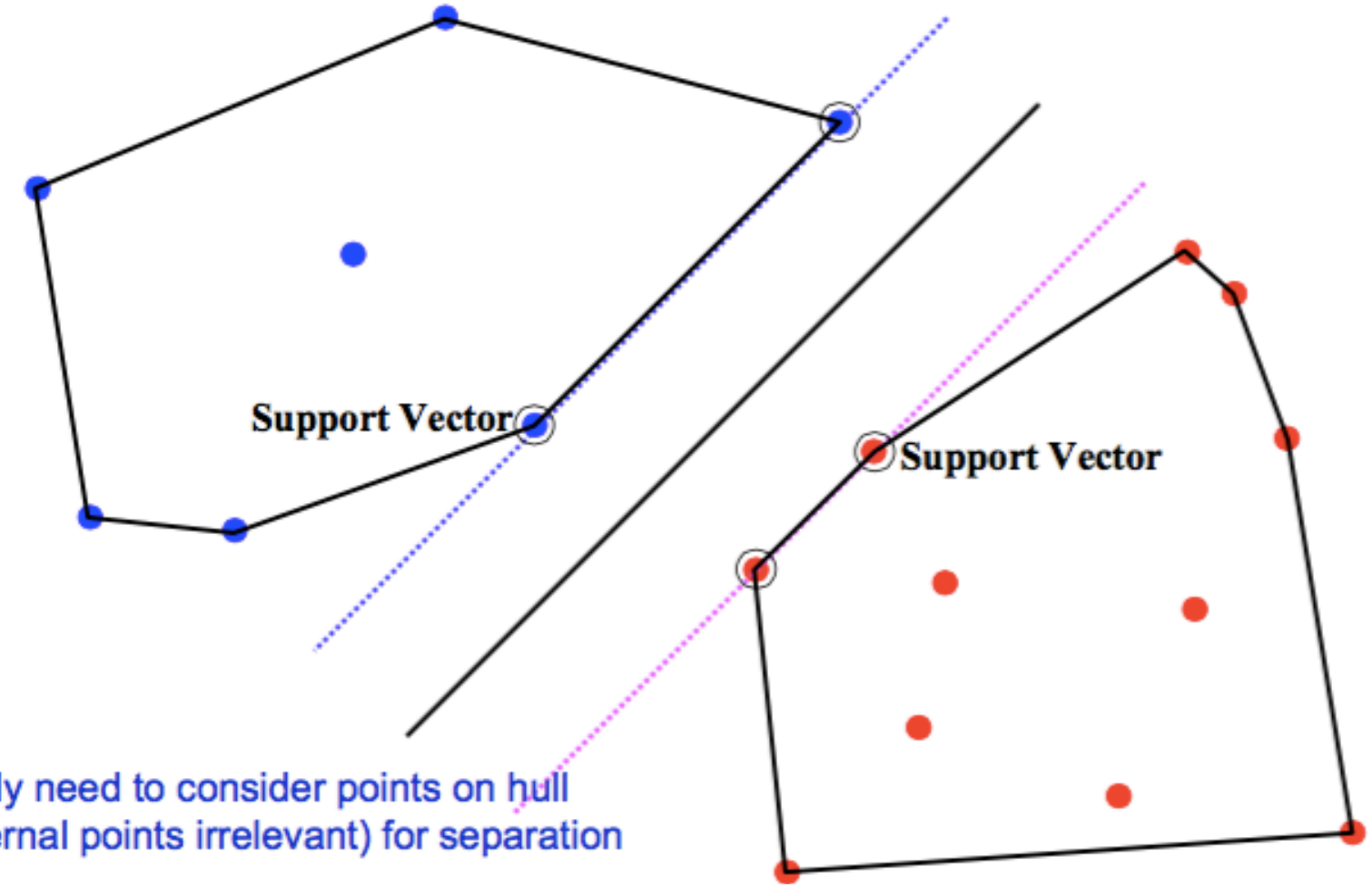
Proof idea: consider solution with component **v** in null-space of **X**

$$\mathbf{w}' = \sum_{i=1}^{N} b^i (y^i \mathbf{x}^i) + \mathbf{v}, \quad \mathbf{v}^T \mathbf{x}^i = 0, \forall i$$

We cannot influence constraints through **v -> b=a -> |w'|>|w|**

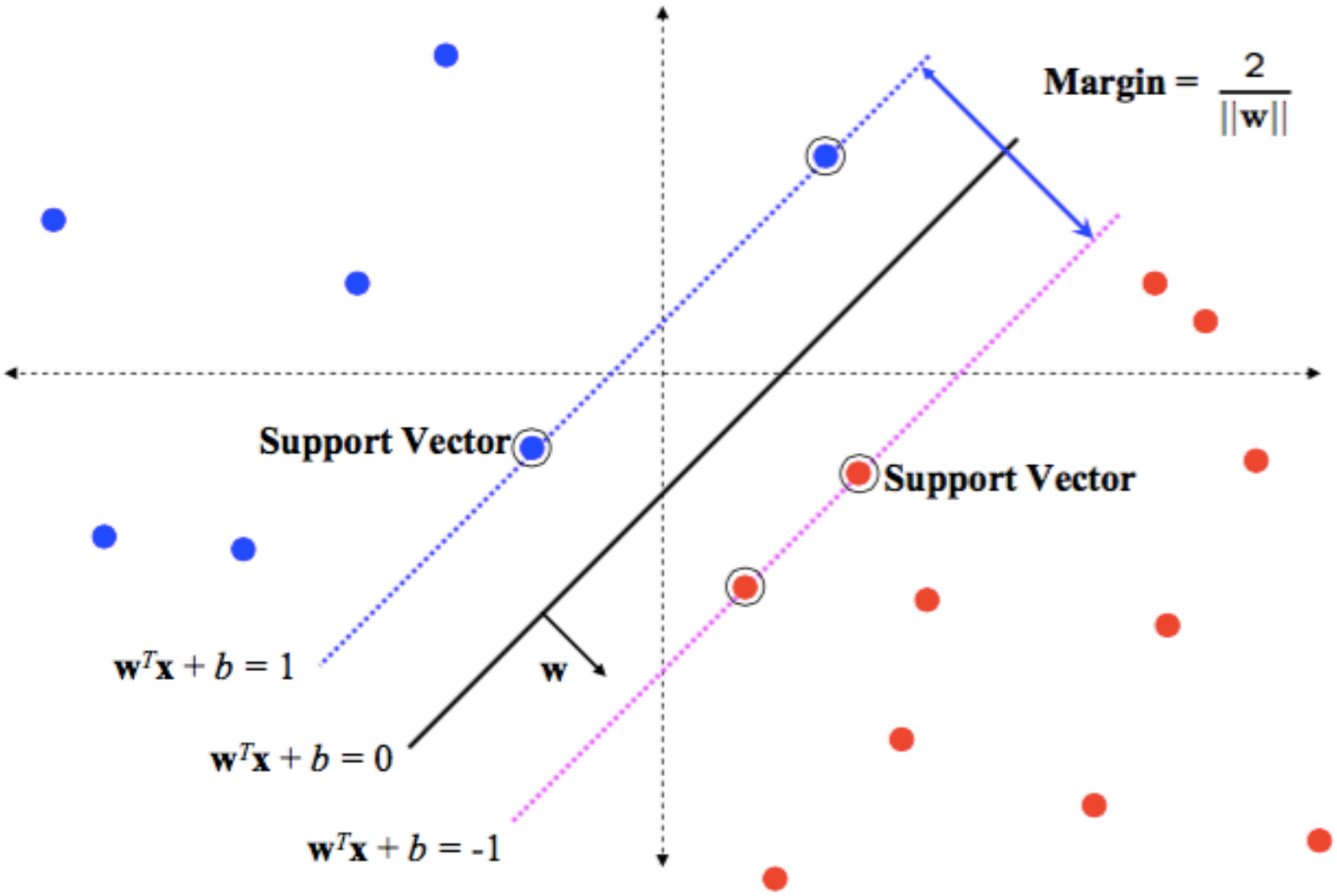
https://en.wikipedia.org/wiki/Representer_theorem + Appendix to slides

Intuitive justification of theorem



• only need to consider points on hull
(internal points irrelevant) for separation

Support Vector Machine (SVM)



Primal and dual problems

Primal, in terms of **w:** $\min_{\mathbf{w}} \|\mathbf{w}\|^2$

$$\text{s.t.}: \quad y^i(\mathbf{w}^T\mathbf{x}^i + b) \geq 1, \quad \forall i$$

But: $\left\|\mathbf{w}^*\right\|^2 = \left\langle \mathbf{w}^*, \mathbf{w}^* \right\rangle$

$$\mathbf{w}^* = \sum_{i=1}^{N} \alpha^i \left(y^i \mathbf{x}^i\right)$$

$$= \left\langle \sum_{i=1}^{N} \alpha^i y^i \mathbf{x}^i, \sum_{j=1}^{N} \alpha^j y^j \mathbf{x}^j \right\rangle = \sum_{i=1}^{N}\sum_{j=1}^{N} \alpha^i \alpha^j y^i y^j \left\langle \mathbf{x}^i, \mathbf{x}^j \right\rangle$$

Dual, in terms of $\boldsymbol{\alpha} = (\alpha_1, \ldots, \alpha_N)$: $\min_{\boldsymbol{\alpha}} \sum_{i=1}^{N}\sum_{j=1}^{N} \alpha^i \alpha^j y^i y^j \left\langle \mathbf{x}^i, \mathbf{x}^j \right\rangle$

$$\text{s.t.}: \quad y^i \left(\sum_{j=1}^{N} \alpha^j y^j \left\langle \mathbf{x}^j, \mathbf{x}^i \right\rangle + b \right) \geq 1, \quad i = 1, \ldots, N$$

Primal vs dual

Primal:
$$\min_{\mathbf{w}} \|\mathbf{w}\|^2 \qquad \mathbf{w} \in \mathbb{R}^D \rightarrow O(D^3)$$
$$\text{s.t. :} \quad y^i(\mathbf{w}^T\mathbf{x}^i + b) \geq 1, \quad \forall i$$

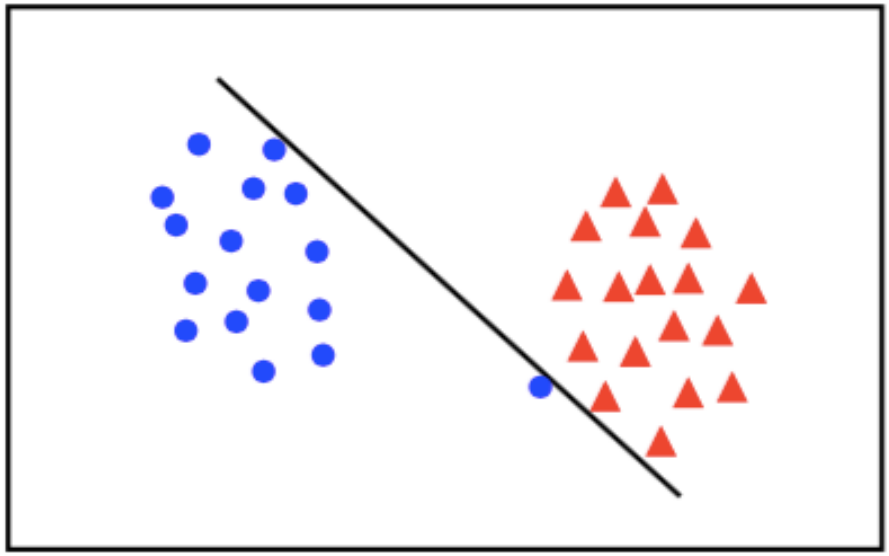
Dual:
$$\min_{\boldsymbol{\alpha}} \sum_{i=1}^{N}\sum_{j=1}^{N} \alpha^i \alpha^j y^i y^j \langle \mathbf{x}^i, \mathbf{x}^j \rangle \qquad \boldsymbol{\alpha} \in \mathbb{R}^N \rightarrow O(N^3)$$
$$\text{s.t. :} \quad y^i \left(\sum_{j=1}^{N} \alpha^j y^j \langle \mathbf{x}^j, \mathbf{x}^i \rangle + b \right) \geq 1, \quad \forall i$$

Dual can be faster if N<D!

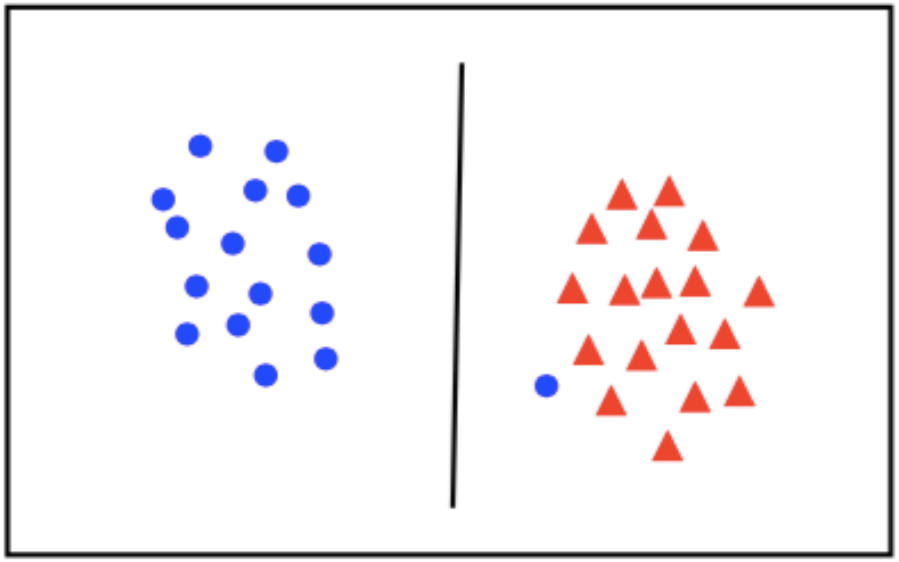
Primal and dual classifier forms:
$$f(\mathbf{x}) = \langle \mathbf{w}, \mathbf{x} \rangle + b = \sum_{i=1}^{N} \alpha^i y^i \langle \mathbf{x}^i, \mathbf{x} \rangle + b$$

Dual form involves only inner products of features (=> kernel trick)

What is the “best” decision plane?



All points on the correct side!



But this looks better overall!

Best: understood at test time

Maybe we could sacrifice classifying some training points correctly

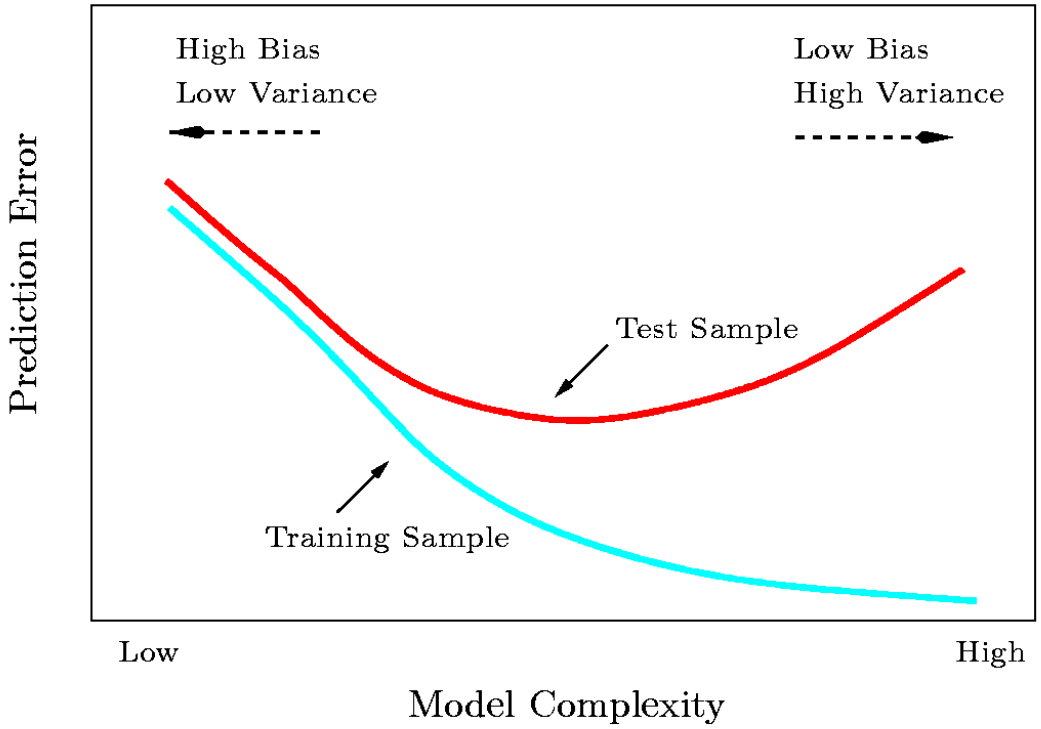
Tuning the model's complexity

A flexible model approximates the target function well in the training set

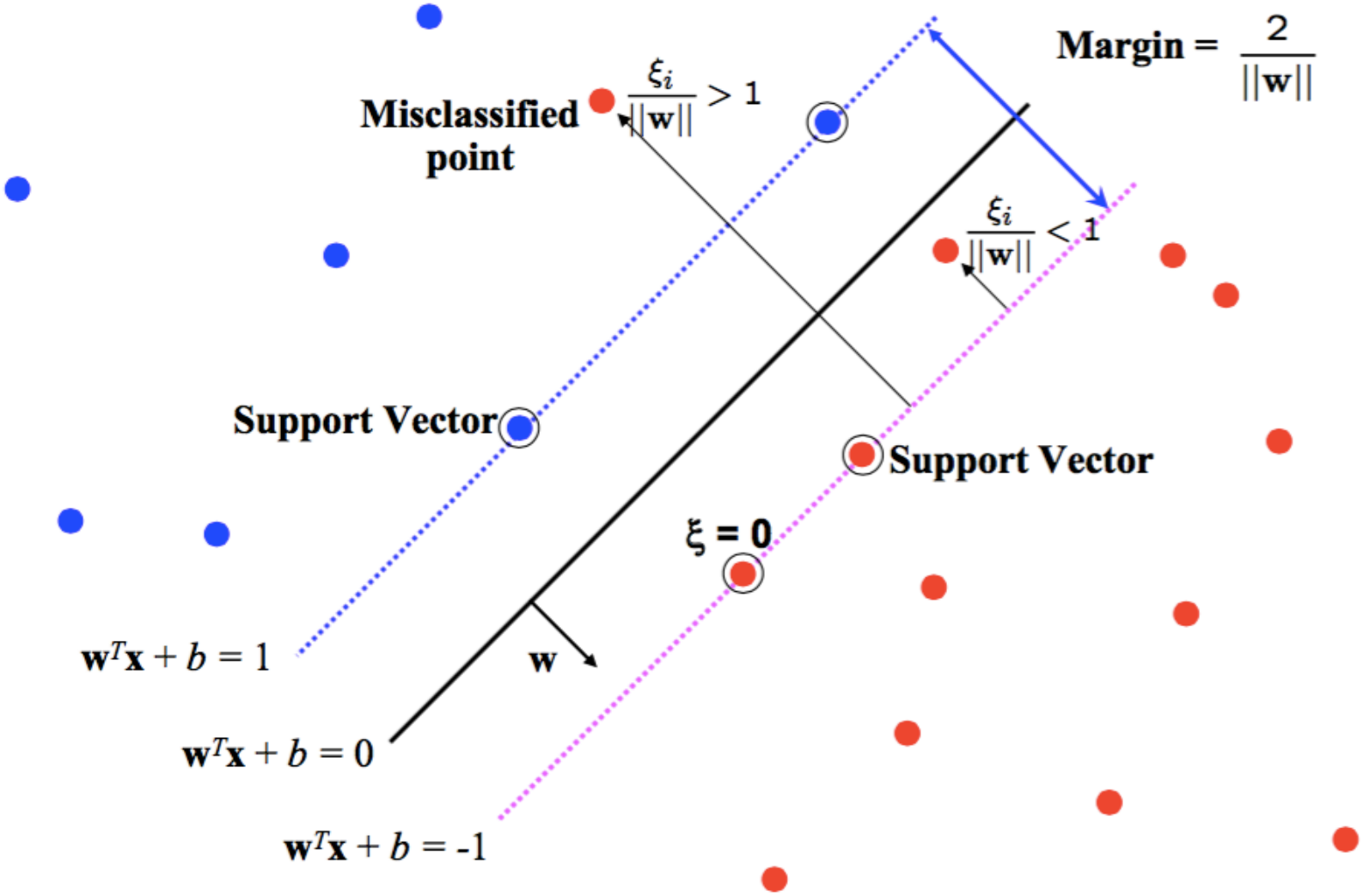
but can "overtrain" and have poor performance on the test set ("variance")

A rigid model's performance is more predictable in the test set

but the model may not be good even on the training set ("bias")



Slack variables: let us make (but also pay) some errors



Objective for non-separable data

$$\min_{\mathbf{w},\boldsymbol{\xi}} \|\mathbf{w}\|^2 + C\sum_{i=1}^{N}\xi^i$$

newcomers

$$\text{s.t.}: \quad y^i(\mathbf{w}^T\mathbf{x}^i + b) \geq 1 - \xi^i, \quad \forall i$$

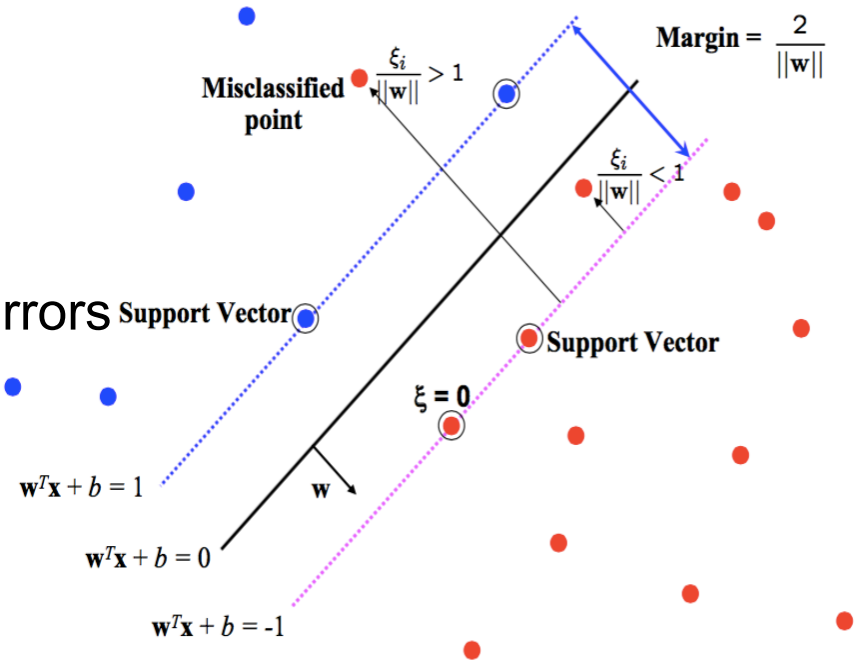
$$\xi^i \geq 0, \quad \forall i$$

misclassification when ξ>1

$\sum_i \xi^i$:upper bound on number of errors

C: hyperparameter

(cross-validation!)



Loss function

Optimization problem:

$$\min_{\mathbf{w},b} \quad \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{N}\xi^i$$

$$s.t. \quad y^i(\mathbf{w}^T x^i + b) \geq 1 - \xi^i$$

$$\xi^i \geq 0$$

At optimum :

$$\xi^i = \max(0, 1 - y^i(\mathbf{w}^T x^i + b))$$

(inspection + theory)

$$= \max(0, 1 - y^i h_{\mathbf{w},b}(x^i))$$

What if we plug that in the optimization objective?

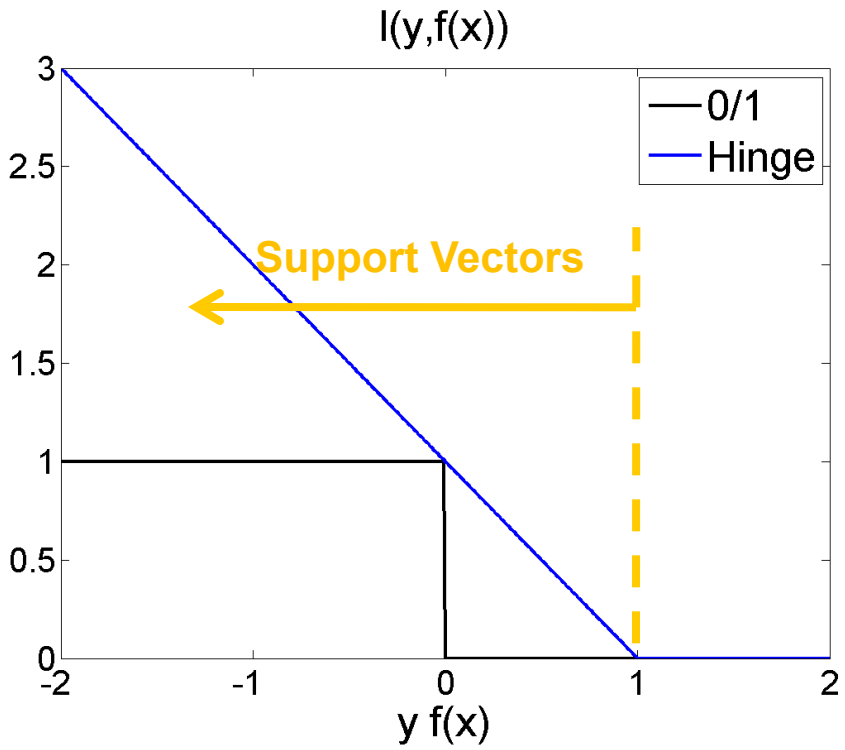
Loss function

Optimization problem: $L(\mathbf{w}) = \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{N}\max(0, 1 - y^i h_{\mathbf{w},b}(x^i))$

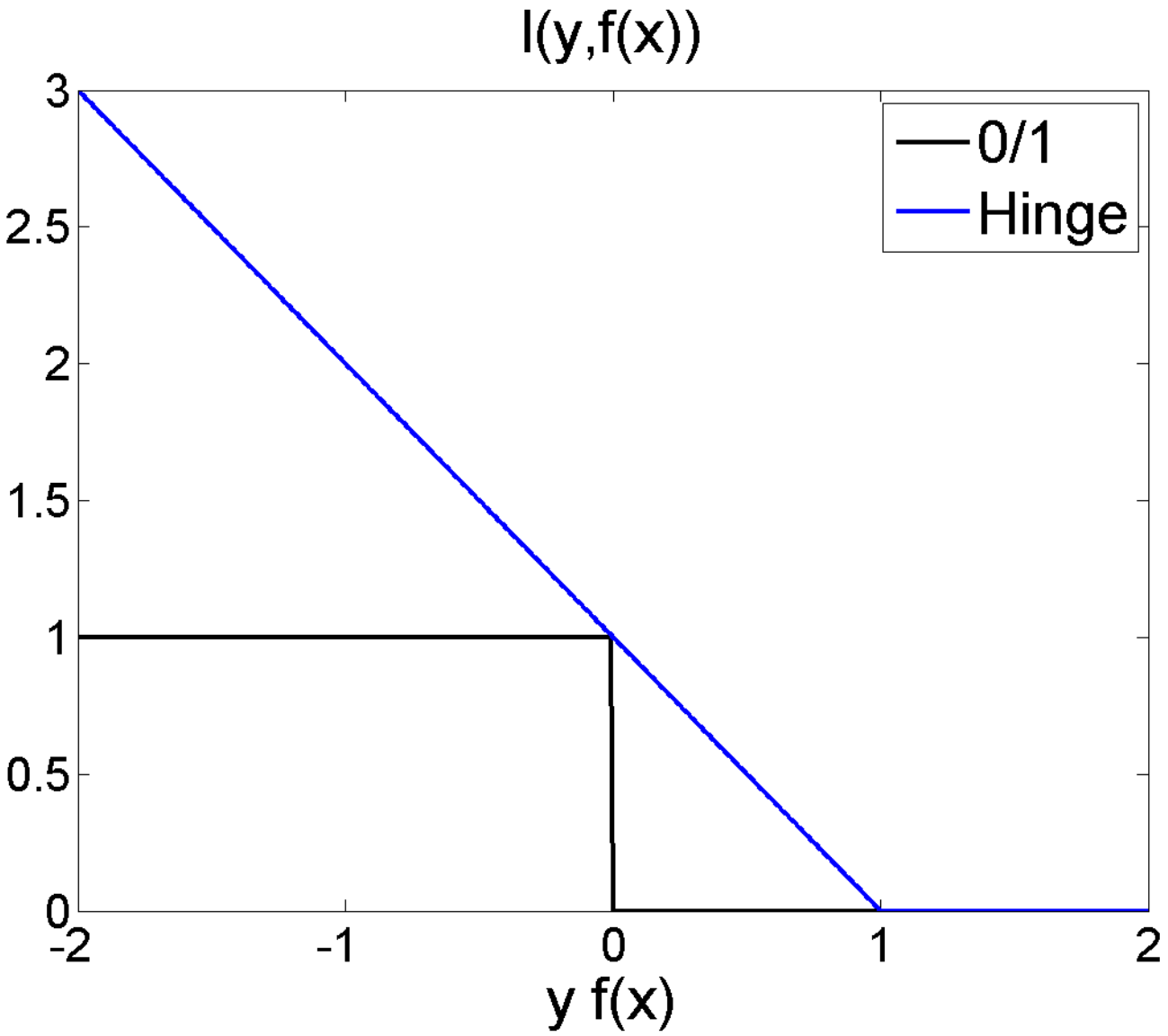
$$\propto \lambda\|\mathbf{w}\|^2 + \sum_{i=1}^{N}\underbrace{\max(0, 1 - y^i h_{\mathbf{w},b}(x^i))}_{l(y^i, x^i)}$$

regularizer **additive loss**

Hinge loss:

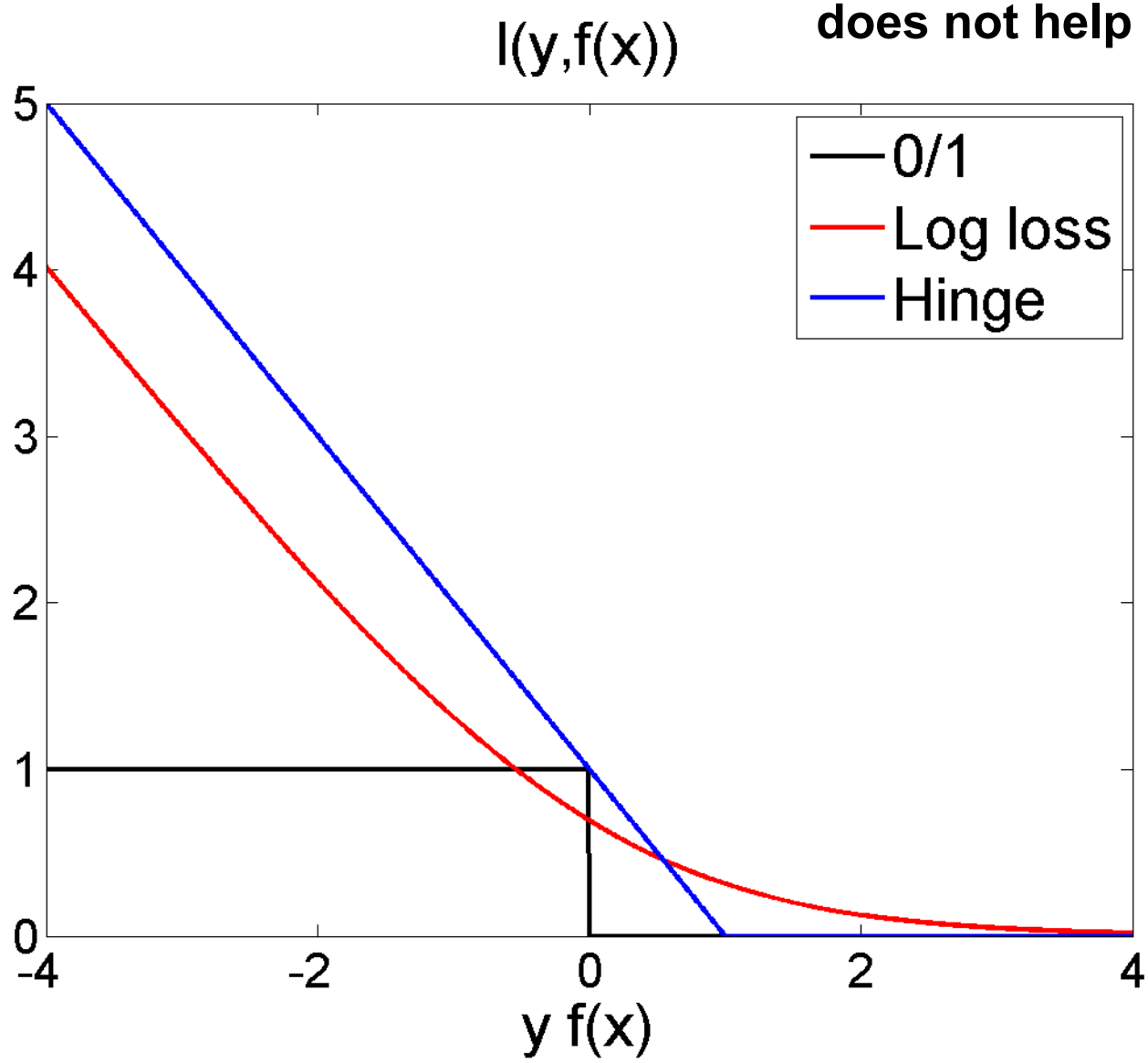


Hinge loss

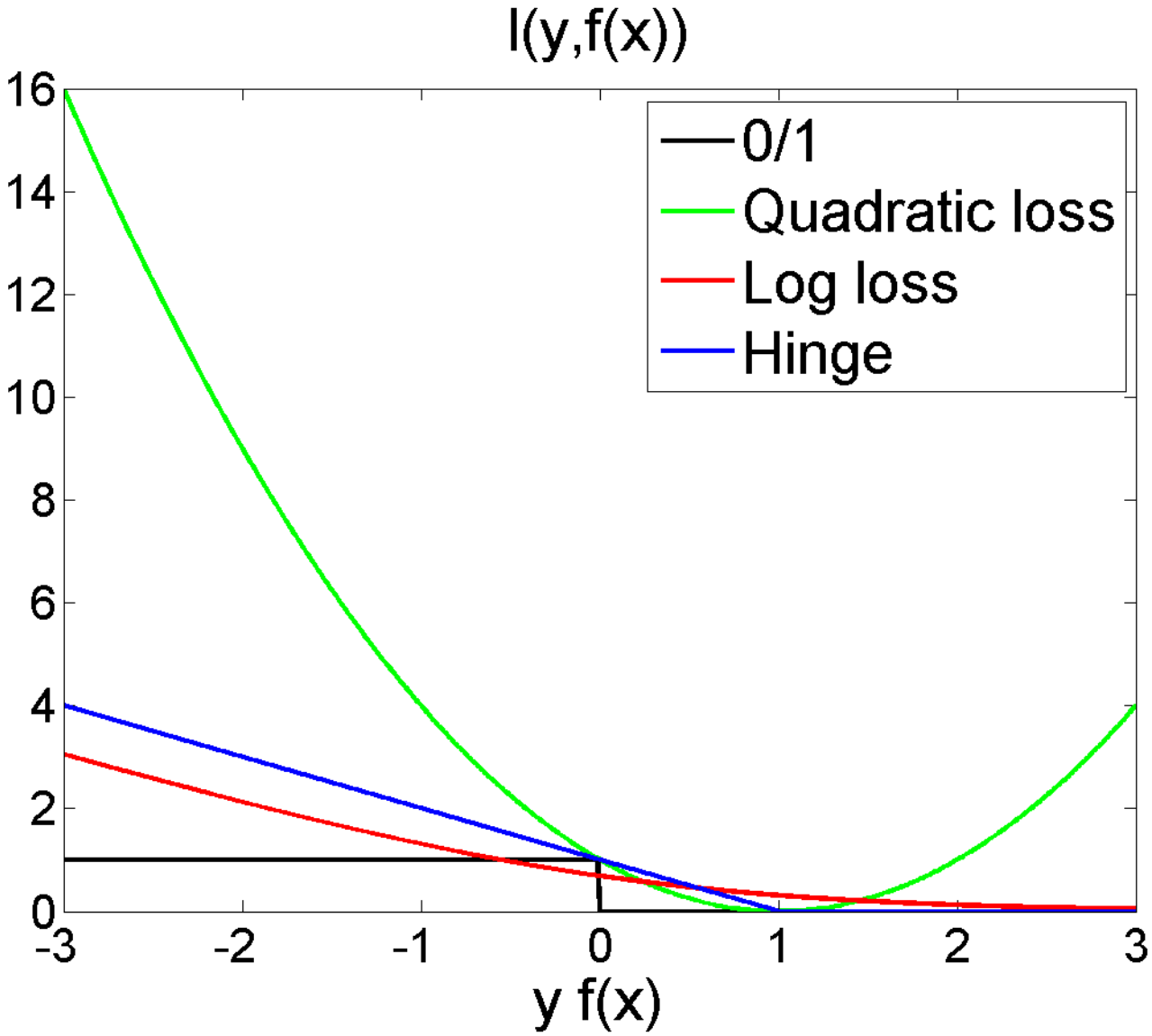


Hinge loss vs log-loss

getting larger than 1: does not harm, but also does not help



Hinge loss vs log-loss vs quadratic



Lecture outline

Recap

Large margins and generalization

Optimization

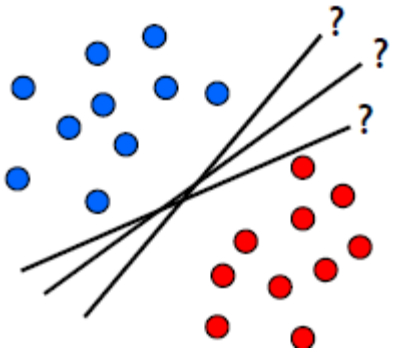
Kernels

Applications to vision

Generalization Error

- What is model complexity?
  - Number of parameters, magnitude of discriminant w?
  - Analyze complexity of hypothesis class

- Linear classifiers:
  - Different decision boundaries
    - Different generalization performance
  - Test error > training error
  - Which line gives smallest test error?



Learning Theory

- V. Vapnik, 1968
  - Mainstream Statistics: Large-sample analysis (`in the limit')
  - Pattern Recognition: Small sample properties

- Distribution-free bounds on worst performance

Empirical and Actual risk

- **Empirical risk**
  - Measured on the training/validation set

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

- **Actual risk (= Expected risk)**
  - Expectation of the error on *all* data.

$$R(\alpha) = \int L(y_i, f(\mathbf{x}; \alpha)) dP_{X,Y}(\mathbf{x}, y)$$

  - $P_{X,Y}(\mathbf{x}, y)$ is the probability distribution of $(\mathbf{x}, y)$. It is fixed, but typically unknown.

Actual and Empirical Risk

- Idea
  - Compute an upper bound on the actual risk based on the empirical risk

$$R(\alpha) \leq R_{emp}(\alpha) + \epsilon(N, p^*, h)$$

  - where

    N: number of training examples

    p^*: probability that the bound is correct

    h: capacity of the learning machine ("VC-dimension")

- With probability $(1\text{-}\eta)$, the following bound holds

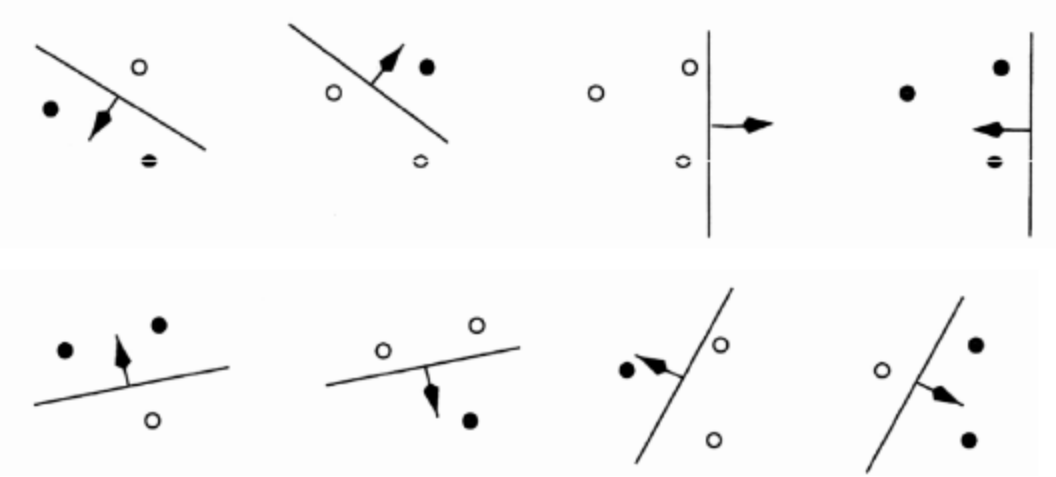
$$R(\alpha) \leq R_{emp}(\alpha) + \sqrt{\frac{h(\log(2N/h) + 1) - \log(\eta/4)}{N}}$$

Vapnik Chervonenkis (VC) Dimension

- Shattering: *If a given set of ℓ points can be labeled in all possible 2^ℓ ways, and for each labeling, a member of the set $\{f(\alpha)\}$ can be found which correctly assigns those labels, we say that the set of points is shattered by the set of functions.*

- VC dimension *The VC dimension for the set of functions $\{f(\alpha)\}$ is defined as the maximum number of training points that can be shattered by $\{f(\alpha)\}$.*

- Example



Large Margins & VC Dimension

- Vapnik: *The class of optimal linear separators has VC dimension h bounded from above as*

$$h \leq \min\left\{\left\lceil \frac{D^2}{\rho^2} \right\rceil, m_0\right\} + 1$$

where ρ is the margin, D is the diameter of the smallest sphere that can enclose all of the training examples, and m_0 is the dimensionality.

- If we maximize the margins, feature dimensionality does not matter

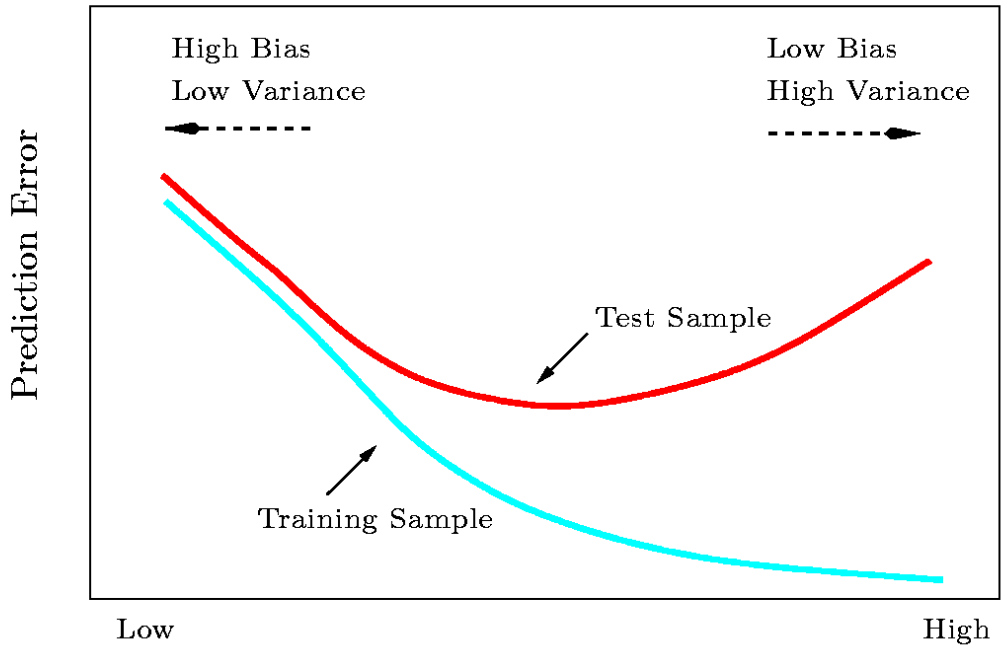
Tuning the model's complexity

A flexible model approximates the target function well in the training set

A rigid model's performance is more predictable in the test set

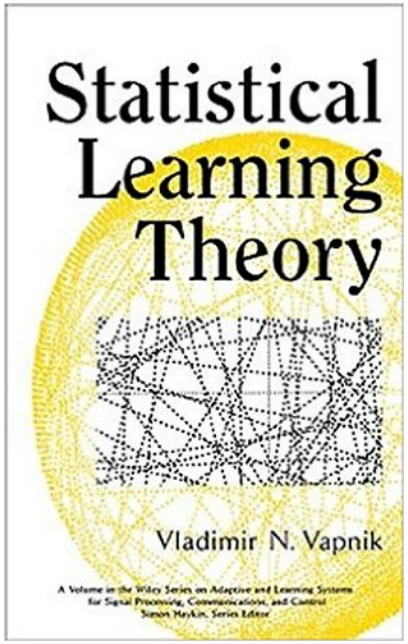
- With probability $(1-\eta)$, the following bound holds

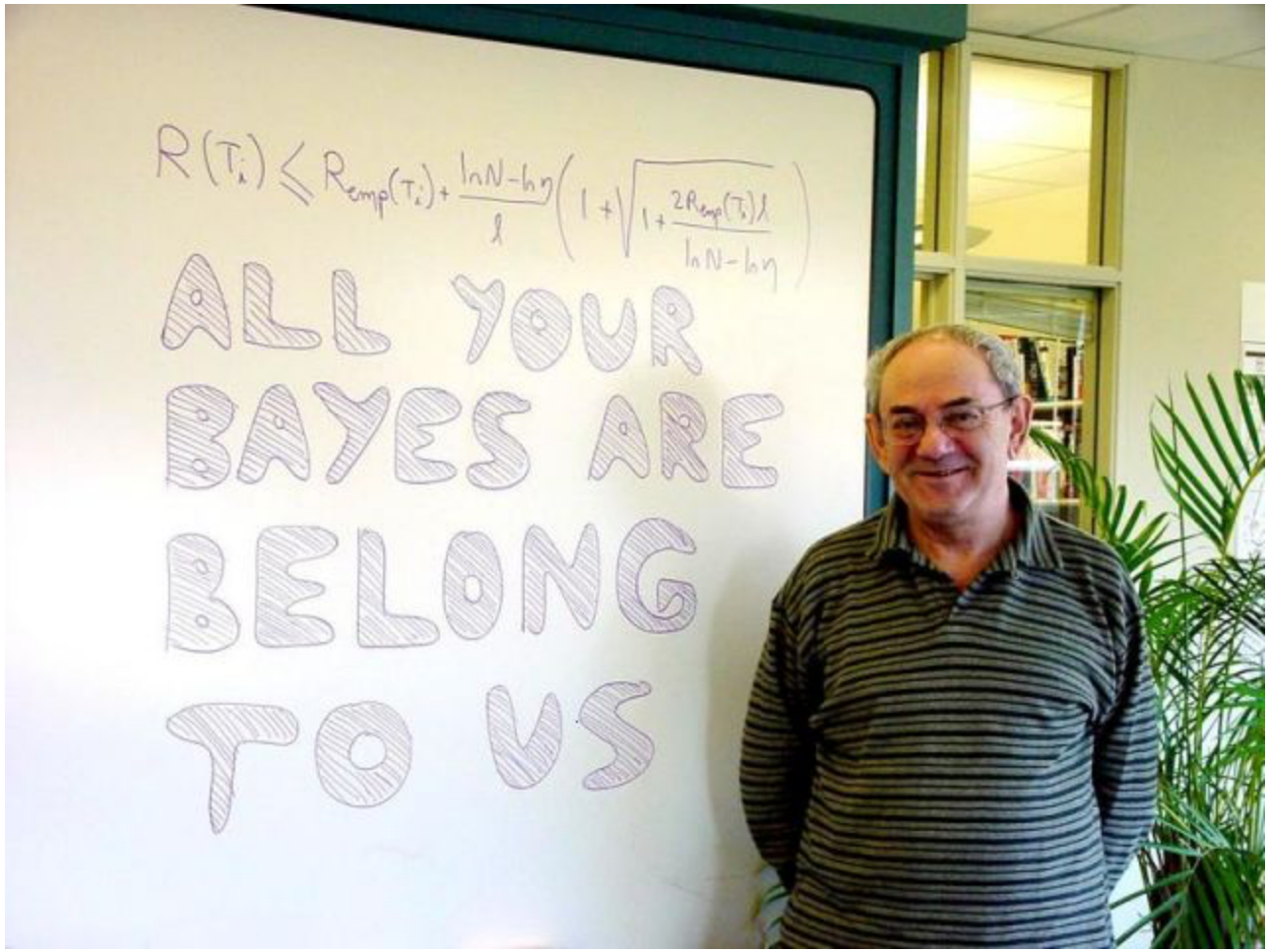
$$R(\alpha) \leq R_{emp}(\alpha) + \underbrace{\sqrt{\frac{h(\log(2N/h)+1) - \log(\eta/4)}{N}}}_{\text{"VC confidence"}}$$



VC dimension

“There’s nothing more practical than a good theory”



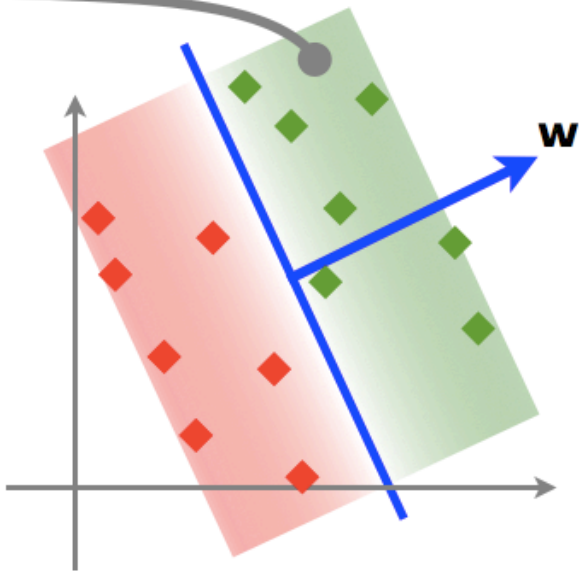


“There’s nothing more practical than a good theory”

SVMs in computer vision

bicycle?

x



linear predictor

$$F(\mathbf{x}) = \langle \mathbf{w}, \mathbf{x} \rangle$$

Slide credit: A. Vedaldi

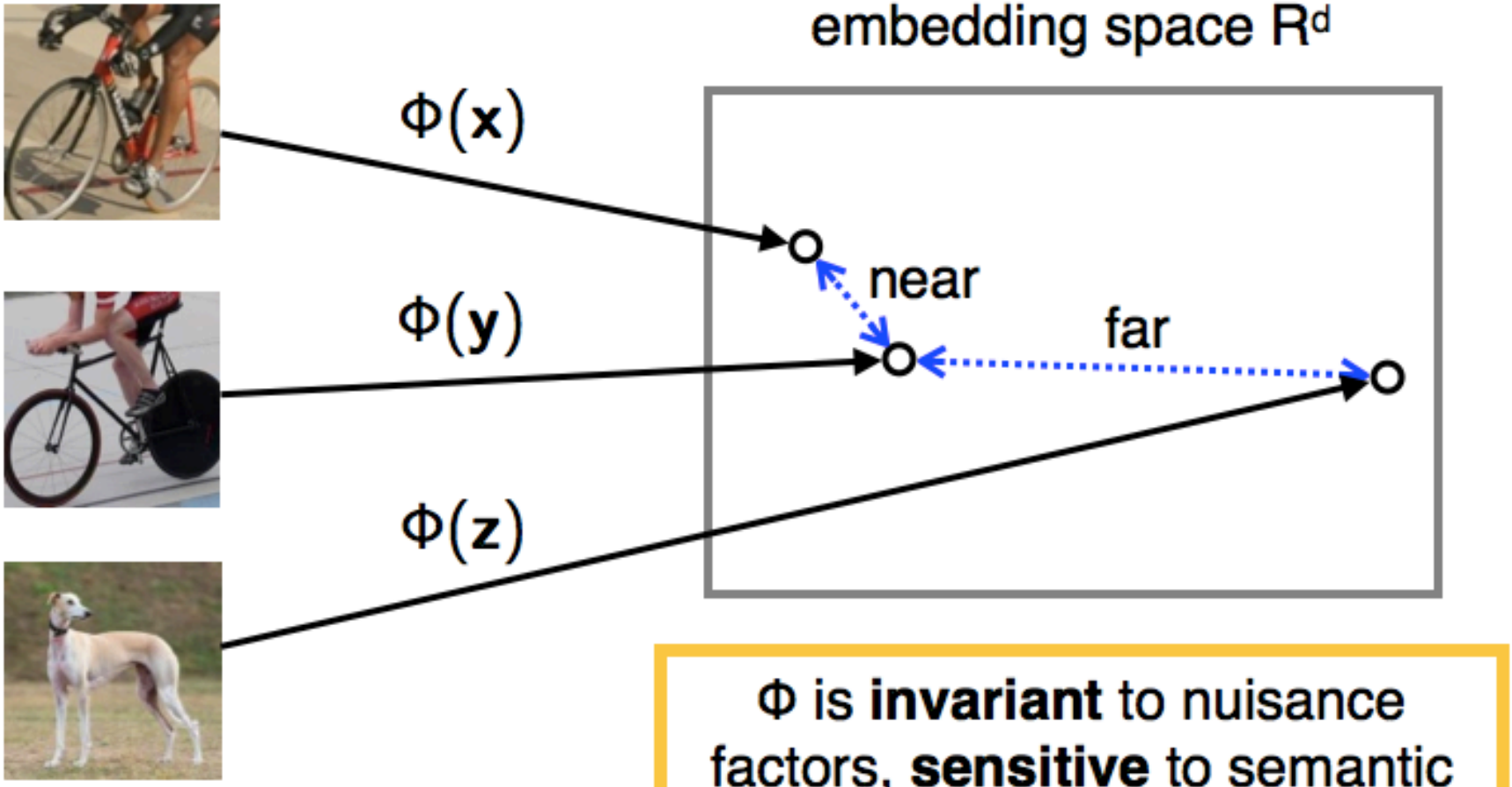
Image features

$\mathbf{x}$ 

encoder Φ → representation $\Phi(\mathbf{x}) \in \mathbb{R}^d$

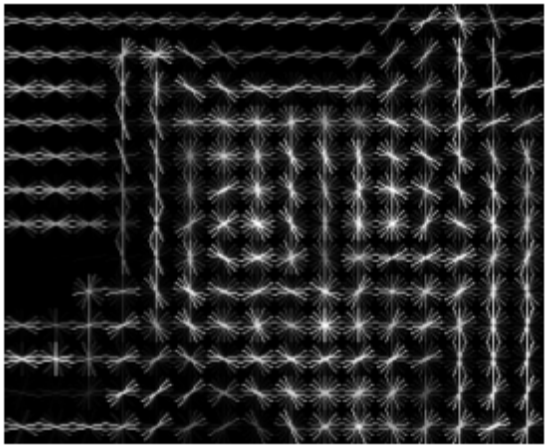
Slides: A. Vedaldi, http://www.robots.ox.ac.uk/~vedaldi/assets/teach/vedaldi14bmvc-tutorial.pdf

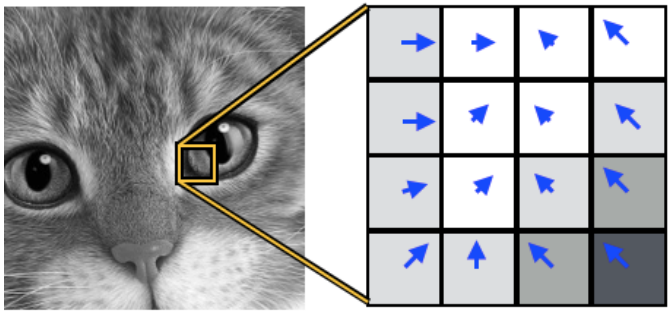
Desirable feature properties

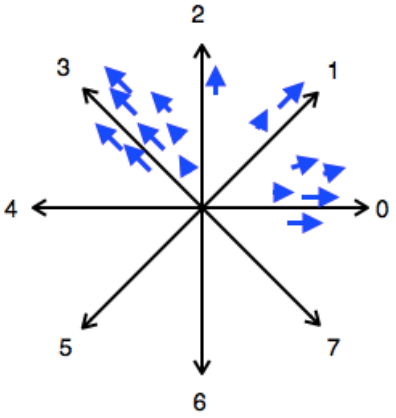


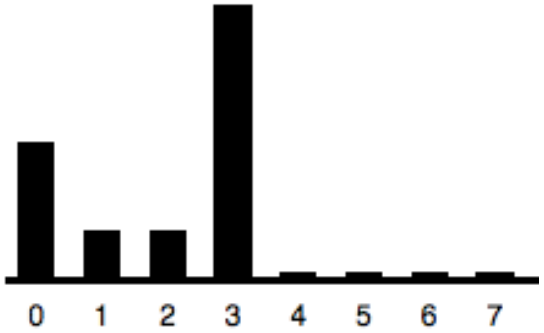
Slide credit: A. Vedaldi

Histogram of Gradient (HOG)/SIFT Features





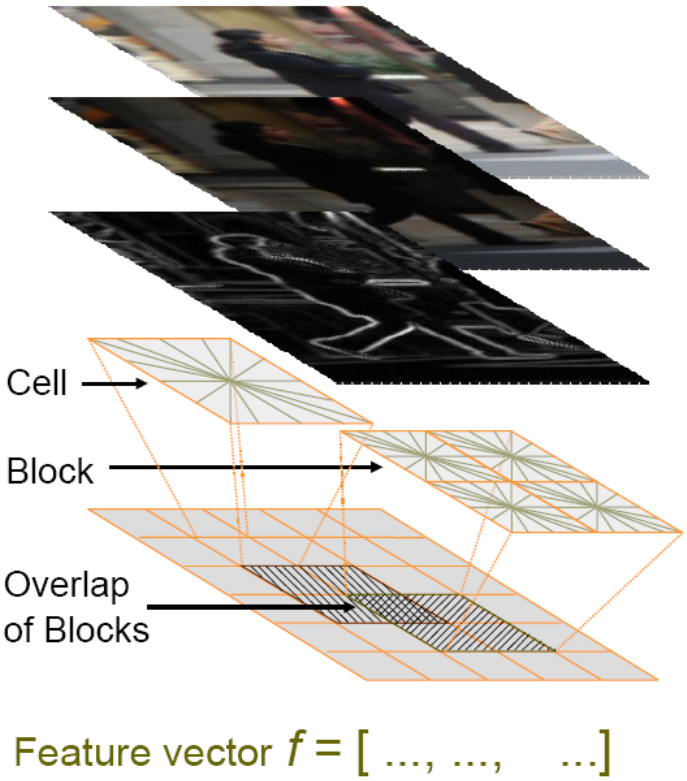




Slide credit: A. Vedaldi

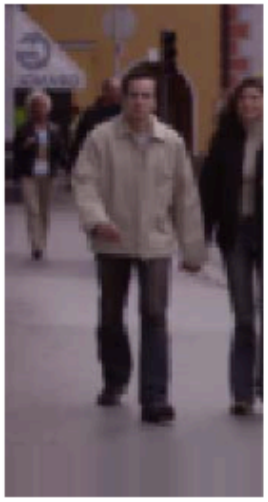
Dalal and Triggs, ICCV 2005

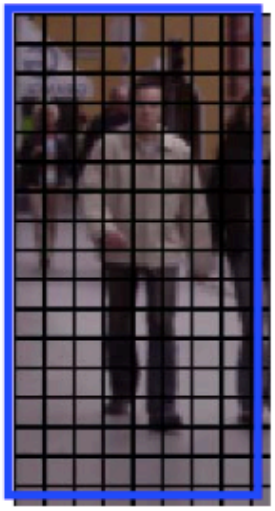
- Histogram of Oriented Gradient (HOG) features
- Highly accurate detection using linear SVM



HOG features for pedestrians

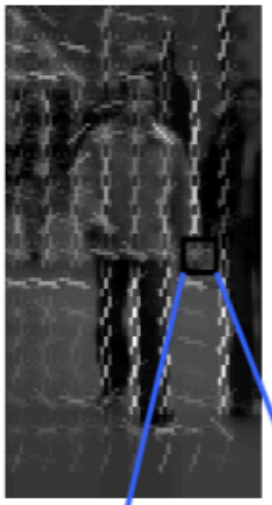
image



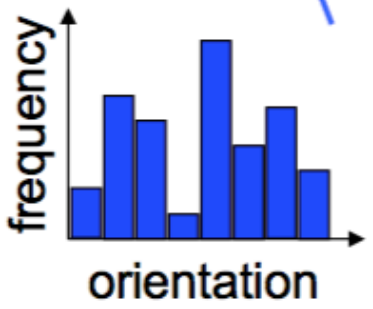


dominant direction

HOG

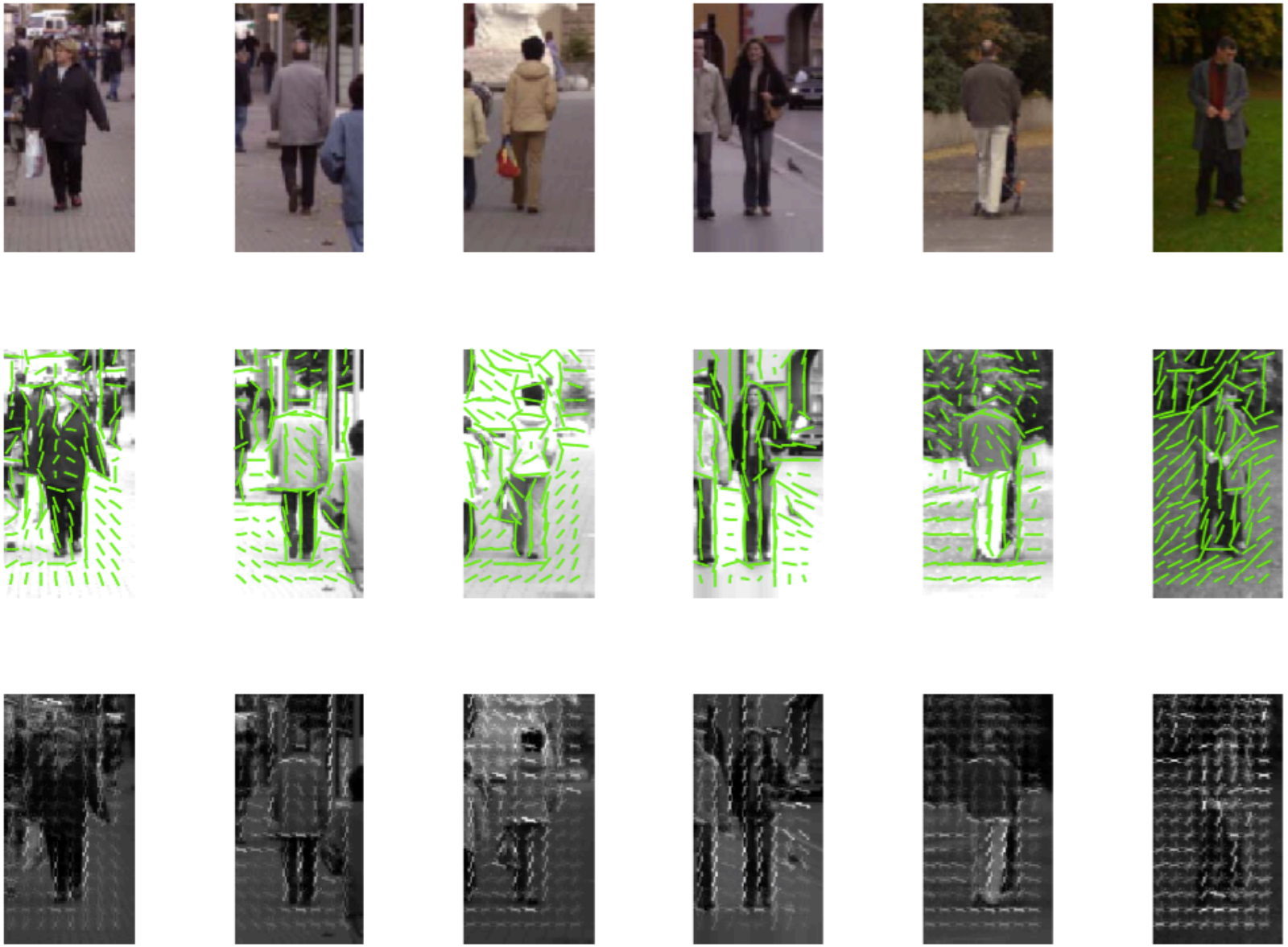


- tile window into 8 x 8 pixel cells
- each cell represented by HOG



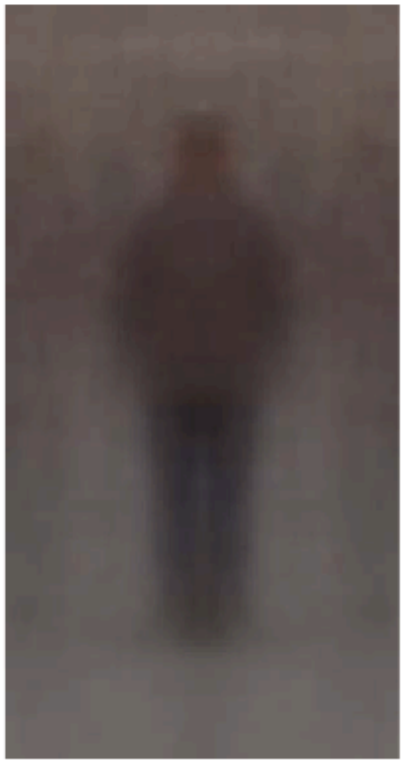
Feature vector dimension = 16 x 8 (for tiling) x 8 (orientations) = 1024

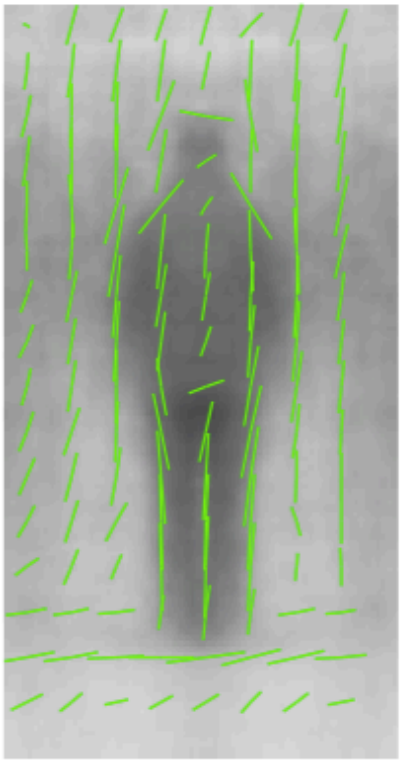
SVMs and Pedestrians

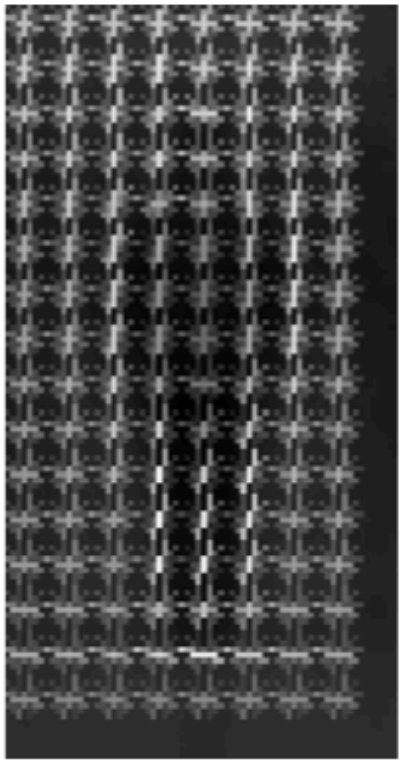


SVMs and Pedestrians

Averaged examples





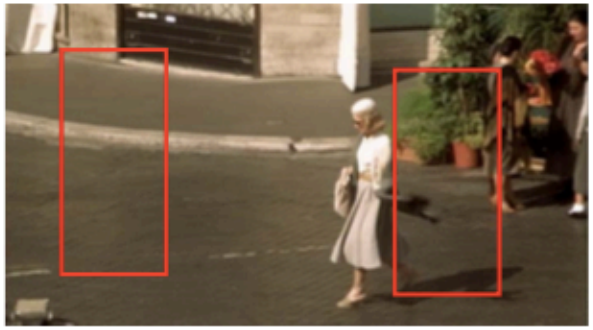


SVMs and Pedestrians

- Positive data – 1208 positive window examples




- Negative data – 1218 negative window examples (initially)

Training (Learning)

- Represent each example window by a HOG feature vector

$$\mathbf{x}_i \in \mathbb{R}^d, \text{ with } d = 1024$$

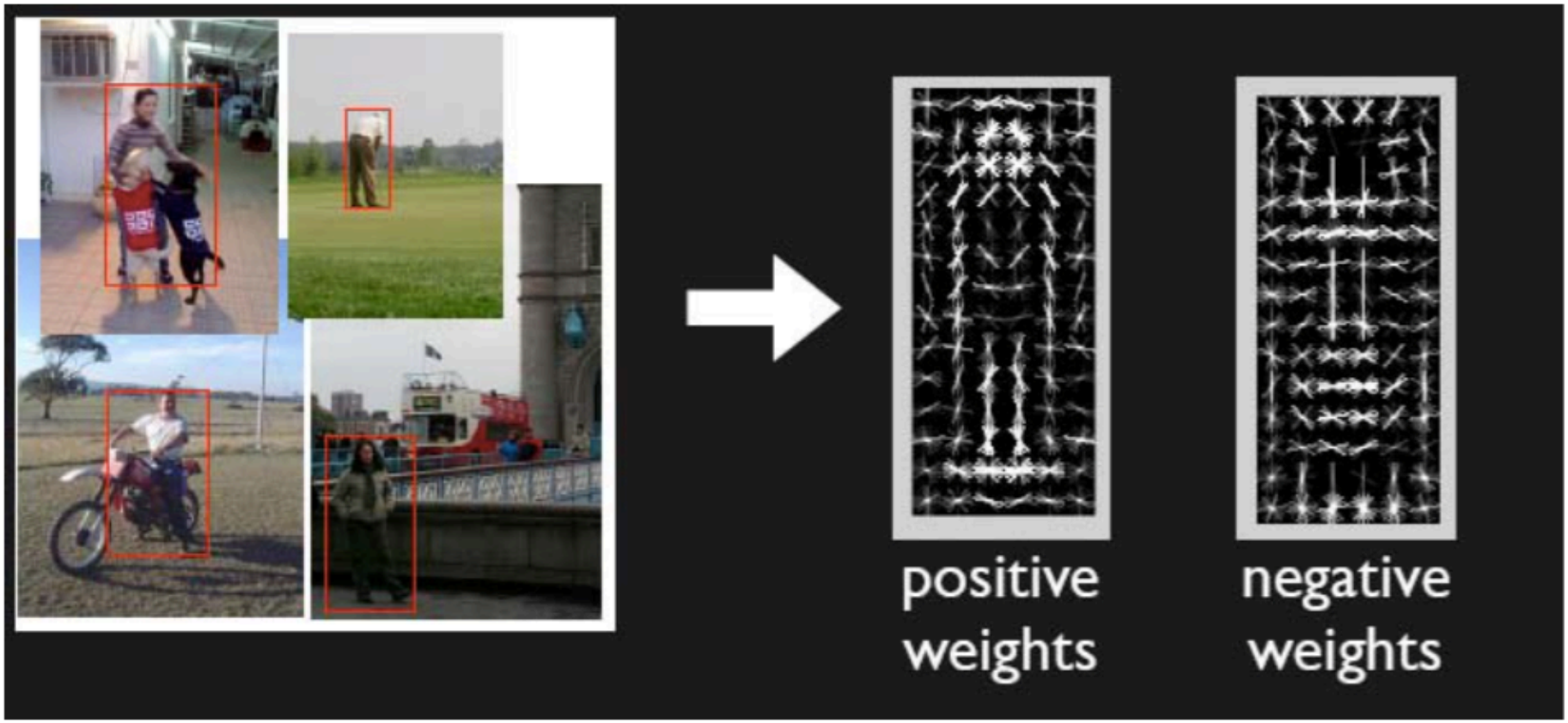
- Train a SVM classifier

Testing (Detection)

- Sliding window classifier

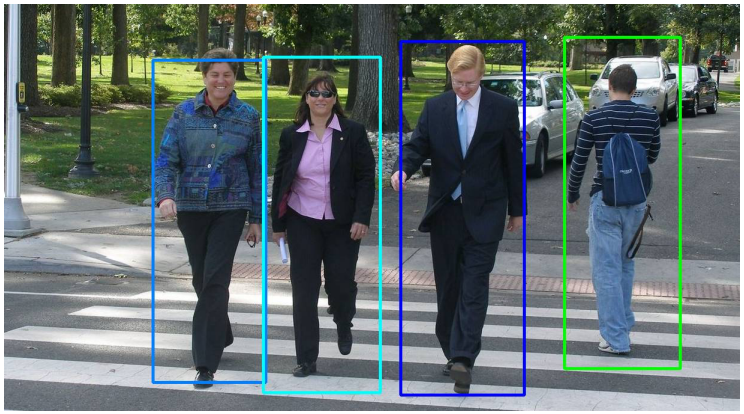
$$f(x) = \sum_i \alpha_i y_i (\mathbf{x}_i{}^\top \mathbf{x}) + b$$

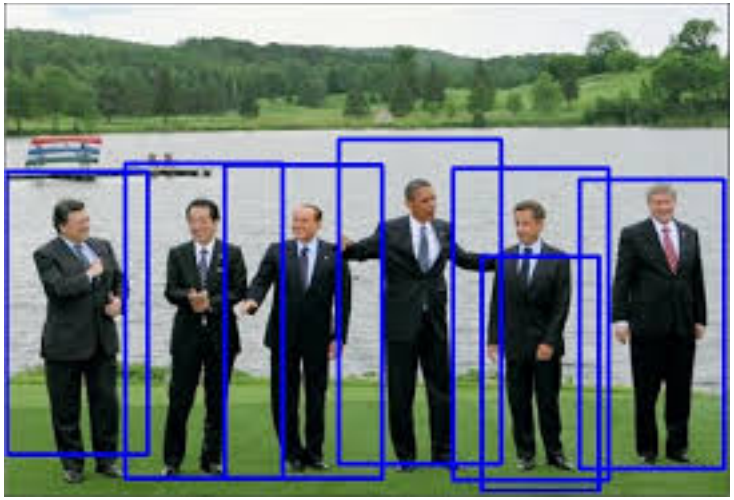
$$f(\mathbf{x}) = \mathbf{w}^\top \mathbf{x} + b$$

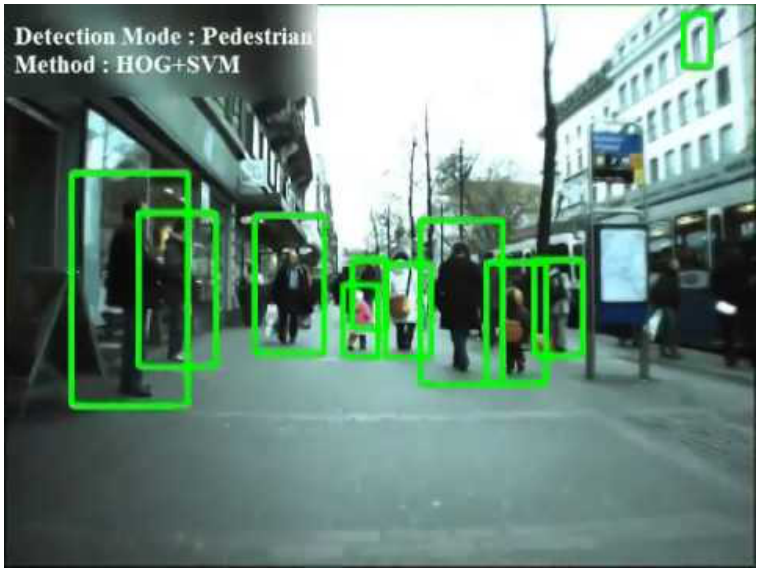


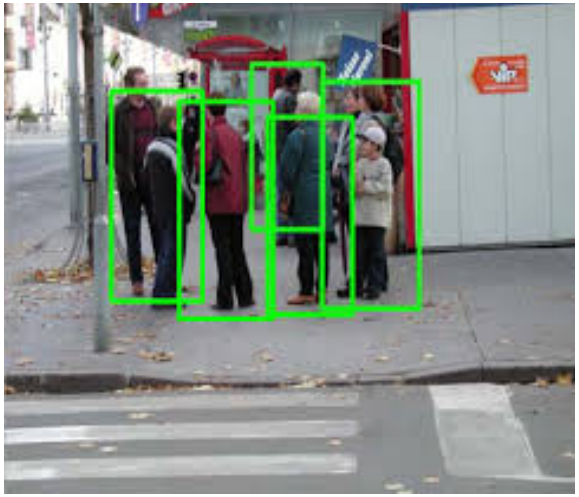
Dalal and Triggs, CVPR 2005

Pedestrian detection: almost done in 2005

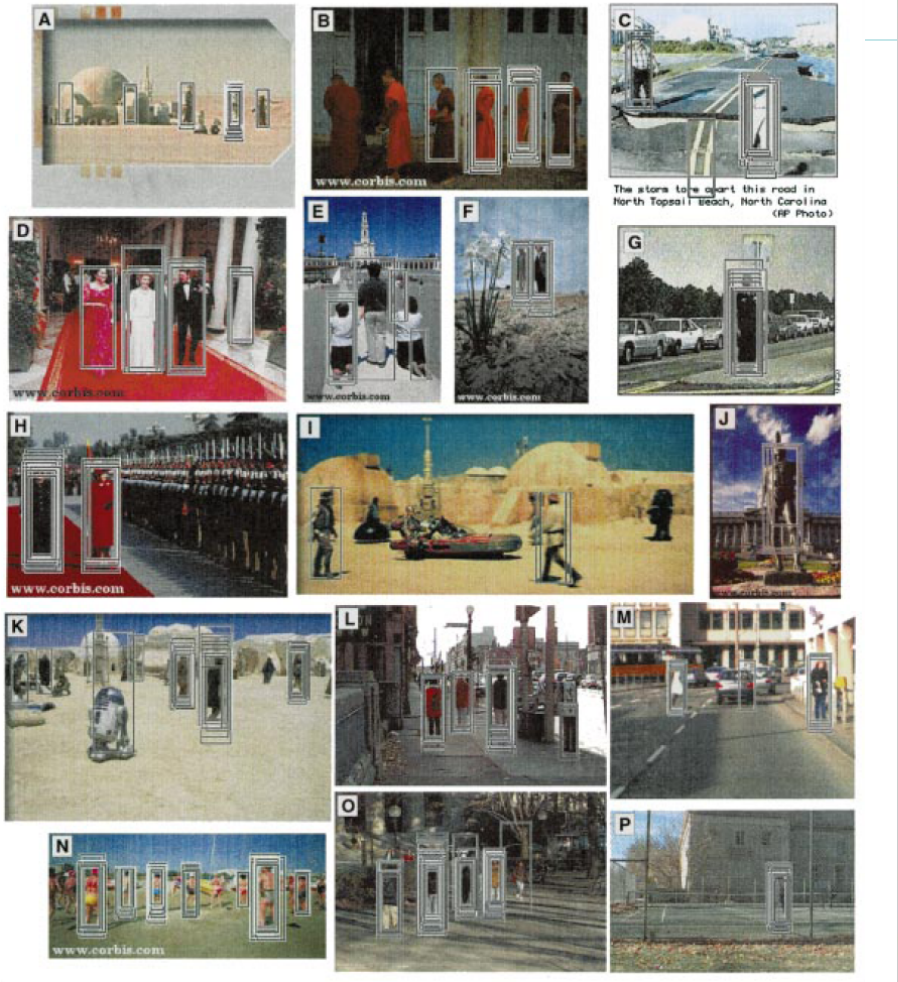


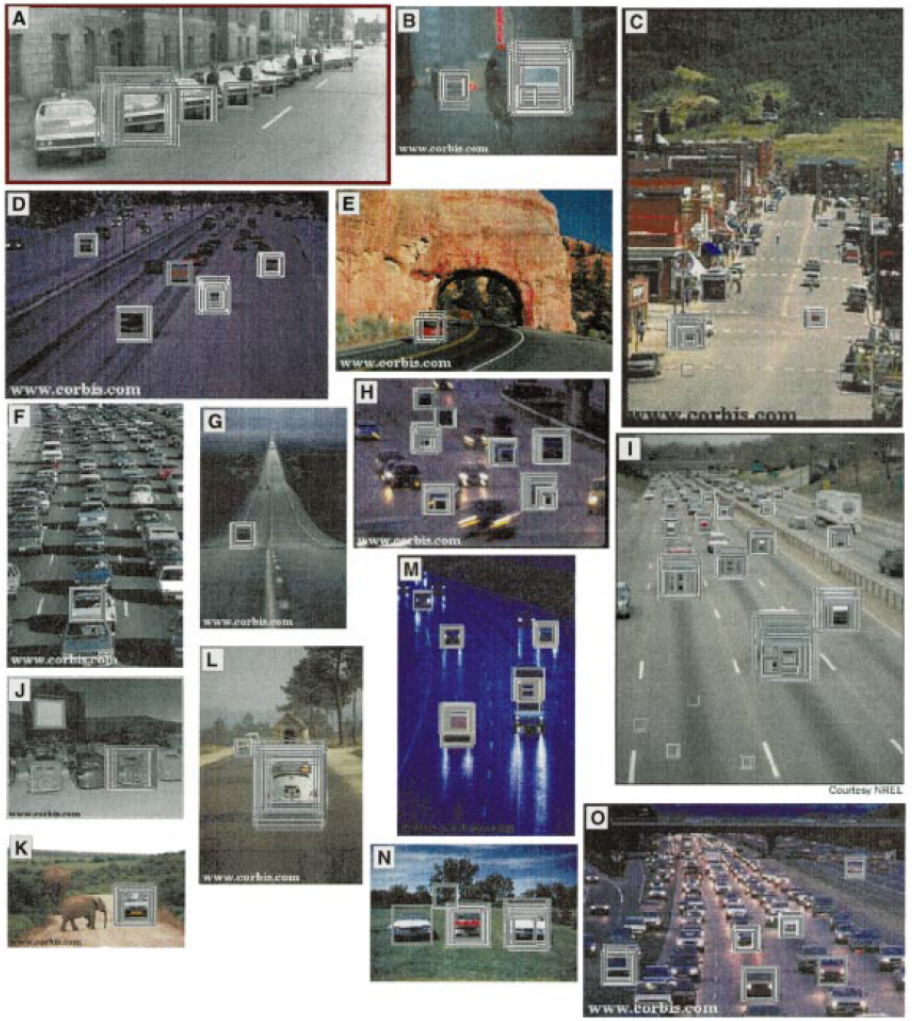




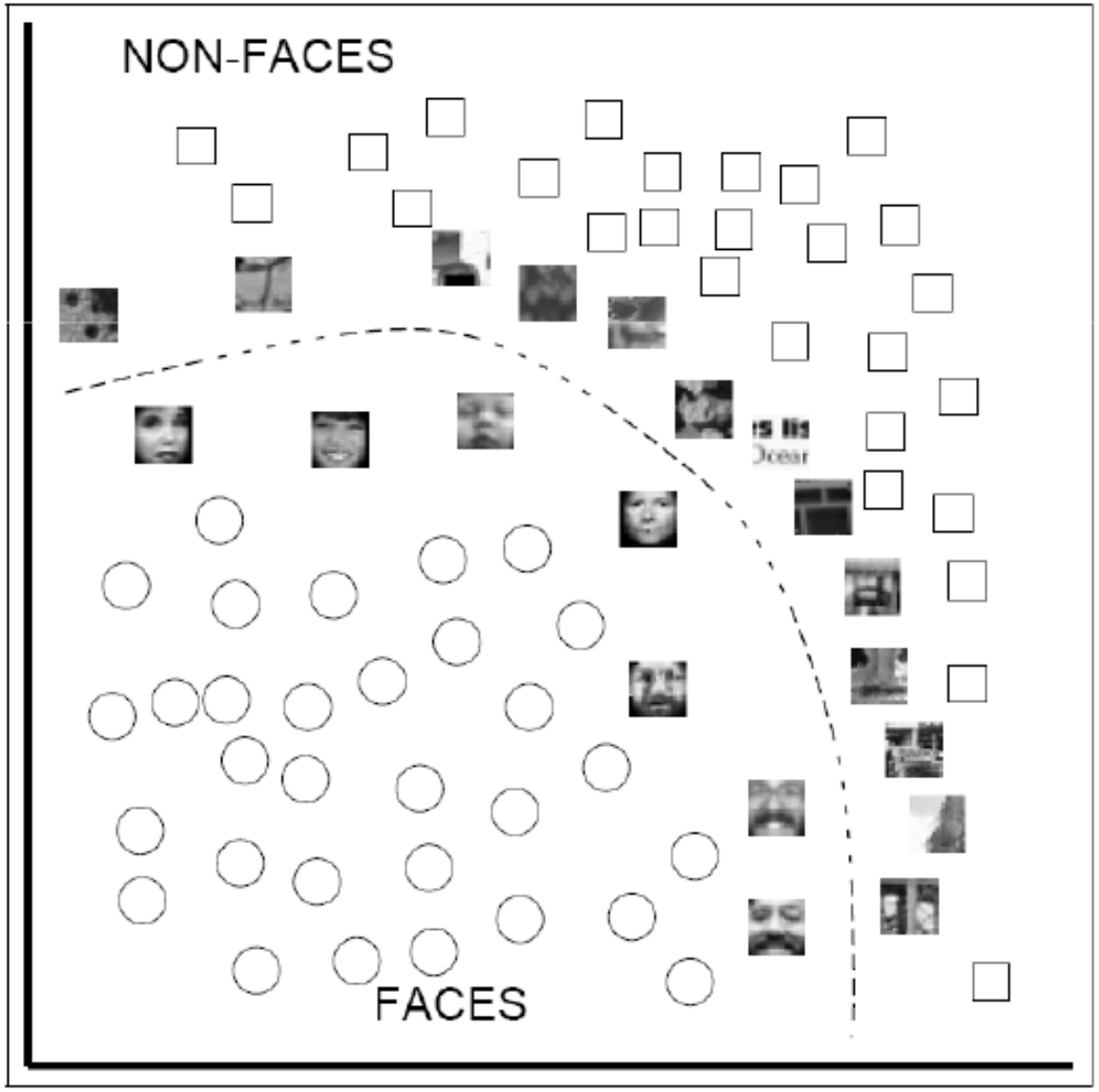


Papageorgiou & Poggio (1998)





Support vectors for Faces (P&P 98)



Lecture outline

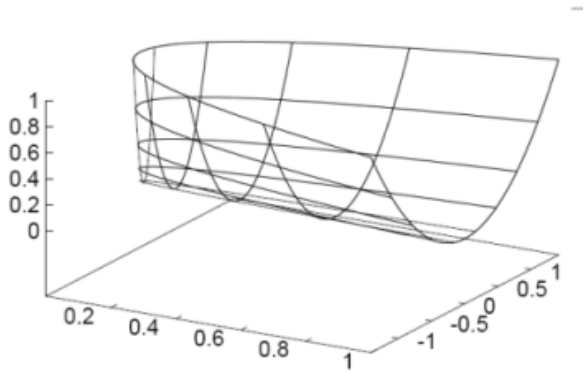
Recap

Large margins and generalization

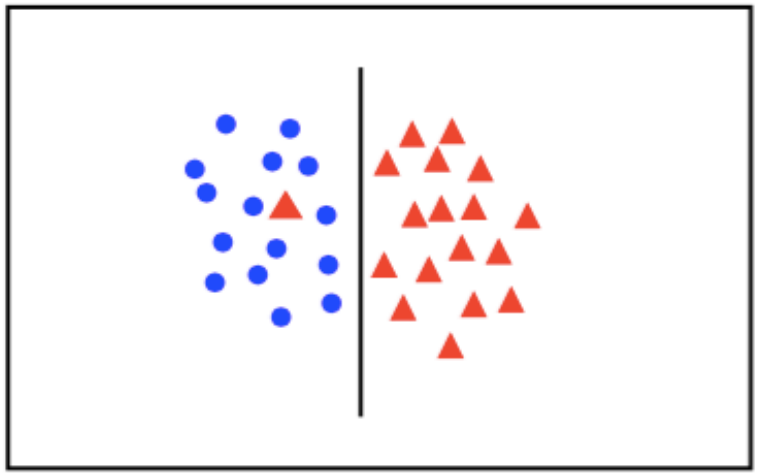
Optimization

Kernels

Applications to vision



Non-separable data

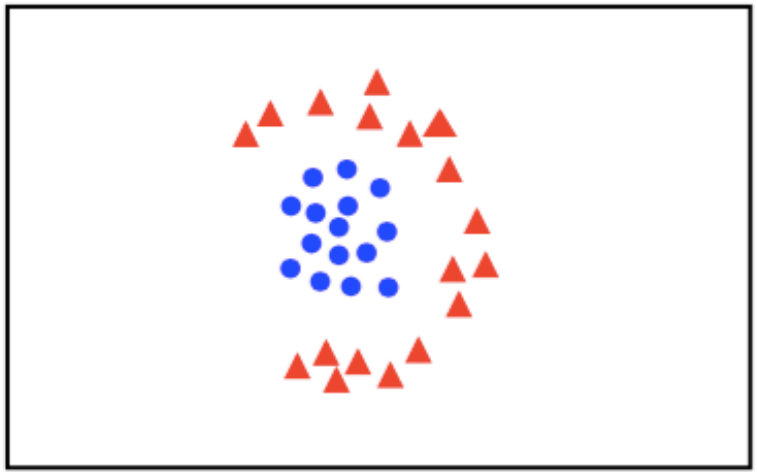


- introduce slack variables

$$\min_{\mathbf{w}\in\mathbb{R}^d,\xi_i\in\mathbb{R}^+} ||\mathbf{w}||^2 + C\sum_i^N \xi_i$$

subject to

$$y_i\left(\mathbf{w}^\top\mathbf{x}_i + b\right) \geq 1 - \xi_i \text{ for } i = 1\ldots N$$

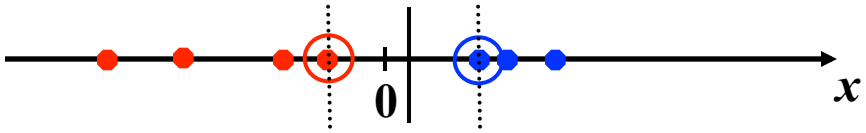


- linear classifier not appropriate

??

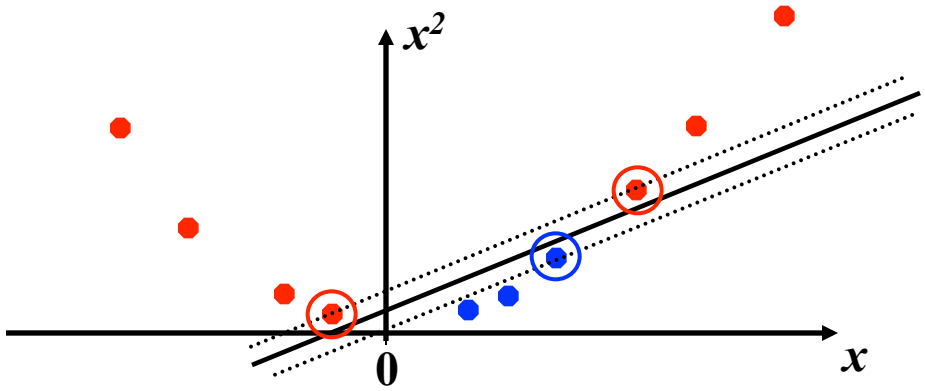
Non-linear SVMs

- Datasets that are linearly separable (with some noise) work out great:



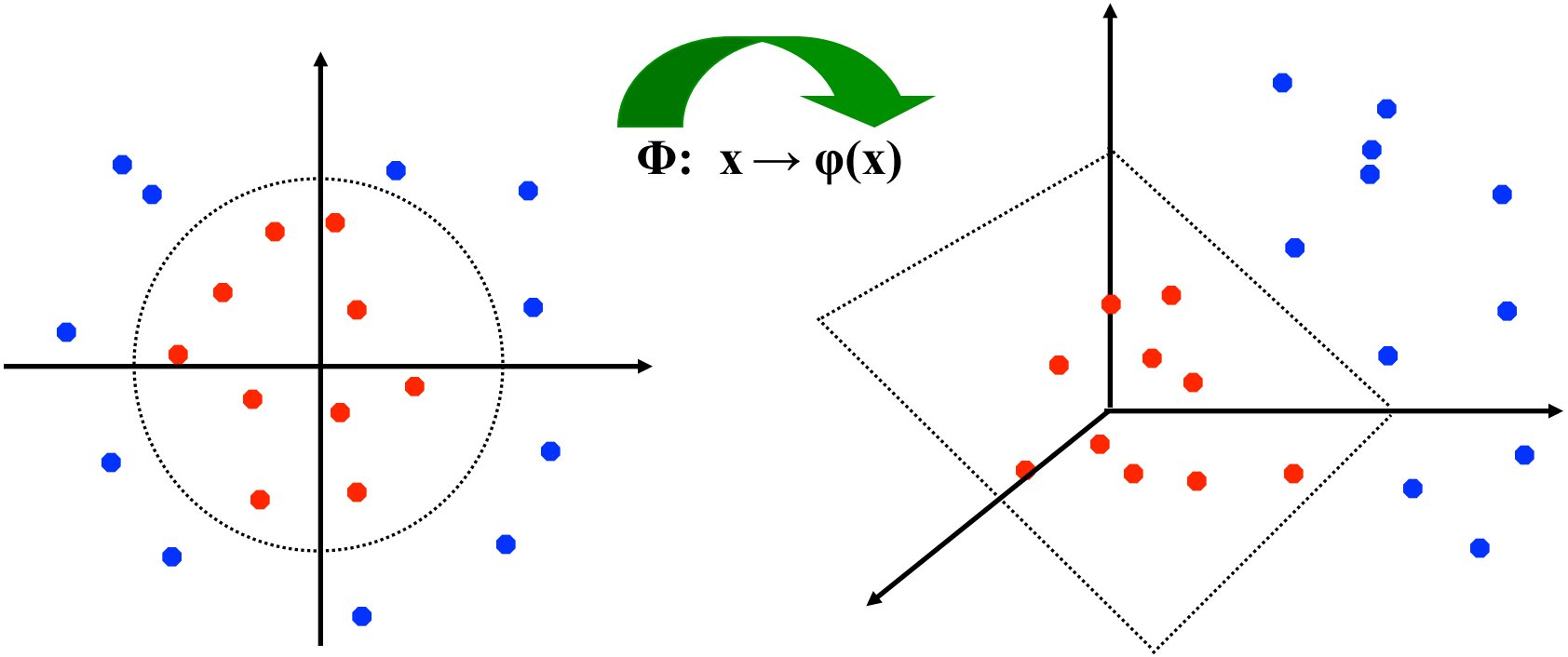
- But what are we going to do if the dataset is just too hard?

- How about … mapping data to a higher-dimensional space:

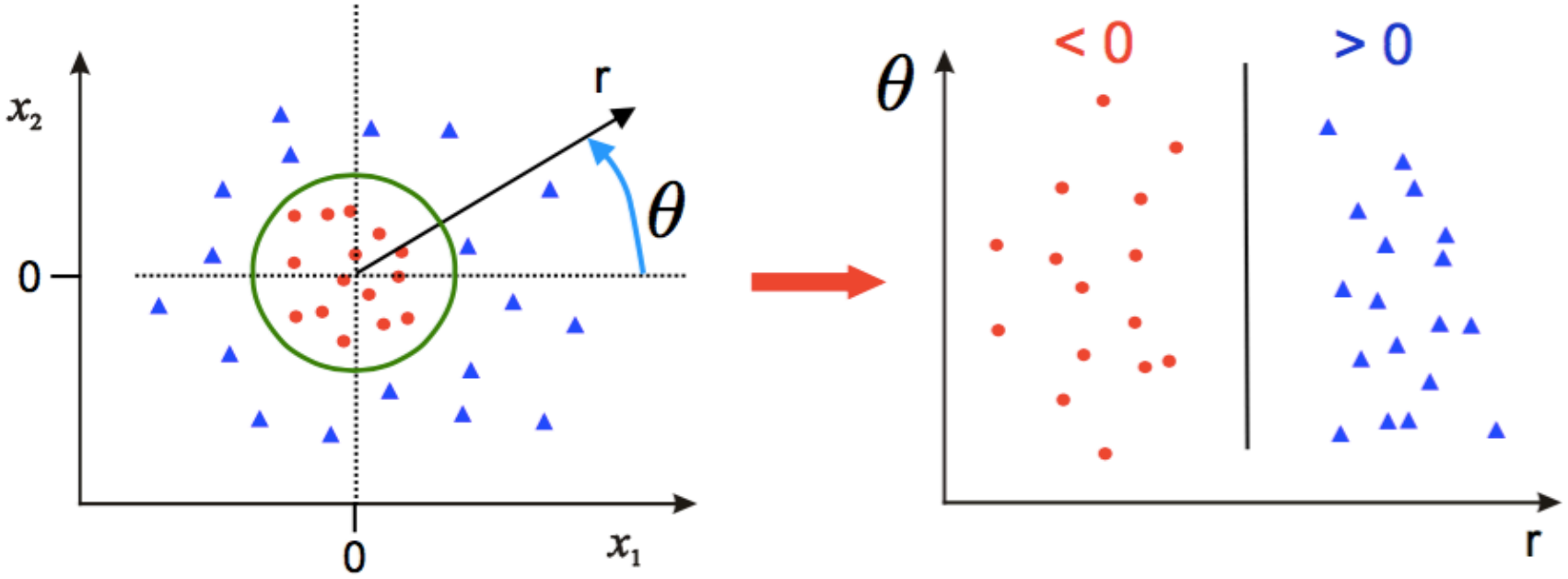


Non-linear SVMs: Feature spaces

- General idea: the original feature space can always be mapped to some higher-dimensional feature space where the training set is separable:



Solution by inspection: hand-crafted features

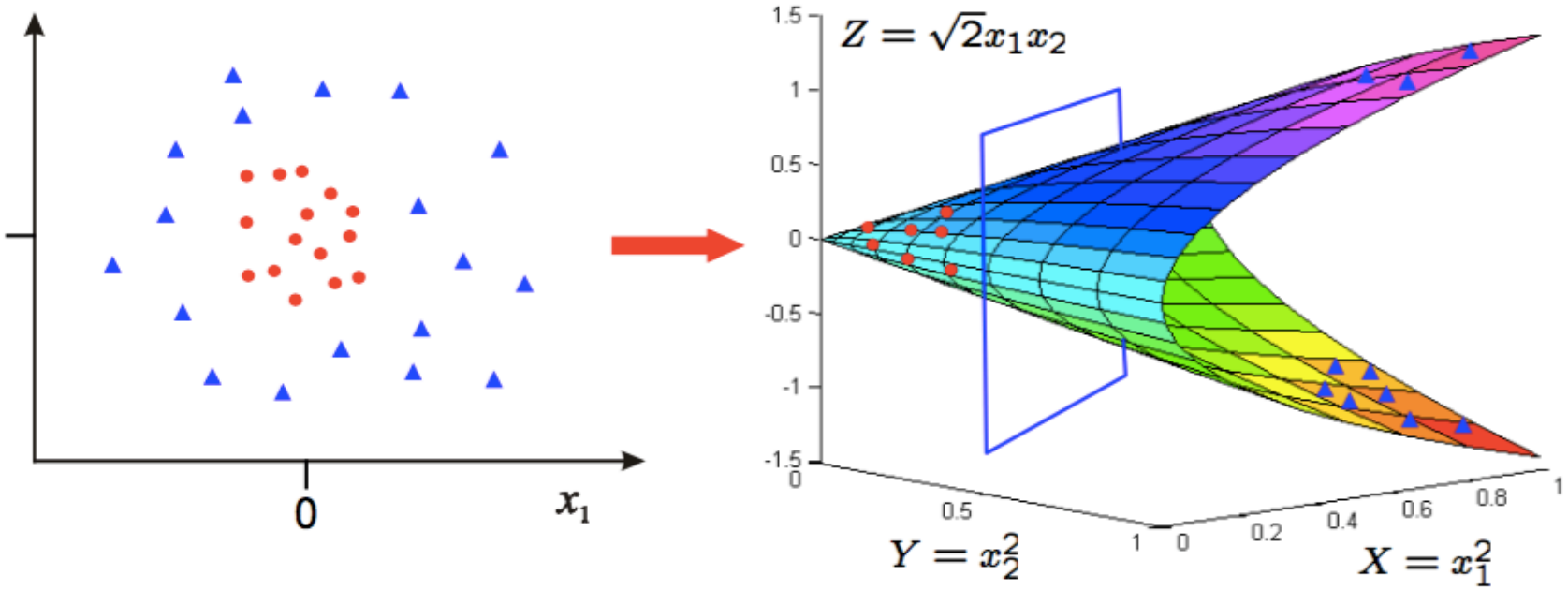


- Data is linearly separable in polar coordinates
- Acts non-linearly in original space

$$\Phi : \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} \rightarrow \begin{pmatrix} r \\ \theta \end{pmatrix} \qquad \mathbb{R}^2 \rightarrow \mathbb{R}^2$$

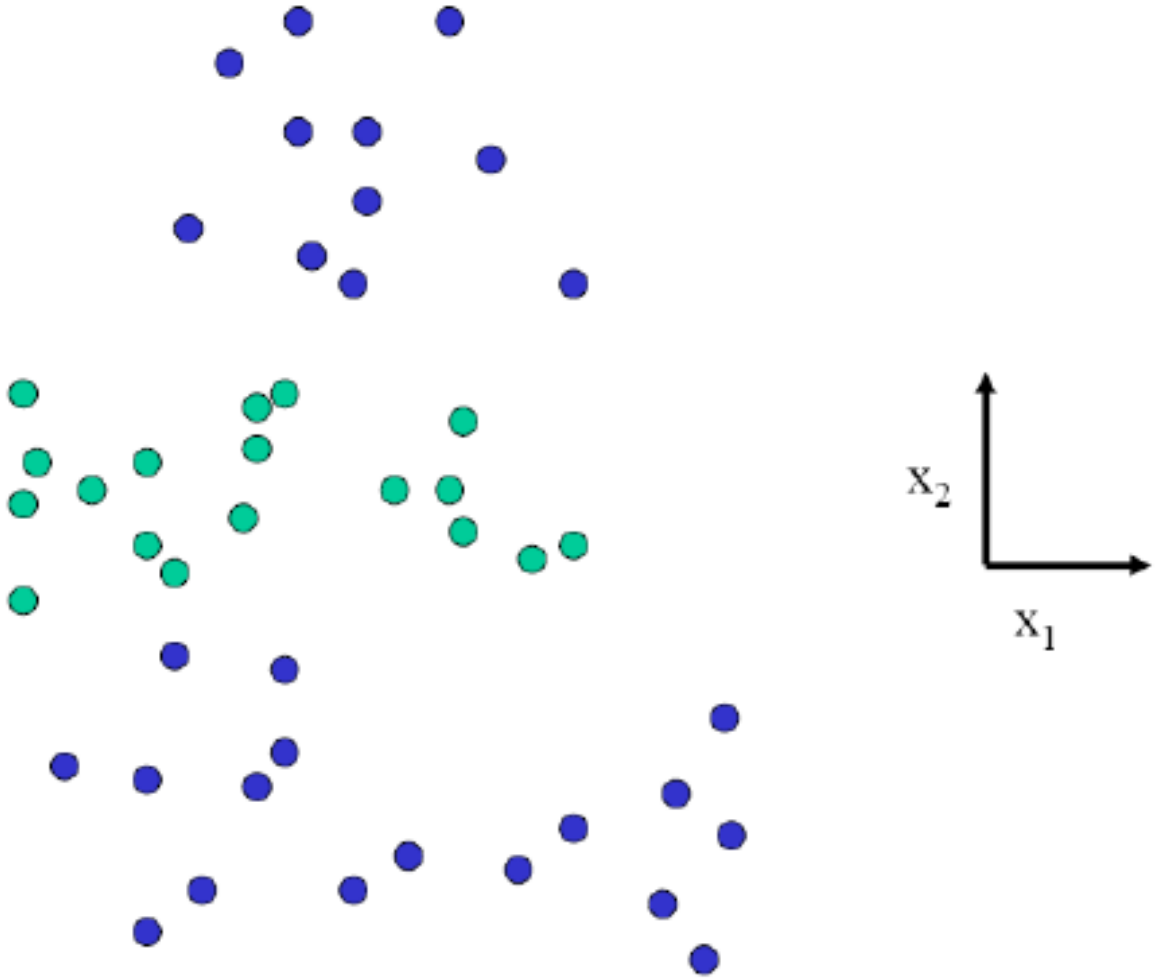
More general method

$$\Phi : \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} \rightarrow \begin{pmatrix} x_1^2 \\ x_2^2 \\ \sqrt{2}x_1x_2 \end{pmatrix} \qquad \mathbb{R}^2 \rightarrow \mathbb{R}^3$$

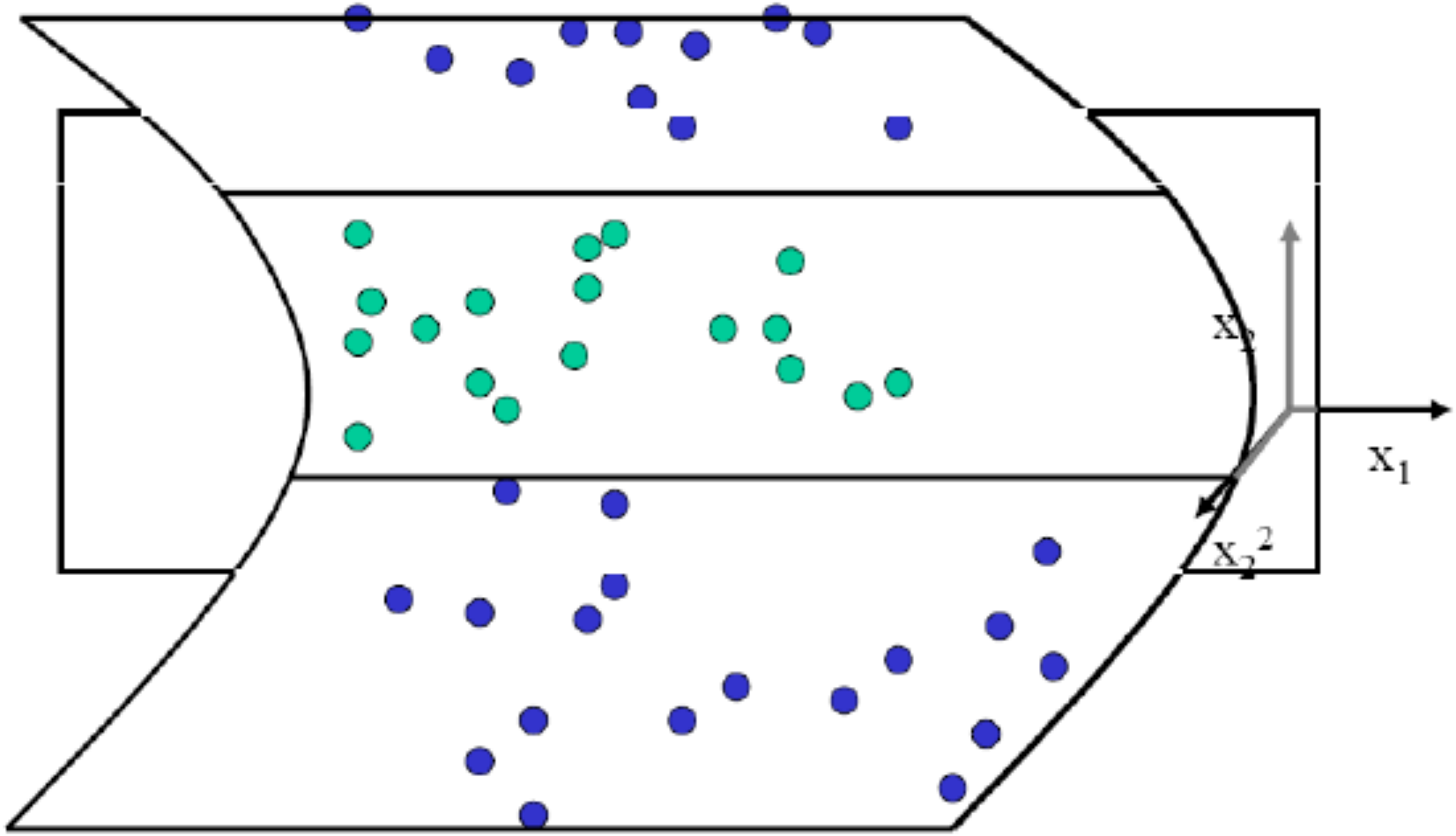


- Data is linearly separable in 3D
- This means that the problem can still be solved by a linear classifier

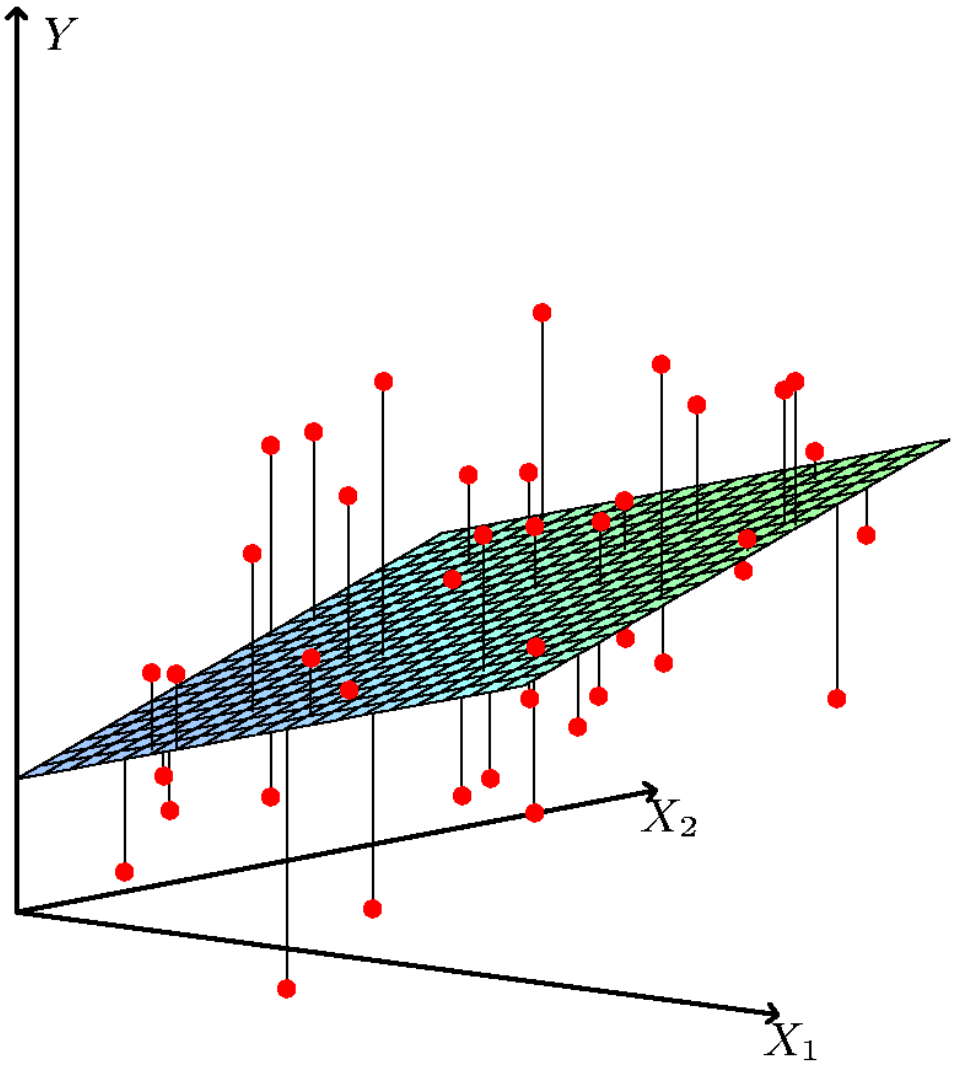
Nonseparable in 2D



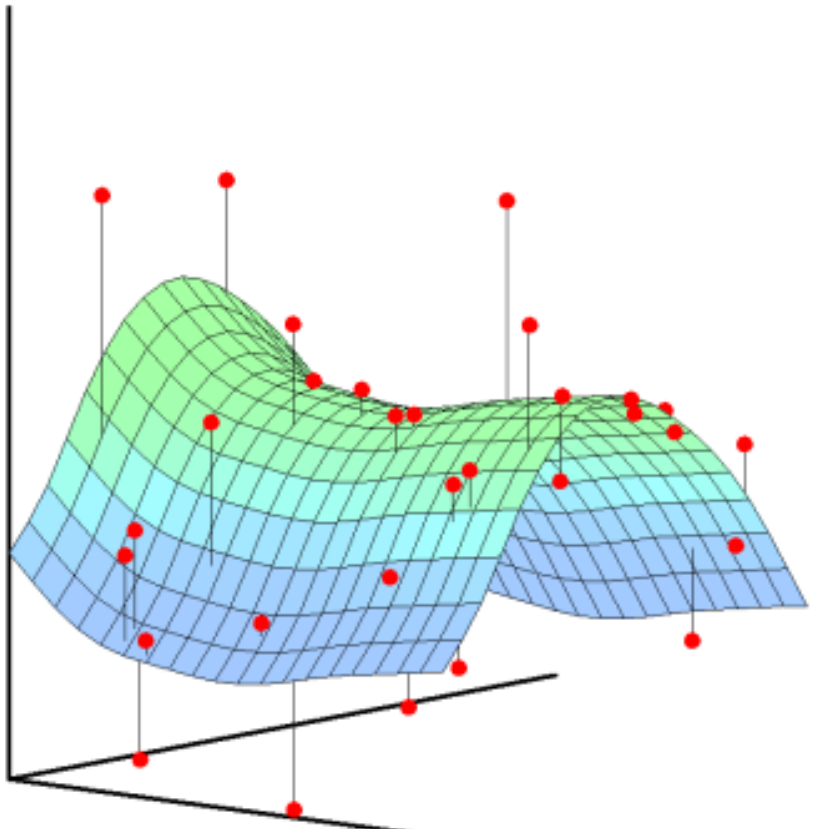
Separable in 3D



Linear regression



Nonlinear regression



$$\mathbf{x} \rightarrow \boldsymbol{\phi}(\mathbf{x}) = \begin{bmatrix} \phi_1(\mathbf{x}) \\ \vdots \\ \phi_M(\mathbf{x}) \end{bmatrix}$$

Example: second-order polynomials

$$\mathbf{x} = (x_1, x_2)$$

$$\boldsymbol{\phi}(\mathbf{x}) = \begin{bmatrix} 1 \\ x_1 \\ x_2 \\ (x_1)^2 \\ (x_2)^2 \\ x_1 x_2 \end{bmatrix}$$

$$\langle \mathbf{w}, \boldsymbol{\phi}(\mathbf{x}) \rangle = w_0 + w_1 x_1 + w_2 x_2 + w_3 x_1^2 + w_4 x_2^2 + w_5 x_1 x_2$$

Non-linear Classifiers

So far, decision is based on the sign of $y = \mathbf{w}^T \mathbf{x}$

Use non-linear transformation, φ(x) of our data, x

e.g. $\mathbf{x} = (x_1, x_2)$ $\quad \boldsymbol{\phi}(\mathbf{x}) = \begin{bmatrix} 1 \\ x_1 \\ x_2 \\ (x_1)^2 \\ (x_2)^2 \\ x_1 x_2 \end{bmatrix}$

Discriminant:

$$\langle \mathbf{w}, \boldsymbol{\phi}(\mathbf{x}) \rangle = w_0 + w_1 x_1 + w_2 x_2 + w_3 x_1^2 + w_4 x_2^2 + w_5 x_1 x_2$$

Non-linear in x, linear in φ(x)

Dual form of SVM & kernel trick

Optimization:

$$\min_{\boldsymbol{\alpha}} \sum_{i=1}^{N} \sum_{j=1}^{N} \alpha^i \alpha^j y^i y^j \langle \mathbf{x}^i, \mathbf{x}^j \rangle$$

$$\text{s.t.}: \; y^i \left(\sum_{j=1}^{N} \alpha^j y^j \langle \mathbf{x}^j, \mathbf{x}^i \rangle + b \right) \geq 1, \quad \forall i$$

$$\boldsymbol{\alpha} \in \mathbb{R}^N \rightarrow O(N^3)$$

Primal and dual classifier forms:

$$f(\mathbf{x}) = \langle \mathbf{w}, \mathbf{x} \rangle + b = \sum_{i=1}^{N} \alpha^i y^i \langle \mathbf{x}^i, \mathbf{x} \rangle + b$$

What if we replace **x** with **φ(x)?**

Everything involves only inner products!

Rewrite everything in terms of Kernel

$$K(\mathbf{x}, \mathbf{y}) = \langle \boldsymbol{\phi}(\mathbf{x}), \boldsymbol{\phi}(\mathbf{y}) \rangle$$

Dual form of SVM & kernel trick

Optimization: $$\min_{\boldsymbol{\alpha}} \sum_{i=1}^{N} \sum_{j=1}^{N} \alpha^i \alpha^j y^i y^j \; K\left(\mathbf{x}^i, \mathbf{x}^j\right)$$

$$\text{s.t.}: \; y^i \left(\sum_{j=1}^{N} \alpha^j y^j K(\mathbf{x}^j, \mathbf{x}^i) + b \right) \geq 1, \quad i = 1, \ldots, N$$

Dual classifier form:

$$f(\mathbf{x}) = \sum_{i=1}^{N} \alpha^i y^i K(\mathbf{x}^i, \mathbf{x}) + b$$

$$= \sum_{\{i:\alpha^i \neq 0\}} w^i K(\mathbf{x}^i, \mathbf{x}) + b, \quad w^i = y^i \alpha^i$$

Compare with general nonlinear form: $$f(\mathbf{x}) = \sum_{k} w_k \phi_k(\mathbf{x})$$

N nonlinear functions – smart choice of sparse coefficients

`Kernel trick'

Consider: $\boldsymbol{\phi}(\mathbf{x}) = \begin{bmatrix} x_1^2 \\ x_2^2 \\ \sqrt{2}x_1x_2 \\ \sqrt{2}x_1 \\ \sqrt{2}x_2 \\ 1 \end{bmatrix}$

We then have: $\langle \boldsymbol{\phi}(\mathbf{x}), \boldsymbol{\phi}(\mathbf{y}) \rangle =$

$$= x_1^2y_1^2 + 2x_1x_2y_1y_2 + x_2^2y_2^2 + 2x_1y_1 + 2x_2y_2 + 1$$

$$= (x_1y_1 + x_2y_2 + 1)^2$$

$$= (\mathbf{x}^T\mathbf{y} + 1)^2 \quad \doteq K(\mathbf{x}, \mathbf{y})$$

Polynomial Kernel $K(\mathbf{x}, \mathbf{y}) = (\mathbf{x}^T\mathbf{y} + 1)^p$

Kernel: linear complexity in D (dimensions of **x,y**), constant in p

Feature space complexity: much higher

Condition for kernel trick: 'Mercer' kernel

- Given some arbitrary function $k(\mathbf{x}_i, \mathbf{x}_j)$, how do we know if it corresponds to a scalar product $\Phi(\mathbf{x}_i)^\top \Phi(\mathbf{x}_j)$ in some space?

- Mercer kernels: if $k(,)$ satisfies:
  - Symmetric $k(\mathbf{x}_i, \mathbf{x}_j) = k(\mathbf{x}_j, \mathbf{x}_i)$
  - Positive definite, $\boldsymbol{\alpha}^\top \mathtt{K} \boldsymbol{\alpha} \geq 0$ for all $\boldsymbol{\alpha} \in \mathbb{R}^N$, where $\mathtt{K}$ is the $N \times N$ Gram matrix with entries $\mathtt{K}_{ij} = k(\mathbf{x}_i, \mathbf{x}_j)$.

  then $k(,)$ is a valid kernel.

Mercer Kernel Examples

Linear kernel

$$K(\mathbf{x}, \mathbf{y}) = \mathbf{x}^T \mathbf{y}$$

Polynomial kernel

$$K(\mathbf{x}, \mathbf{y}) = (\mathbf{x}^T \mathbf{y} + 1)^p$$

Underlying feature dimension: $\begin{pmatrix} D+p \\ p-1 \end{pmatrix} = \frac{(D+p)!}{(D-1)!(p-1)!}$

$$x! = x \cdot (x-1) \cdot \ldots \cdot 2 \cdot 1$$

Radial Basis Function (a.k.a. Gaussian) kernel

$$K(\mathbf{x}, \mathbf{y}) = \exp\left(-\frac{1}{2\sigma^2}\|\mathbf{x} - \mathbf{y}\|^2\right)$$

Underlying feature dimension: **Infinite**

RBF kernel SVM

N = size of training data

$$f(\mathbf{x}) = \sum_{i}^{N} \alpha_i y_i k(\mathbf{x}_i, \mathbf{x}) + b$$

weight (may be zero)

support vector

Gaussian kernel $k(\mathbf{x}, \mathbf{x}') = \exp\left(-||\mathbf{x} - \mathbf{x}'||^2 / 2\sigma^2\right)$

Radial Basis Function (RBF) SVM

RBF kernel SVM (next week's assignment)

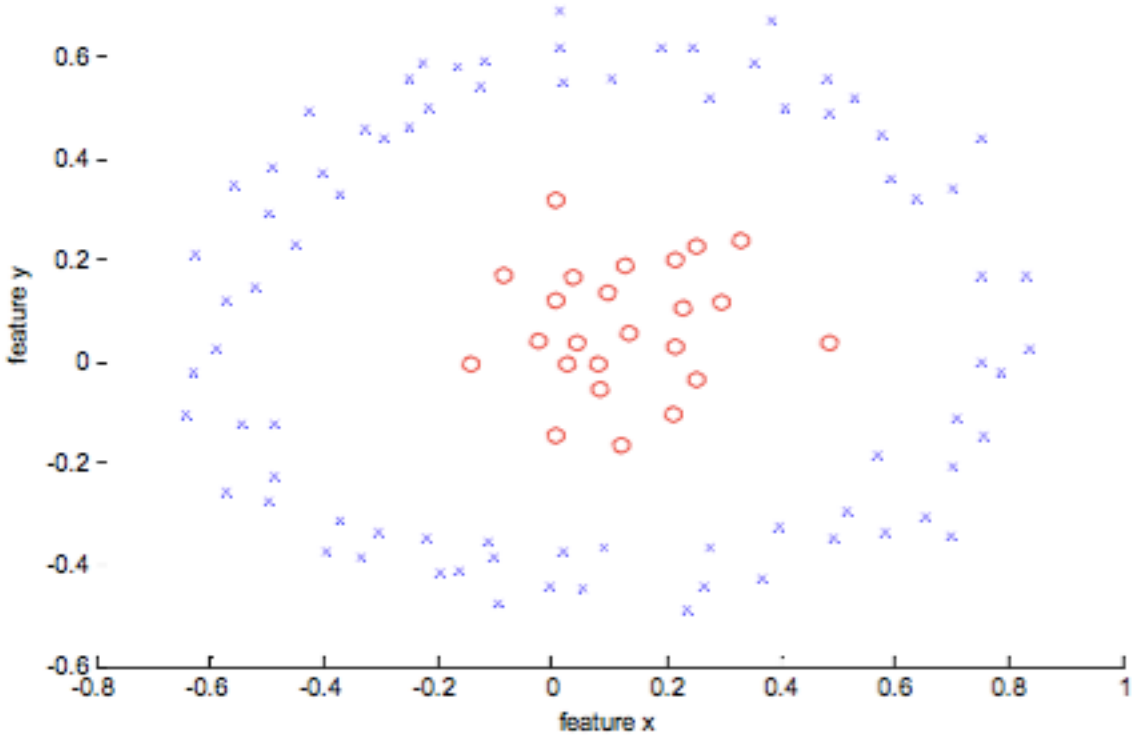
$$f(\mathbf{x}) = \sum_{i=1}^{N} \alpha^i y^i K(\mathbf{x}^i, \mathbf{x}) + b$$

$$= \sum_{\{i:\alpha^i \neq 0\}} \alpha^i y^i K(\mathbf{x}^i, \mathbf{x}) + b$$

$$= \sum_{\{i:\alpha^i \neq 0\}} w^i \exp\left(-\frac{1}{2\sigma^2}\|\mathbf{x}^i - \mathbf{x}\|_2^2\right) + b$$

Discriminant form: sum of bumps centered on training points

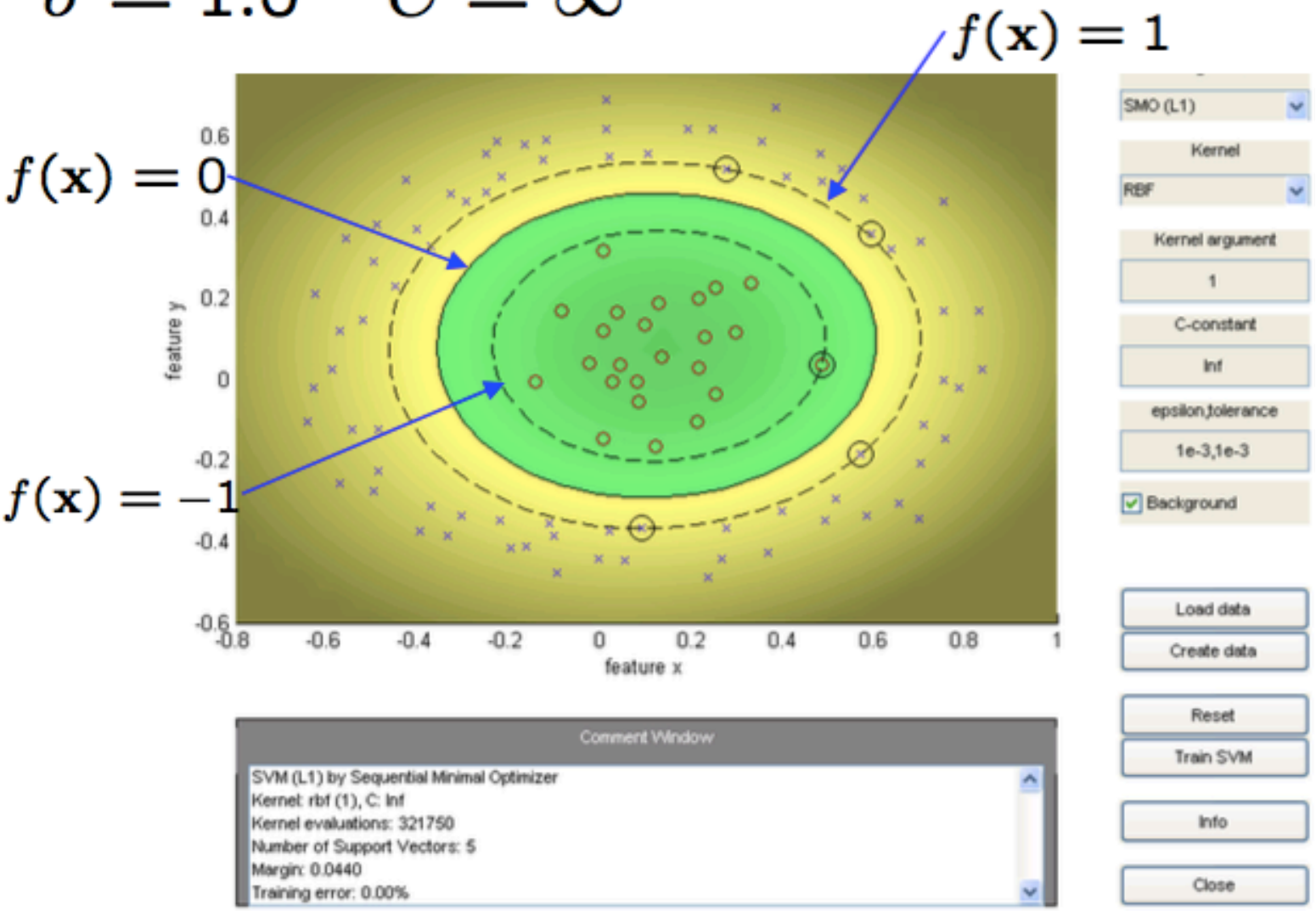
RBF-SVM example



- data is not linearly separable in original feature space

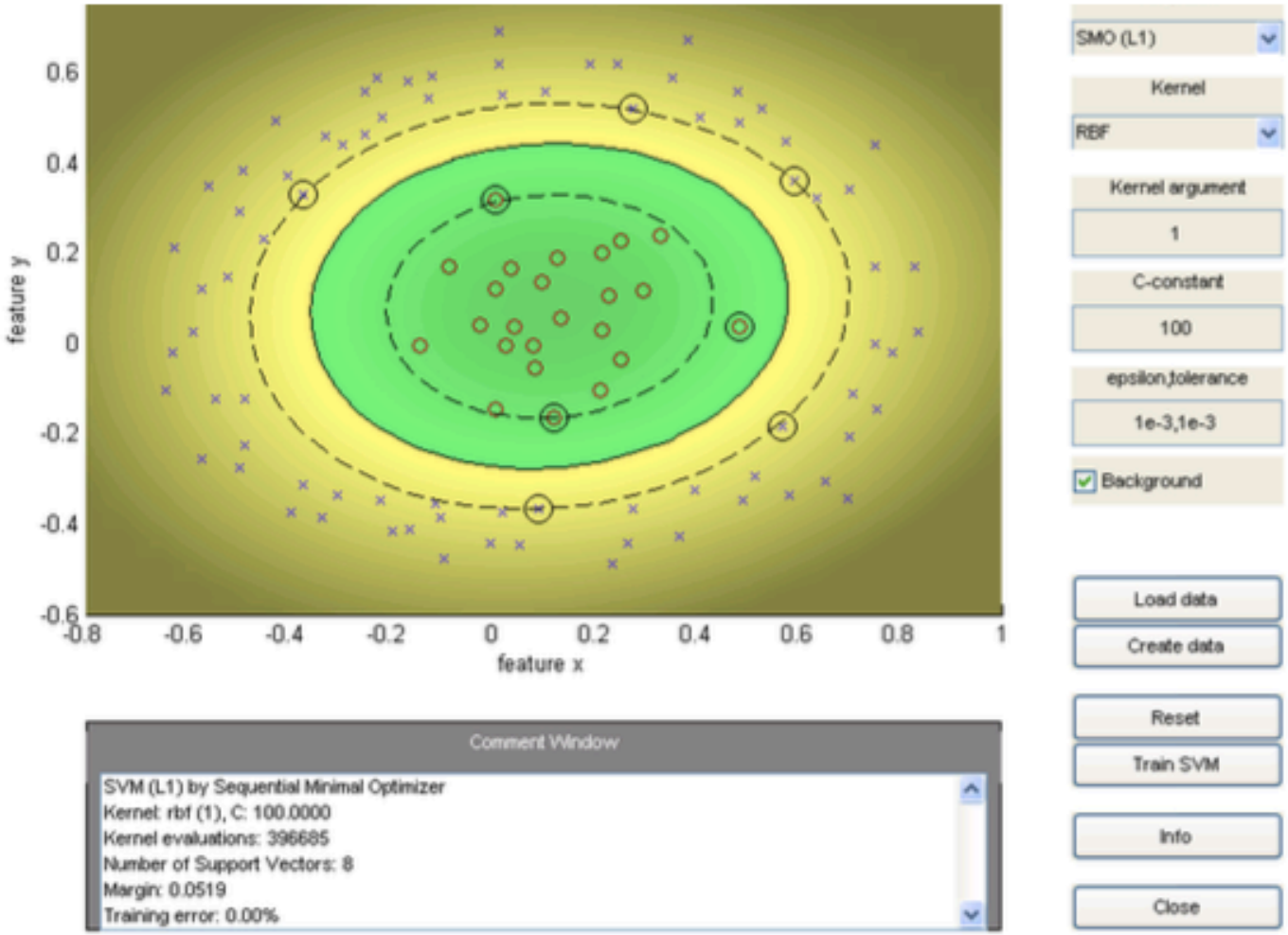
RBF-SVM example

$\sigma = 1.0 \quad C = \infty$



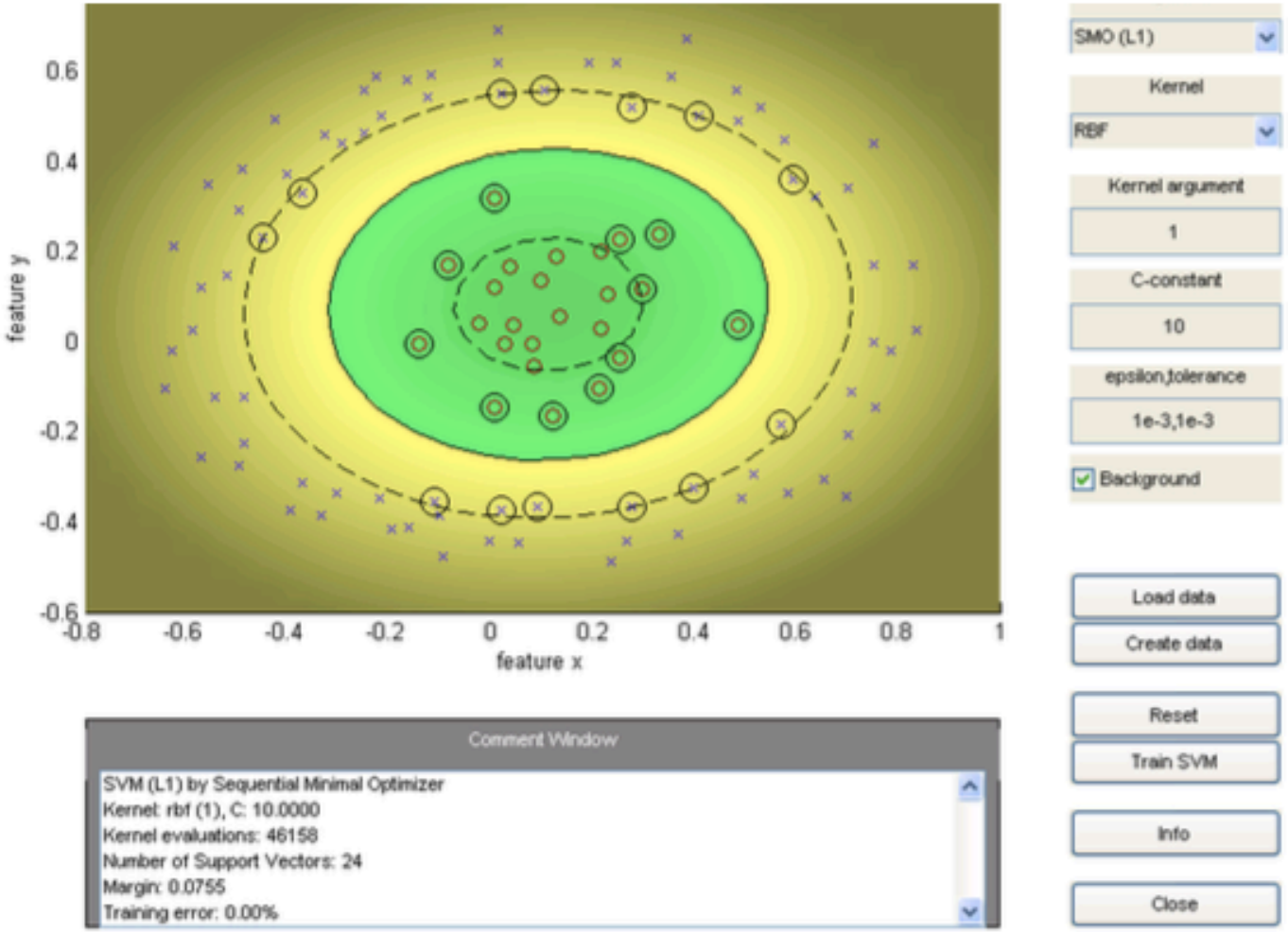
RBF-SVM example

$$\sigma = 1.0 \quad C = 100$$



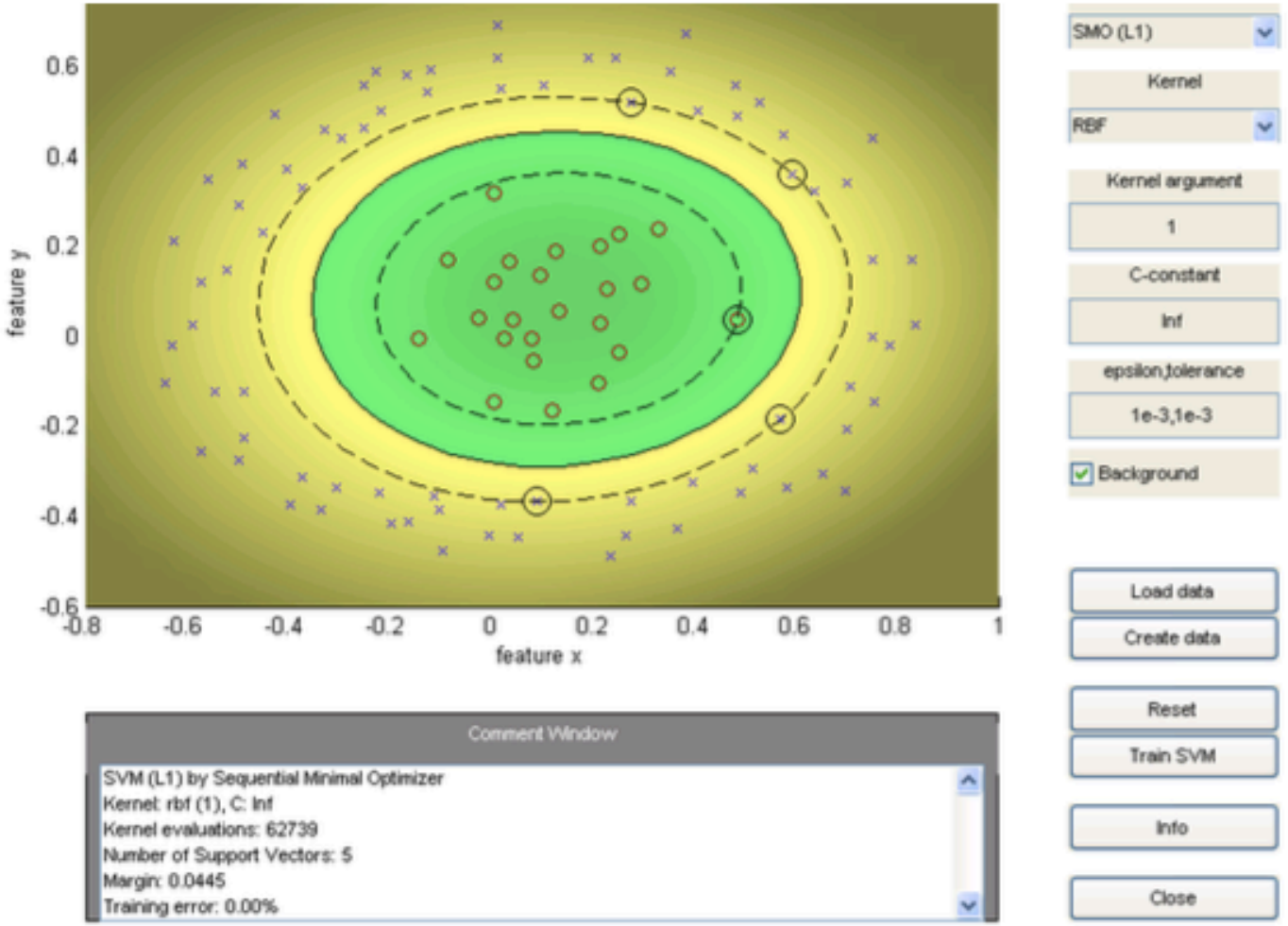
RBF-SVM example

$$\sigma = 1.0 \quad C = 10$$



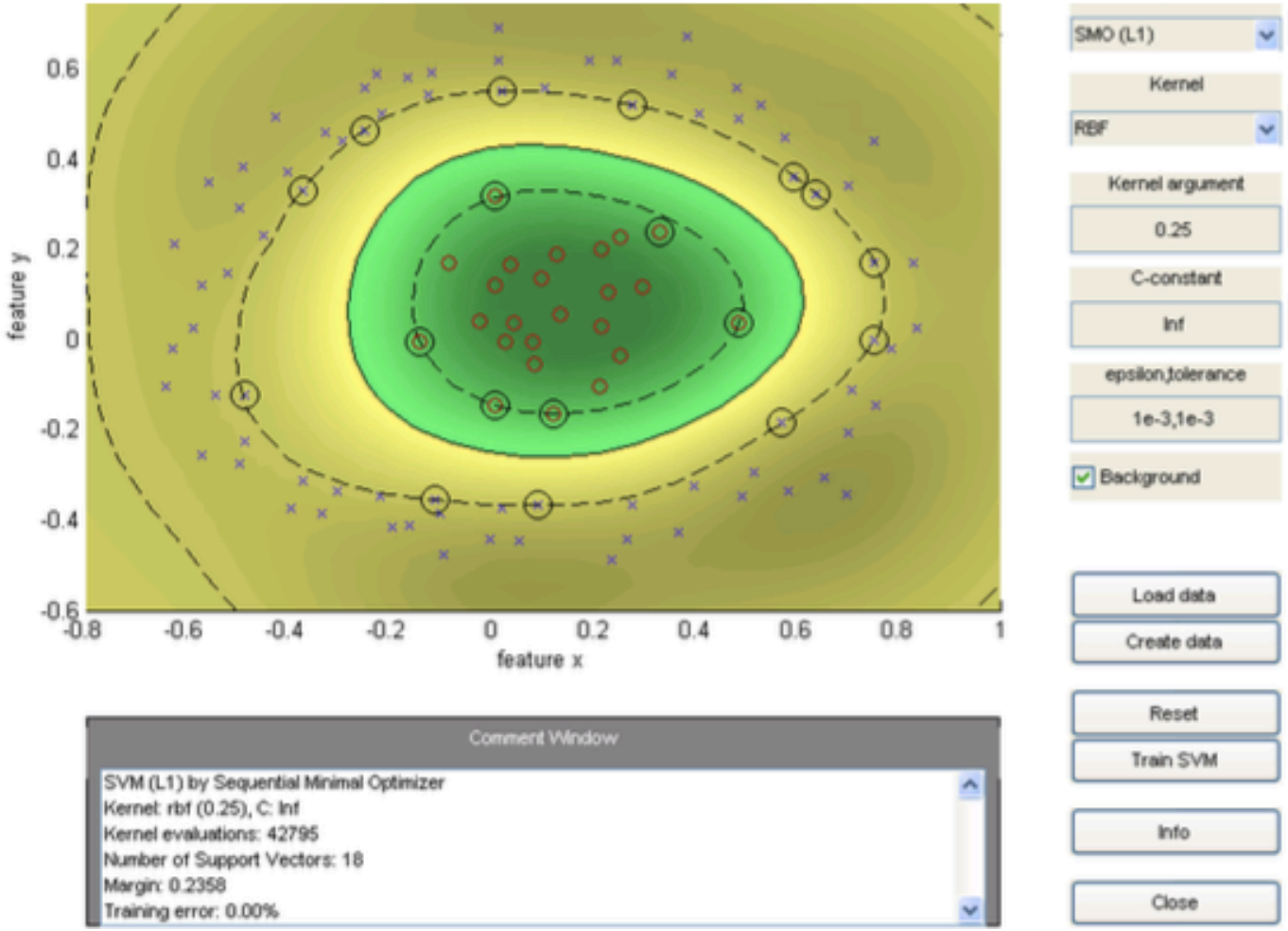
RBF-SVM example

$$\sigma = 1.0 \quad C = \infty$$



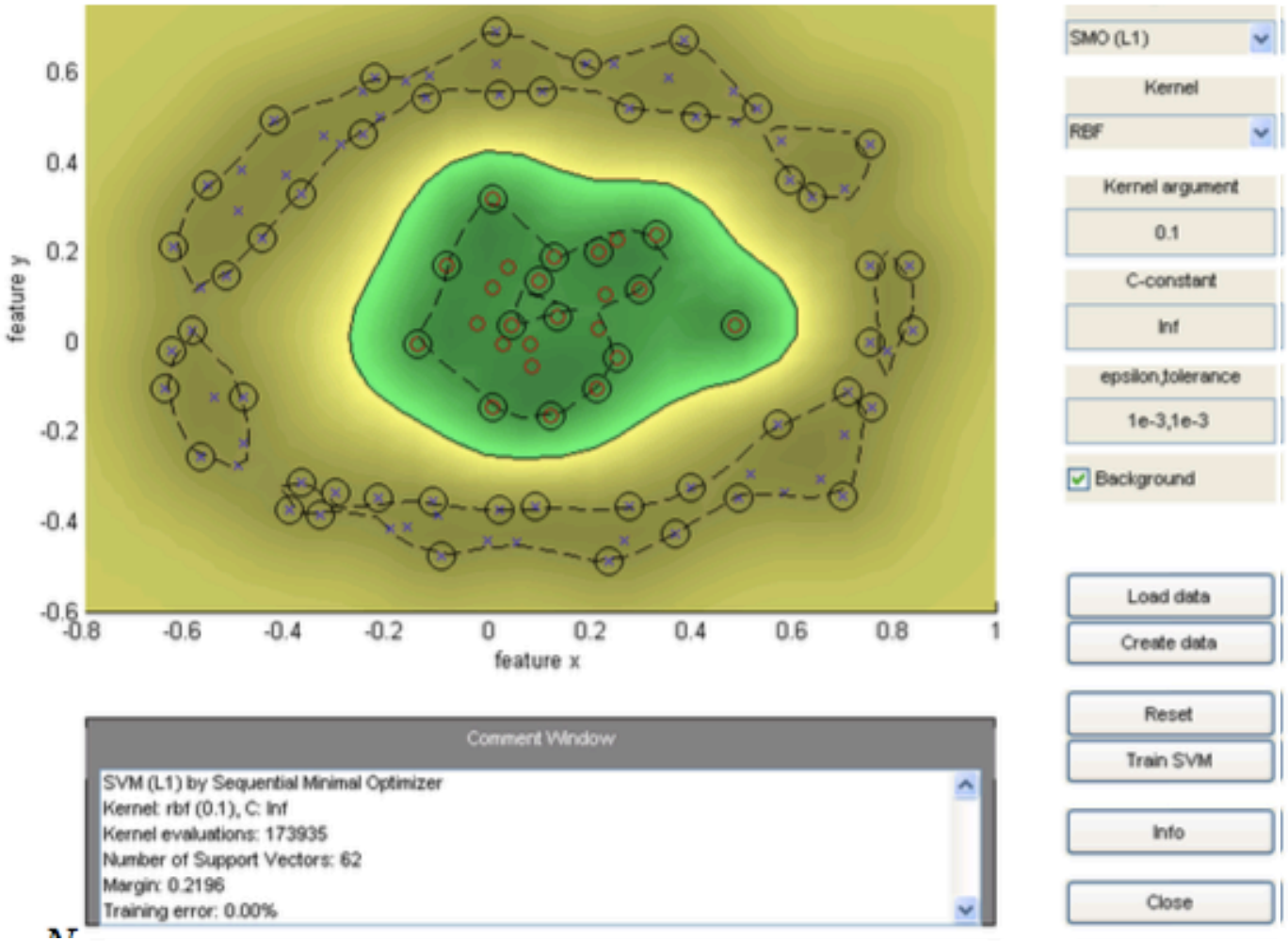
RBF-SVM example

$$\sigma = 0.25 \quad C = \infty$$

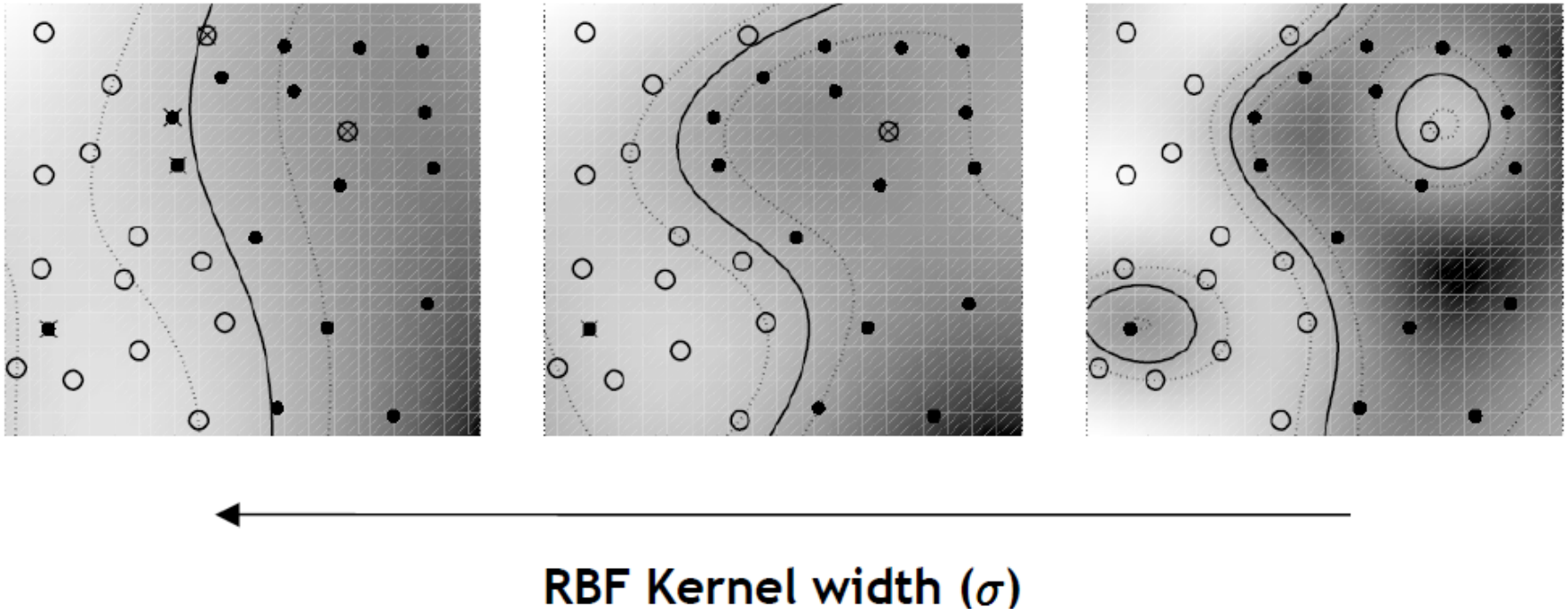


RBF-SVM example

$$\sigma = 0.1 \quad C = \infty$$



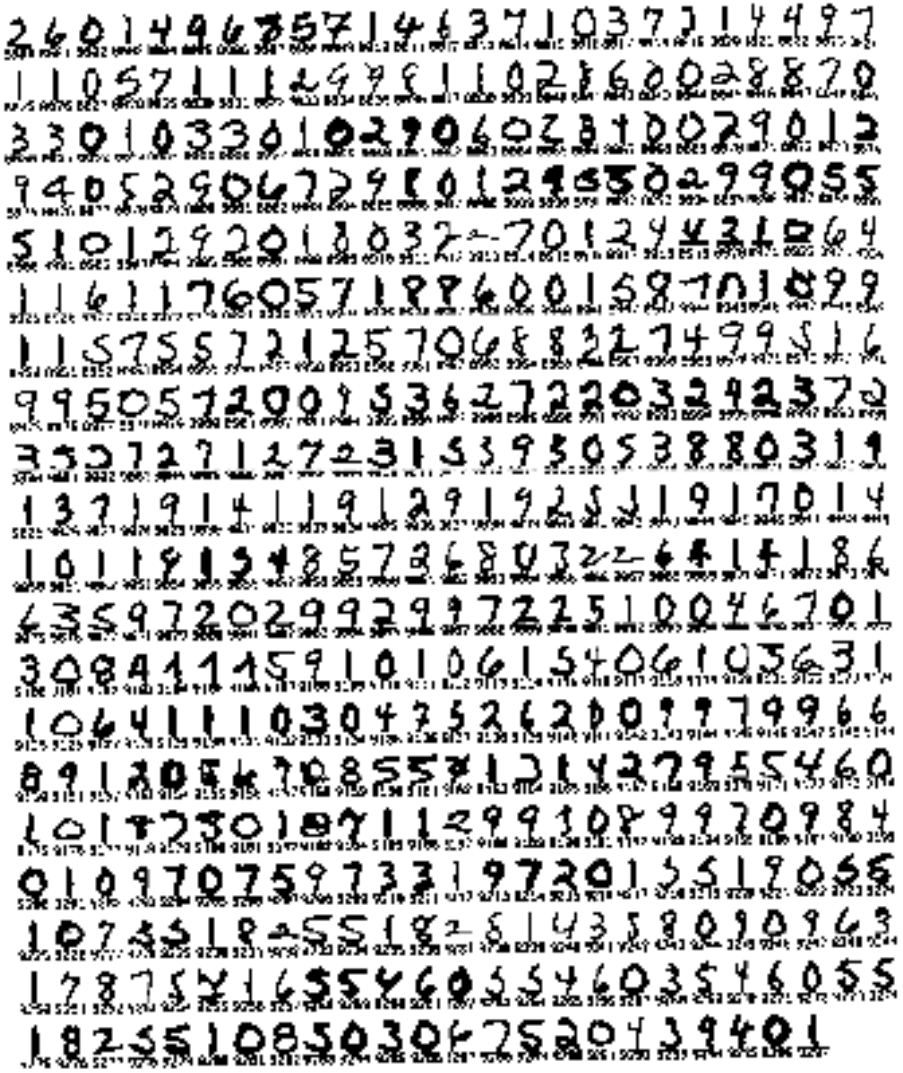
Large margins for nonlinear classifiers



Margin size: determined by both σ and regularizer

Guyon & Vapnik, 1995

- Handwritten digit recognition
  - US Postal Service Database
  - Standard benchmark task for many learning algorithms



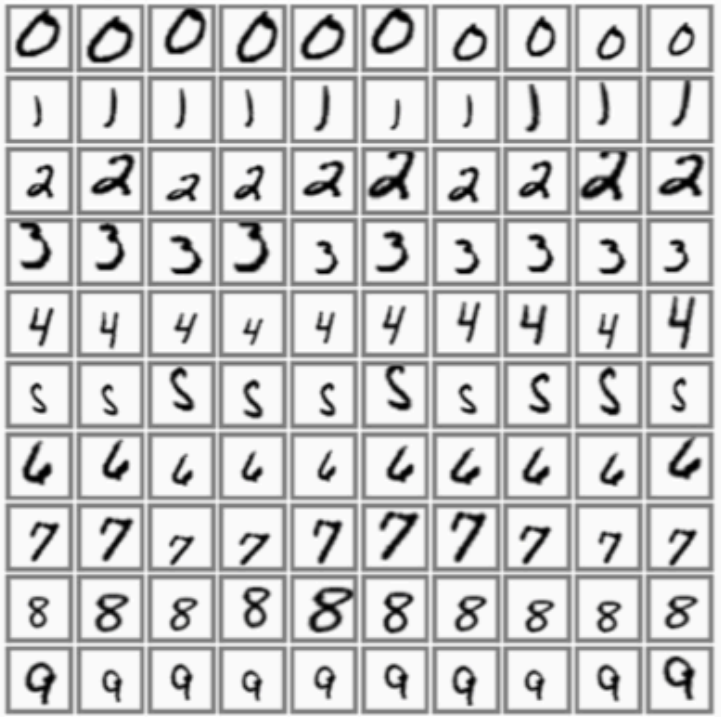
Application: Handwritten digit recognition

- Feature vectors: each image is 28 x 28 pixels. Rearrange as a 784-vector **x**

- Training: learn k=10 two-class 1 vs the rest SVM classifiers f_k (**x**)

- Classification: choose class with most positive score

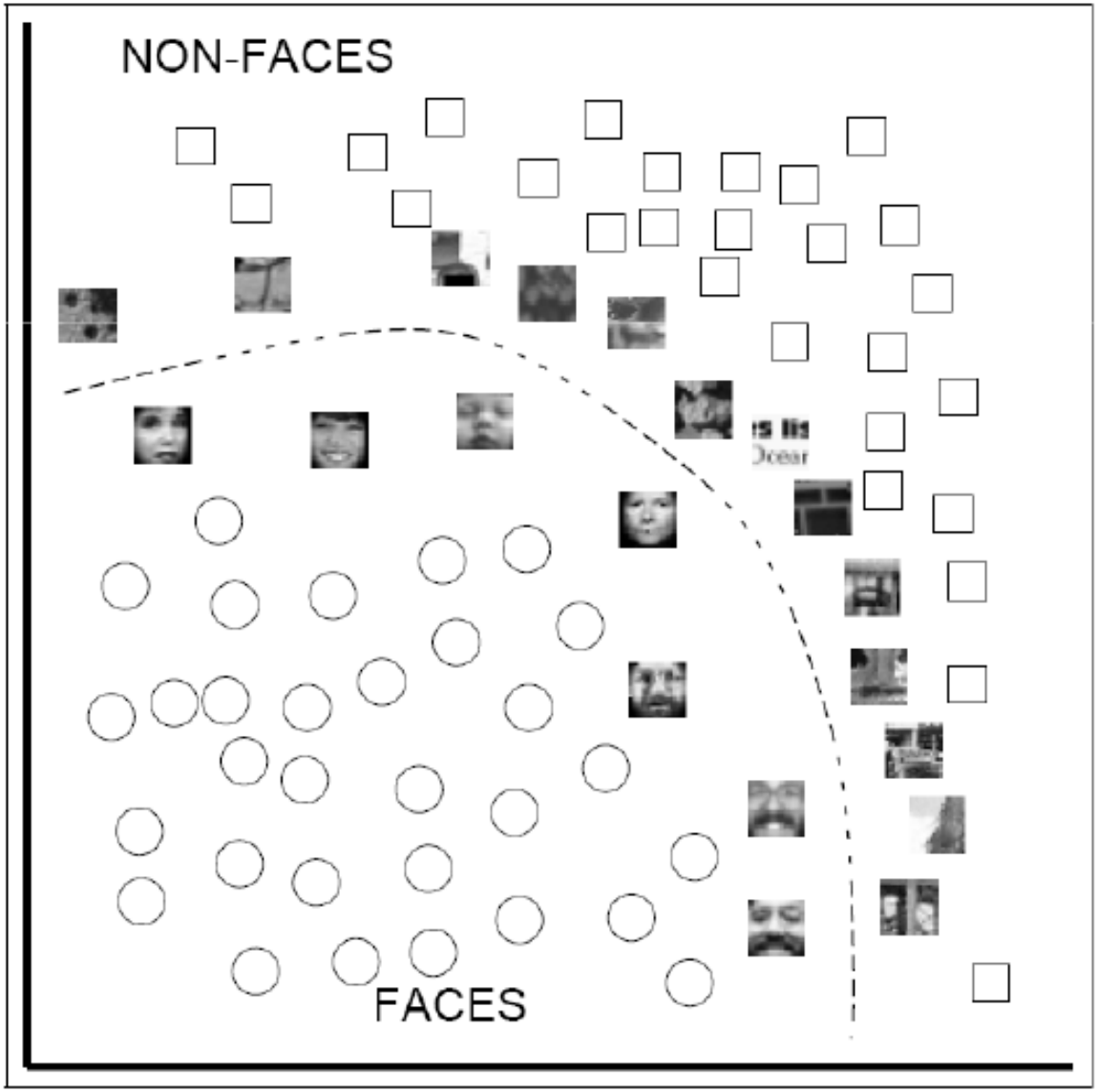
$$f(\mathbf{x}) = \max_k f_k(\mathbf{x})$$



Guyon & Vapnik 1995

- USPS benchmark
  - 2.5% error: human performance
- Different learning algorithms
  - 16.2% error: Decision tree (C4.5)
  - 5.9% error: (best) 2-layer Neural Network
  - 5.1% error: LeNet 1 - (massively hand-tuned) 5-layer network
- Different SVMs
  - 4.0% error: Polynomial kernel (p=3, 274 support vectors)
  - 4.1% error: Gaussian kernel (σ=0.3, 291 support vectors)

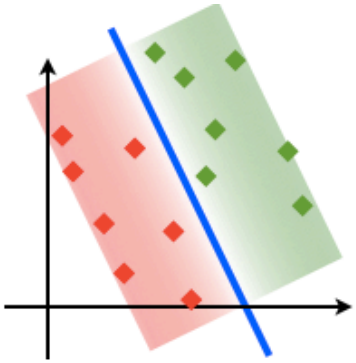
Support vectors for Faces (P&P 98)



Linear vs. Nonlinear

Linear SVM

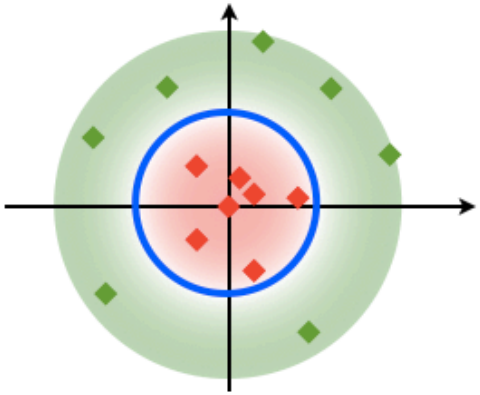
✔ fast
✘ restrictive



$$F(\mathbf{x}) = \langle \mathbf{w}, \mathbf{x} \rangle$$

Non-linear SVM

✘ much slower
✔ powerful



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

Slide credit: A. Vedaldi

Text Classification: Examples

- Classify news stories as *World, US, Business, SciTech, Sports, Entertainment, Health, Other*
- Add MeSH terms to Medline abstracts
  - e.g. “Conscious Sedation” [E03.250]
- Classify business names by industry.
- Classify student essays as *A,B,C,D,* or *F.*
- Classify email as *Spam, Other.*
- Classify email to tech staff as *Mac, Windows, ..., Other.*
- Classify pdf files as *ResearchPaper, Other*
- Classify documents as *WrittenByReagan, GhostWritten*
- Classify movie reviews as *Favorable,Unfavorable,Neutral.*
- Classify technical papers as *Interesting, Uninteresting.*
- Classify jokes as *Funny, NotFunny.*
- Classify web sites of companies by Standard Industrial Classification (SIC) code.

Text Classification: Examples

- Best-studied benchmark: *Reuters-21578* newswire stories
  - 9603 train, 3299 test documents, 80-100 words each, 93 classes

ARGENTINE 1986/87 GRAIN/OILSEED REGISTRATIONS

BUENOS AIRES, Feb 26

Argentine grain board figures show crop registrations of grains, oilseeds and their products to February 11, in thousands of tonnes, showing those for future shipments month, 1986/87 total and 1985/86 total to February 12, 1986, in brackets:

- Bread wheat prev 1,655.8, Feb 872.0, March 164.6, total 2,692.4 (4,161.0).
- Maize Mar 48.0, total 48.0 (nil).
- Sorghum nil (nil)
- Oilseed export registrations were:
- Sunflowerseed total 15.0 (7.9)
- Soybean May 20.0, total 20.0 (nil)

The board also detailed export registrations for subproducts, as follows....

→ Categories: **grain**, **wheat** (of 93 binary choices)

Representing text for classification

f(ARGENTINE 1986/87 GRAIN/OILSEED REGISTRATIONS
BUENOS AIRES, Feb 26
Argentine grain board figures show crop registrations of grains, oilseeds and their products to February 11, in thousands of tonnes, showing those for future shipments month, 1986/87 total and 1985/86 total to February 12, 1986, in brackets:

- Bread wheat prev 1,655.8, Feb 872.0, March 164.6, total 2,692.4 (4,161.0).
- Maize Mar 48.0, total 48.0 (nil).
- Sorghum nil (nil)
- Oilseed export registrations were:
- Sunflowerseed total 15.0 (7.9)
- Soybean May 20.0, total 20.0 (nil)

The board also detailed export registrations for subproducts, as follows....)=y

?

simplest useful

What is the ~~**best**~~ representation for the document *x* being classified?

Bag of words representation

ARGENTINE 1986/87 **GRAIN/OILSEED** REGISTRATIONS
BUENOS AIRES, Feb 26

Argentine **grain** board figures show crop registrations of **grains**, **oilseeds** and their products to February 11, in thousands of **tonnes**, showing those for future **shipments** month, 1986/87 **total** and 1985/86 **total** to February 12, 1986, in brackets:

- Bread **wheat** prev 1,655.8, Feb 872.0, March 164.6, **total** 2,692.4 (4,161.0).
- **Maize** Mar 48.0, total 48.0 (nil).
- **Sorghum** nil (nil)
- **Oilseed** export registrations were:
- **Sunflowerseed** total 15.0 (7.9)
- **Soybean** May 20.0, total 20.0 (nil)

The board also detailed export registrations for subproducts, as follows....

Categories: **grain**, **wheat**

Bag of words representation

xxxxxxxxxxxxxxxxxxxx **GRAIN/OILSEED** xxxxxxxxxxxxx
xxxxxxxxxxxxxxxxxxxxxx
xxxxxxxxx **grain** xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx **grains**, **oilseeds** xxxxxxxxxx xxxxxxxxxxxxxxxxxxxxxxxxxx **tonnes**, xxxxxxxxxxxxxxxxx **shipments** xxxxxxxxxxxx **total** xxxxxxxxx **total** xxxxxxxx xxxxxxxxxxxxxxxxxxxx:

- Xxxxx **wheat** xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx, **total** xxxxxxxxxxxxxxxx
- **Maize** xxxxxxxxxxxxxxxxx
- **Sorghum** xxxxxxxxxx
- **Oilseed** xxxxxxxxxxxxxxxxxxxxx
- **Sunflowerseed** xxxxxxxxxxxxxx
- **Soybean** xxxxxxxxxxxxxxxxxxxxxx

xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx....

Categories: **grain**, **wheat**

Bag of words representation

xxxxxxxxxxxxxxxxxx **GRAIN/OILSEED** xxxxxxxxxxxxx
xxxxxxxxxxxxxxxxxxxxxx

xxxxxxxxx **grain** xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx **grains**, **oilseeds** xxxxxxxxxx xxxxxxxxxxxxxxxxxxxxxxxxxx **tonnes**, xxxxxxxxxxxxxxxxx **shipments** xxxxxxxxxxxx **total** xxxxxxxxx **total** xxxxxxxx xxxxxxxxxxxxxxxxxxxx:

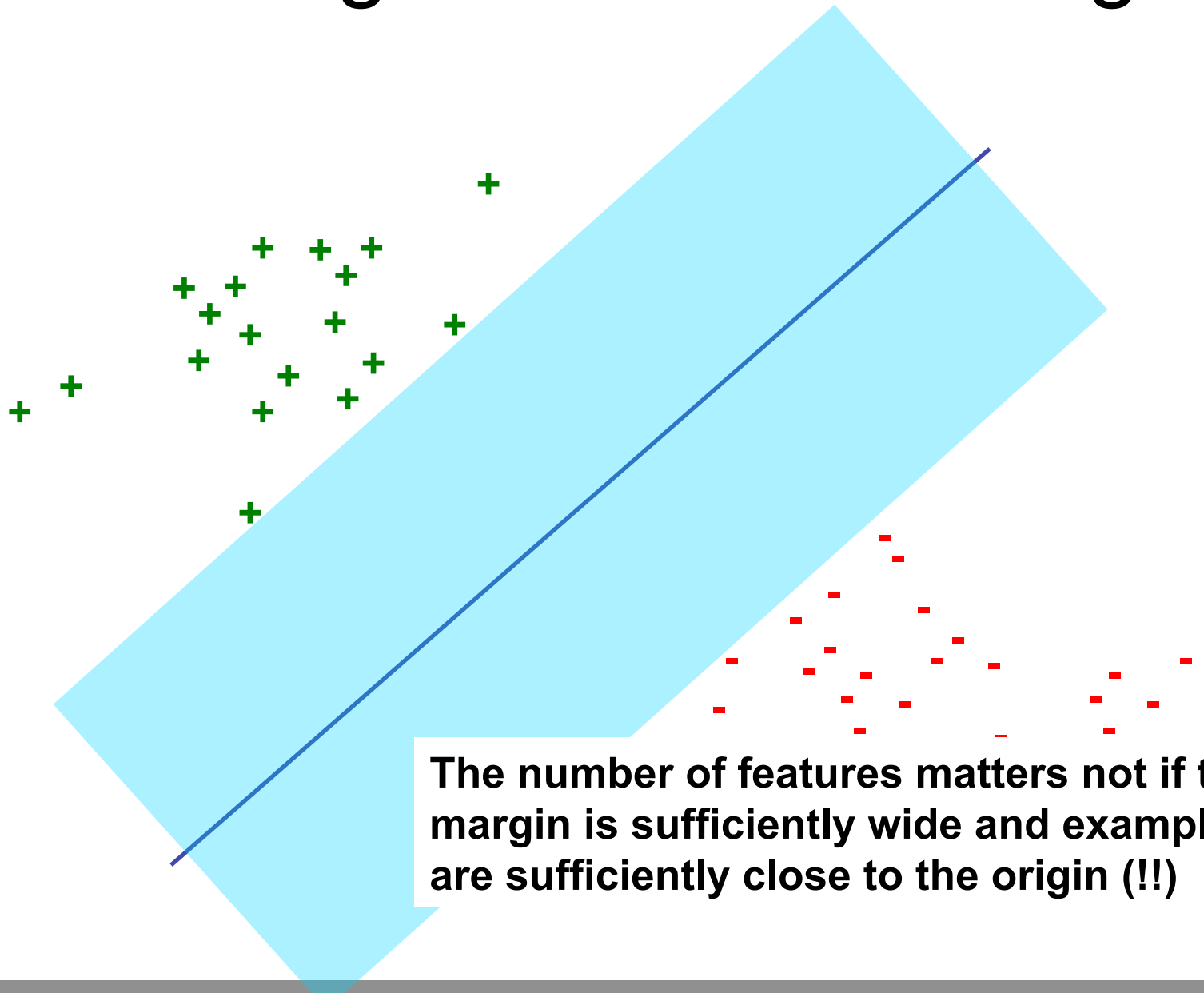
- Xxxxx **wheat** xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx, **total** xxxxxxxxxxxxxxxxx
- **Maize** xxxxxxxxxxxxxxxxx
- **Sorghum** xxxxxxxxxx
- **Oilseed** xxxxxxxxxxxxxxxxxxxxx
- **Sunflowerseed** xxxxxxxxxxxxxx
- **Soybean** xxxxxxxxxxxxxxxxxxxxx

xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx....

word	*freq*
grain(s)	3
oilseed(s)	2
total	3
wheat	1
maize	1
soybean	1
tonnes	1
...	...

Categories: **grain**, **wheat**

Margin-based Learning



The number of features matters not if the margin is sufficiently wide and examples are sufficiently close to the origin (!!)

Support Vector Machine Results

					SVM (poly) degree $d =$					SVM (rbf) width $\gamma =$			
	Bayes	Rocchio	C4.5	k-NN	1	2	3	4	5	0.6	0.8	1.0	1.2
earn	95.9	96.1	96.1	97.3	98.2	98.4	**98.5**	98.4	98.3	**98.5**	98.5	98.4	98.3
acq	91.5	92.1	85.3	92.0	92.6	94.6	**95.2**	95.2	95.3	95.0	95.3	95.3	**95.4**
money-fx	62.9	67.6	69.4	78.2	66.9	72.5	75.4	74.9	**76.2**	74.0	75.4	**76.3**	75.9
grain	72.5	79.5	89.1	82.2	91.3	93.1	**92.4**	91.3	89.9	**93.1**	91.9	91.9	90.6
crude	81.0	81.5	75.5	85.7	86.0	87.3	88.6	**88.9**	87.8	**88.9**	89.0	88.9	88.2
trade	50.0	77.4	59.2	77.4	69.2	75.5	76.6	77.3	**77.1**	76.9	78.0	**77.8**	76.8
interest	58.0	72.5	49.1	74.0	69.8	63.3	67.9	73.1	**76.2**	74.4	75.0	**76.2**	76.1
ship	78.7	83.1	80.9	79.2	82.0	85.4	86.0	**86.5**	86.0	**85.4**	86.5	87.6	87.1
wheat	60.6	79.4	85.5	76.6	83.1	84.5	85.2	**85.9**	83.8	**85.2**	85.9	85.9	85.9
corn	47.3	62.2	87.7	77.9	86.0	86.5	85.3	**85.7**	83.9	**85.1**	85.7	85.7	84.5
microavg.	**72.0**	**79.9**	**79.4**	**82.3**	84.2	85.1	85.9	86.2	85.9	86.4	86.5	86.3	86.2
					combined: **86.0**					combined: **86.4**			

Learning Sequence-based Protein Classification

- Problem: classification of protein sequence data into families and superfamilies
- Motivation: Many proteins have been sequenced, but often structure/function remains unknown
- Motivation: infer *structure/function* from *sequence-based* classification

Sequence Data versus Structure and Function

Sequences for four chains of human hemoglobin

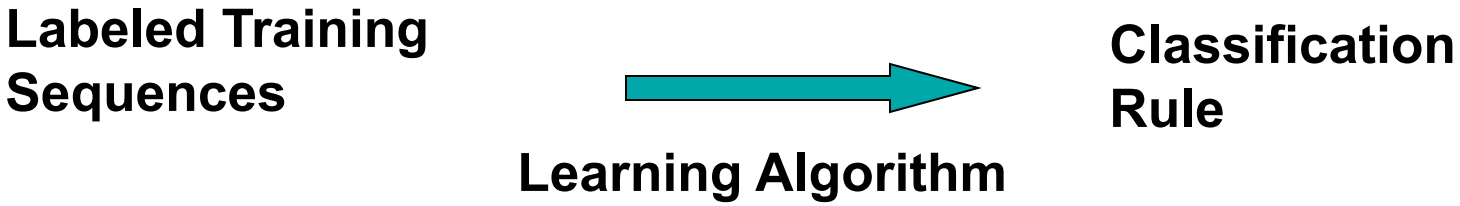
```
>1A3N:A HEMOGLOBIN
VLSPADKTNVKAAWGKVGAHAGEYGAEALERMFLSFPTTKTYFPHFDLSHGSAQVKGHGK
KVADALTNAVAHVDDMPNALSALSDLHAHKLRVDPVNFKLLSHCLLVTLAAHLPAEFTPA
VHASLDKFLASVSTVLTSKYR
>1A3N:B HEMOGLOBIN
VHLTPEEKSAVTALWGKVNVDEVGGEALGRLLVVYPWTQRFFESFGDLSTPDAVMGNPKV
KAHGKKVLGAFSDGLAHLDNLKGTFATLSELHCDKLHVDPENFRLLGNVLVCVLAHHFGK
EFTPPVQAAYQKVVAGVANALAHKYH
>1A3N:C HEMOGLOBIN
VLSPADKTNVKAAWGKVGAHAGEYGAEALERMFLSFPTTKTYFPHFDLSHGSAQVKGHGK
KVADALTNAVAHVDDMPNALSALSDLHAHKLRVDPVNFKLLSHCLLVTLAAHLPAEFTPA
VHASLDKFLASVSTVLTSKYR
>1A3N:D HEMOGLOBIN
VHLTPEEKSAVTALWGKVNVDEVGGEALGRLLVVYPWTQRFFESFGDLSTPDAVMGNPKV
KAHGKKVLGAFSDGLAHLDNLKGTFATLSELHCDKLHVDPENFRLLGNVLVCVLAHHFGK
EFTPPVQAAYQKVVAGVANALAHKYH
```

Tertiary Structure

Function: oxygen transport

Learning Problem

- Reduce to binary classification problem: positive (+) if example belongs to a family (e.g. G proteins) or superfamily (e.g. nucleoside triphosphate hydrolases), negative (-) otherwise
- Focus on *remote homology detection*
- Use *supervised learning* approach to *train* a classifier



SVMs for Protein Classification

- Want to define feature map from space of protein sequences to vector space
- Goals:
  - Computational efficiency
  - Competitive performance with known methods
  - No reliance on generative model – general method for sequence-based classification problems

Spectrum Feature Map for SVM Protein Classification

New feature map based on
spectrum of a sequence

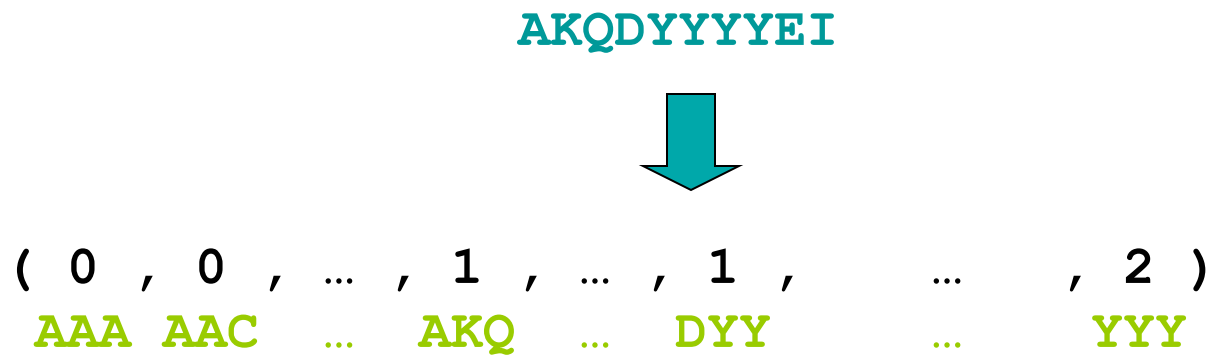
1. C. Leslie, E. Eskin, and W. Noble, *The Spectrum Kernel: A String Kernel for SVM Protein Classification.* Pacific Symposium on Biocomputing, 2002.
2. C. Leslie, E. Eskin, J. Weston and W. Noble, *Mismatch String Kernels for SVM Protein Classification.* NIPS 2002.

The k-Spectrum of a Sequence

- Feature map for SVM based on *spectrum* of a sequence
- The k-spectrum of a sequence is the set of all k-length contiguous subsequences that it contains
- Feature map is indexed by all possible k-length subsequences ("k-mers") from the alphabet of amino acids
- Dimension of feature space = 20^k
- Generalizes to any sequence data

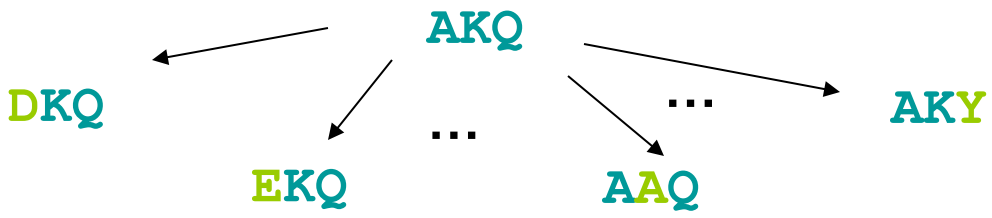
k-Spectrum Feature Map

- Feature map for k-spectrum with no mismatches:
  - For sequence x, $F_{(k)}(x) = (F_t(x))_{\{k\text{-mers } t\}}$, where $F_t(x)$ = #occurrences of t in x



(k,m)-Mismatch Feature Map

- Feature map for k-spectrum, allowing m mismatches:
  - if s is a k-mer, $F_{(k,m)}(s) = (F_t(s))_{\{\text{k-mers } t\}}$, where $F_t(s) = 1$ if s is within m mismatches from t, 0 otherwise
  - extend additively to longer sequences x by summing over all k-mers s in x



The Kernel Trick

- To train an SVM, can use *kernel* rather than explicit feature map
  - For sequences x, y, feature map F, kernel value is inner product in feature space:

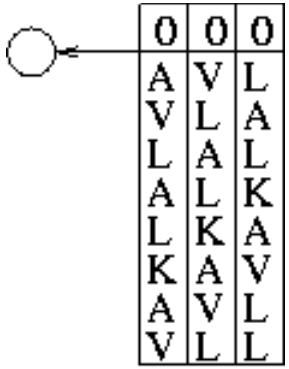
    $K(x, y) = \langle F(x), F(y) \rangle$
  - Gives sequence similarity score
  - Example of a string kernel
  - Can be efficiently computed via traversal of *trie data structure*

Computing the (k,m)-Spectrum Kernel

- Use *trie* (retrieval tree) to organize lexical traversal of all instances of k-length patterns (with mismatches) in the training data
  - Each path down to a leaf in the trie corresponds to a coordinate in feature map
  - Kernel values for all training sequences updated at each leaf node
  - If m=0, traversal time for trie is linear in size of training data
  - Traversal time grows exponentially with m, but usually small values of m are useful
  - Depth-first traversal makes efficient use of memory

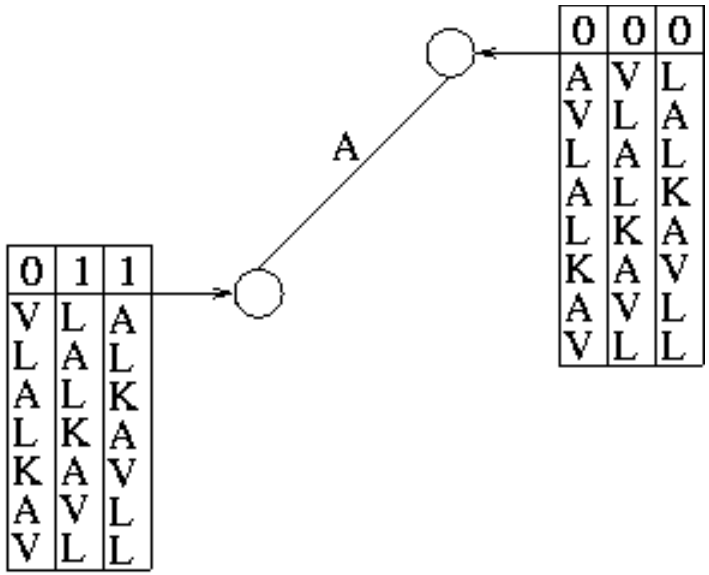
Example: Traversing the Mismatch Tree

– Traversal for input sequence: **AVLALKAVLL**, k=8, m=1



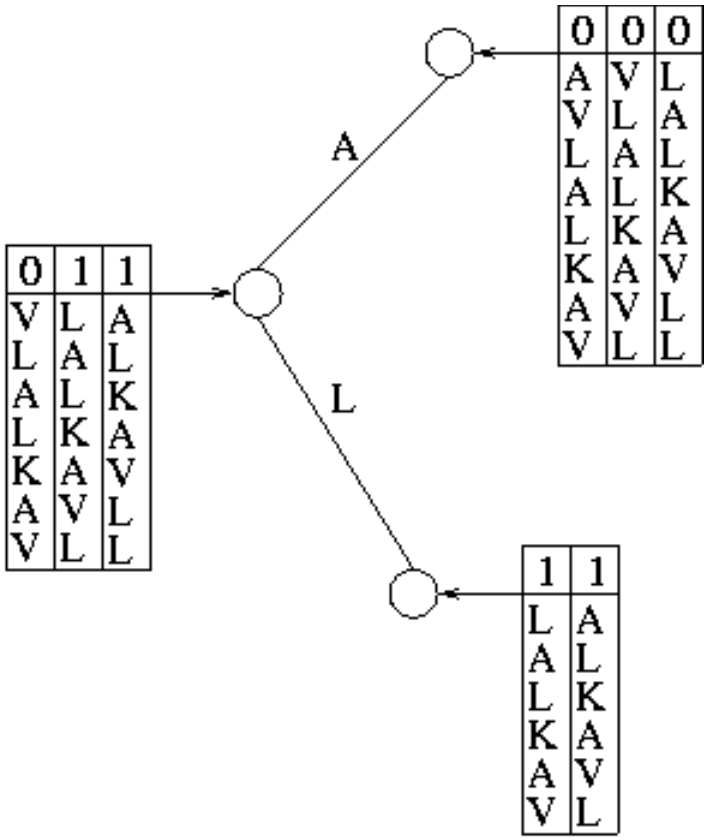
Example: Traversing the Mismatch Tree

– Traversal for input sequence: **AVLALKAVLL**, k=8, m=1



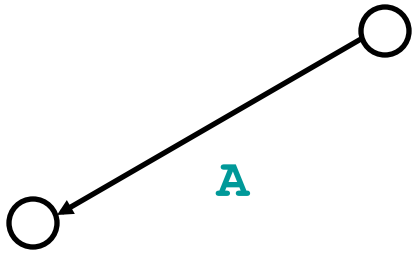
Example: Traversing the Mismatch Tree

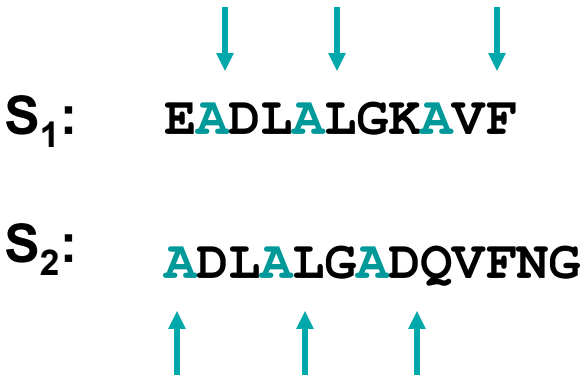
– Traversal for input sequence: **AVLALKAVLL**, k=8, m=1



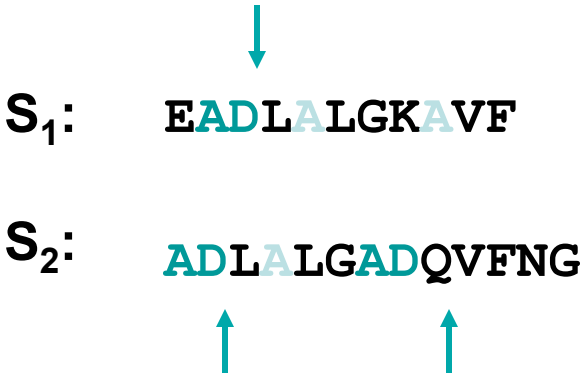
Example: Computing the Kernel for Pair of Sequences

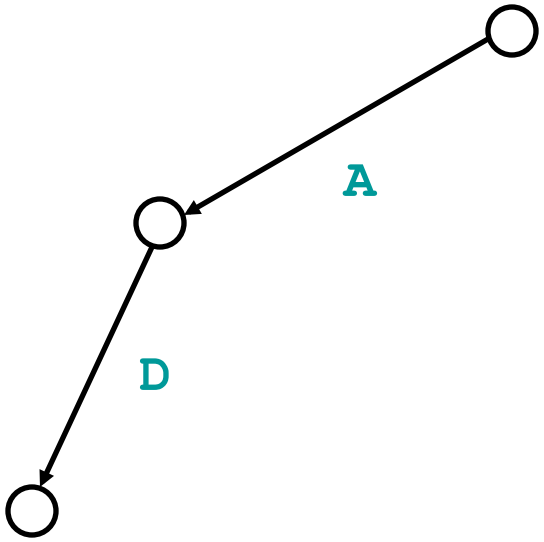
– Traversal of trie for k=3 (m=0)



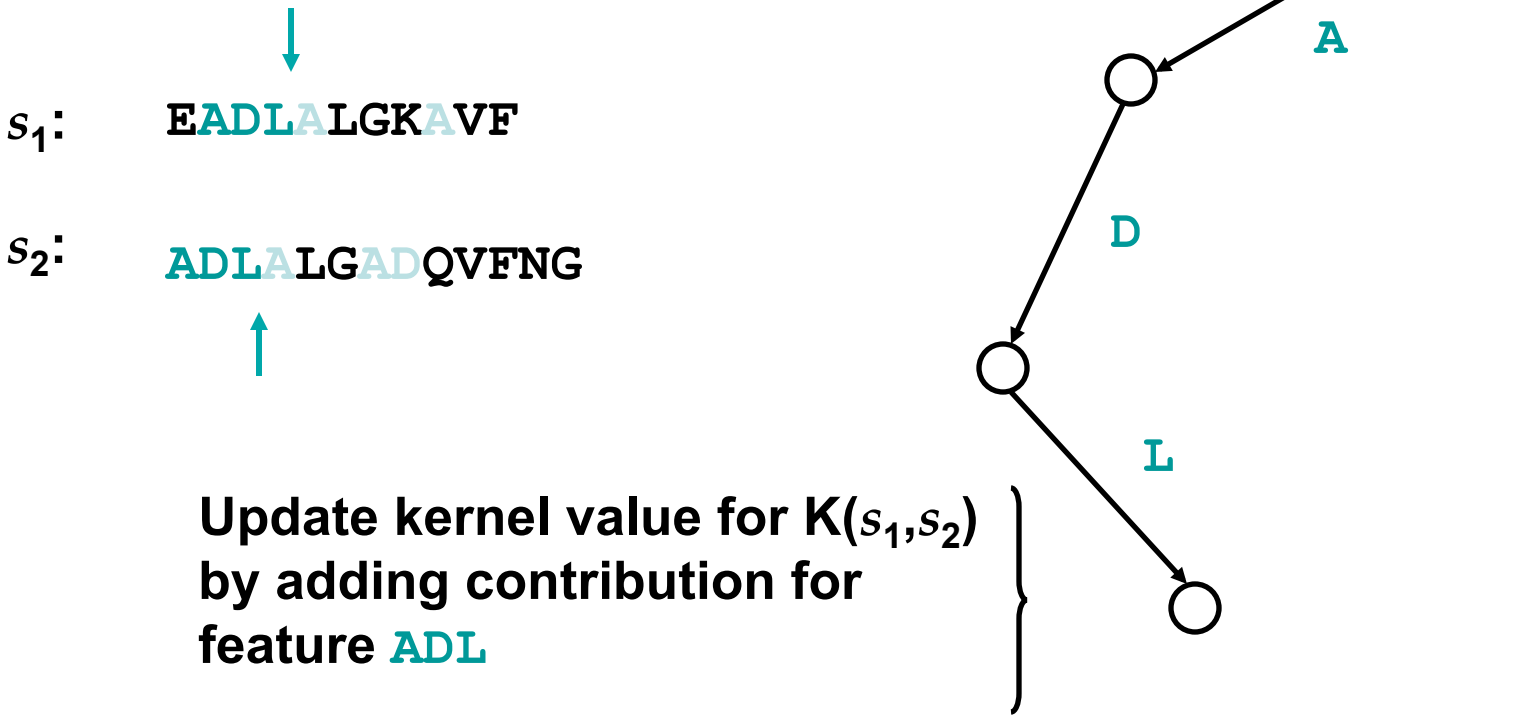


– Traversal of trie for k=3 (m=0)





– Traversal of trie for k=3 (m=0)



Update kernel value for K(s_1,s_2) by adding contribution for feature ADL

Fast prediction

- SVM training: determines subset of training sequences corresponding to *support vectors* and their weights:

  (x_i, α_i), i = 1 .. r

- Prediction with no mismatches:
  - Represent SVM classifier by hash table mapping support k-mers to weights
  - Test sequences can be classified in linear time via look-up of k-mers
- Prediction with mismatches:
  - Represent classifier as sparse trie; traverse k-mer paths occurring with mismatches in test sequence

Conclusions for SCOP Experiments

- Spectrum Kernel with SVM performs as well as the best-known method for remote homology detection problem
- Efficient computation of string kernel
- Fast prediction
  - Can precompute per k-mer scores and represent classifier as a lookup table
  - Gives linear time prdiction for both spectrum kernel, (unnormalized) mismatch kernel
- General approach to classification problems for sequence data

Bag-of-word models

Object → Bag of 'words'

Analogy to documents

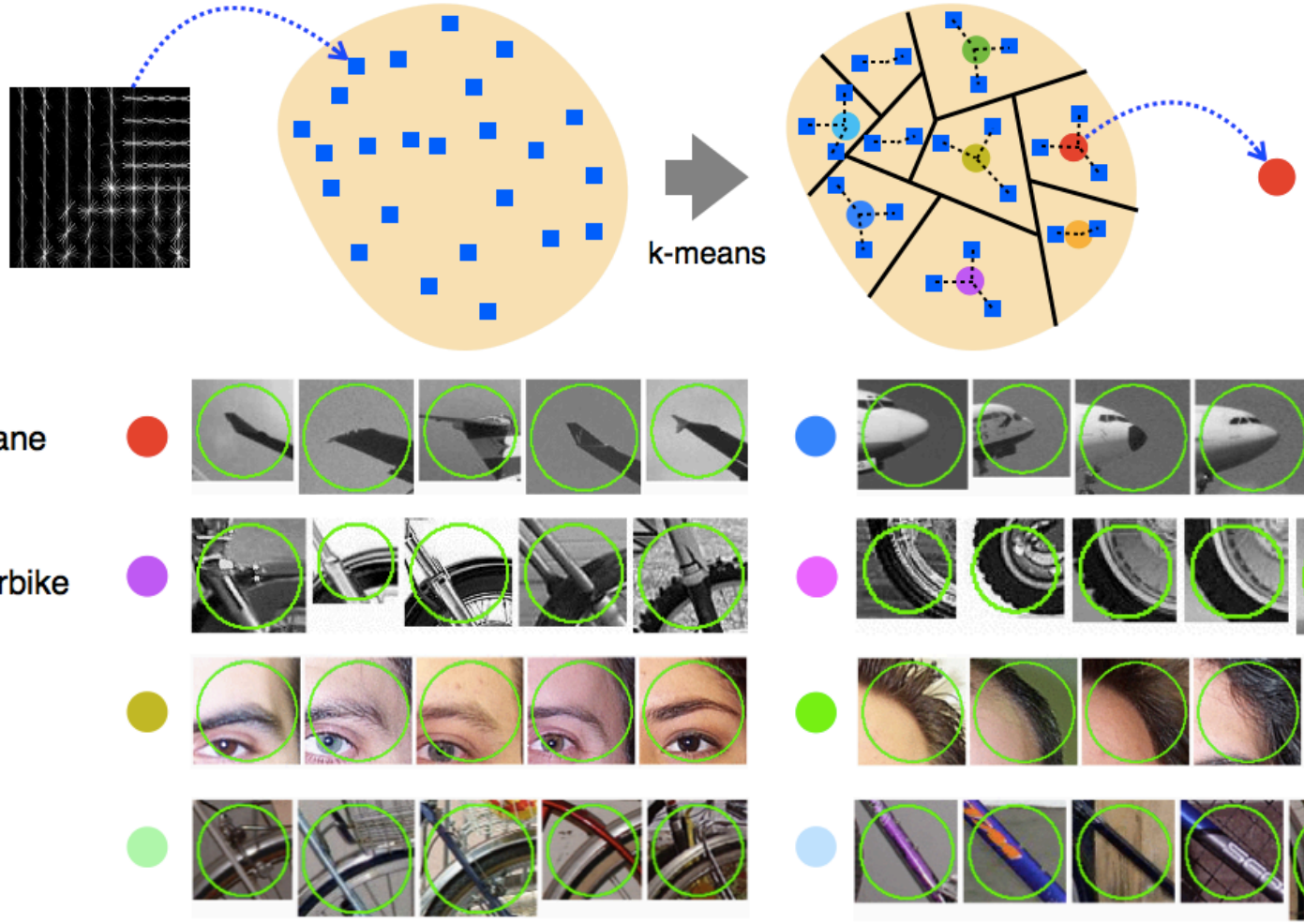
Of all the sensory impressions proceeding to the brain, the visual experiences are the dominant ones. Our perception of the world around us is based essentially on the messages that [illegible] brain from our eyes. For [illegible] ought that the re[illegible] point by [illegible] brain; t[illegible] screen[illegible] image [illegible] the dis[illegible] now kn[illegible] visual pe[illegible] considerab[illegible] of events. By fo[illegible] ses along their path [illegible] vers of the optical cortex, Hubel and [illegible] have been able to demonstrate th[illegible] *message about the image falling o[illegible] retina undergoes a step-wise analys[illegible] system of nerve cells stored in colum[illegible] In this system each cell has its specific function and is responsible for a specific detail in the pattern of the retinal image.*

sensory, brain, visual, perception, retinal, cerebral cortex, eye, cell, optical nerve, image Hubel, Wiesel

China is forecasting a trade surplus of $90bn (£51bn) to $100bn this year, a threefold increase on 2004's $32bn. The Commerce Ministry said the surplus would be created by a predicted 30% jump in exports to $[illegible] with a 18% rise in [illegible] figures are likely [illegible] ch has lon[illegible] e unfairl[illegible] under[illegible] surplu[illegible] only o[illegible] Zhou X[illegible] needed t[illegible] demand so[illegible] the country. China[illegible] f the yuan against the d[illegible] ly and permitted it to trade within a[illegible] w band, but the US wants the yuan t[illegible] allowed to trade freely. However, Be[illegible] has made it clear that it will take its t[illegible] and tread carefully before allowing the yuan to rise further in value.

China, trade, surplus, commerce, exports, imports, US, yuan, bank, domestic, foreign, increase, trade, value

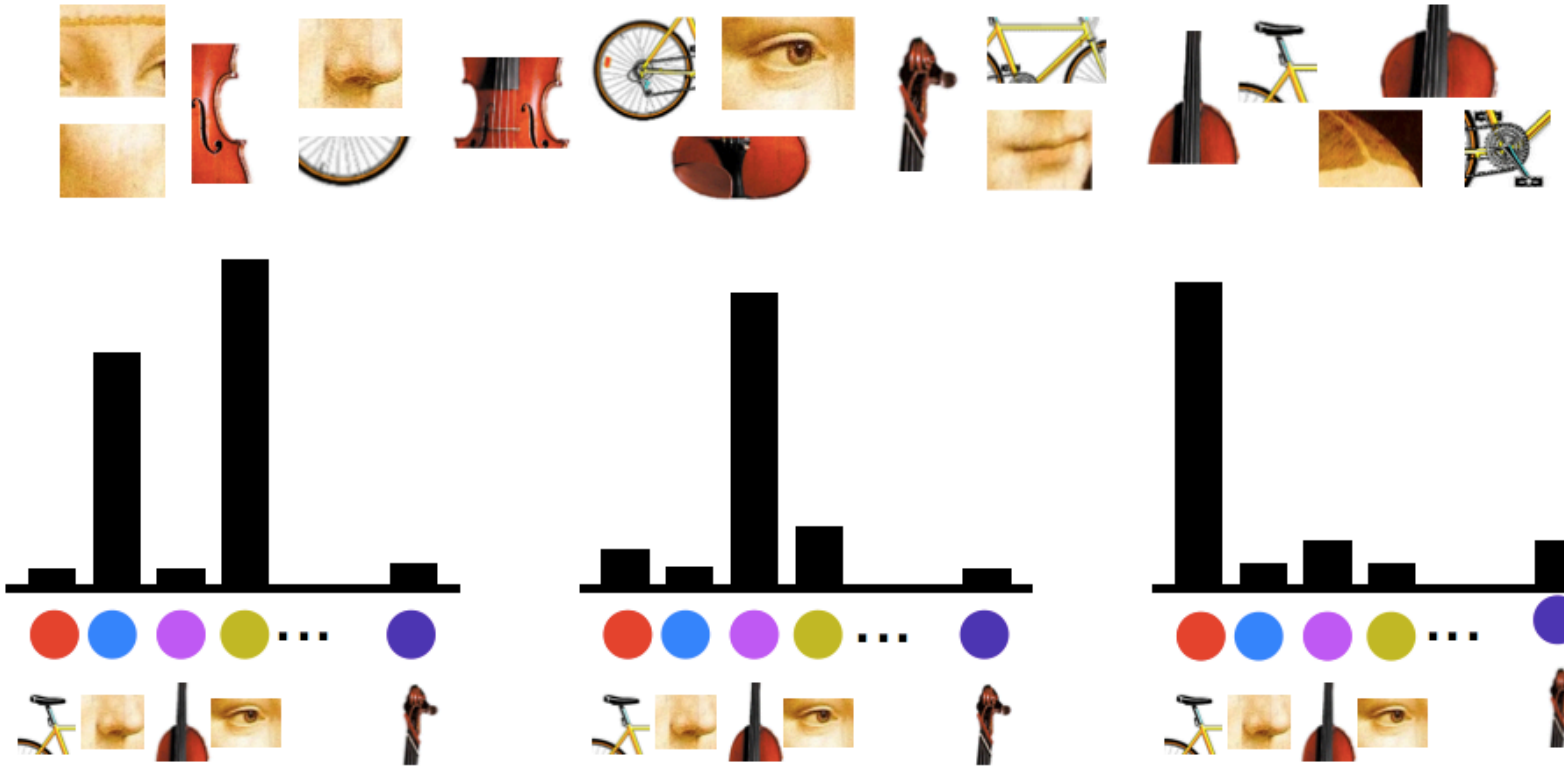
Feature quantization



Slide credit: A. Vedaldi

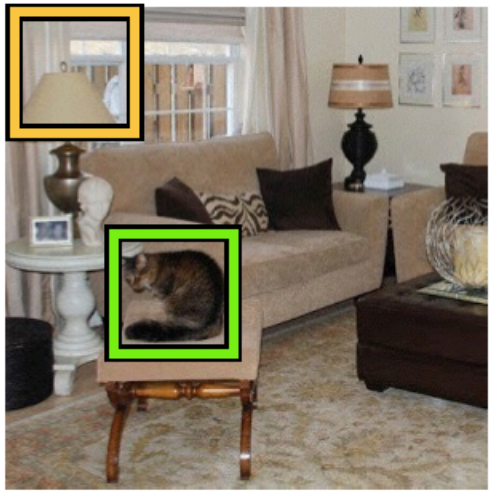
Visual words representation

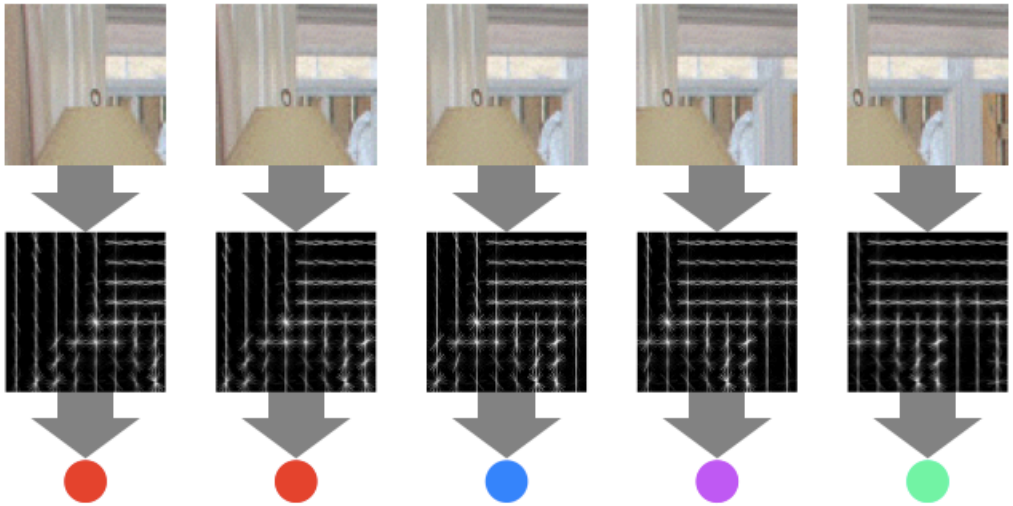
Visual words represent **“iconic” image fragments**



Bag-of-word features

[Sivic & Zisserman 2003, Csurka *et al.* 2004, Nowak *et al.* 2006]

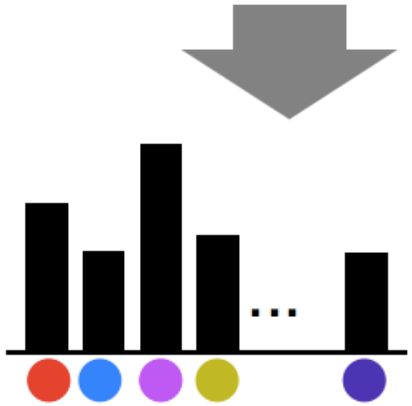




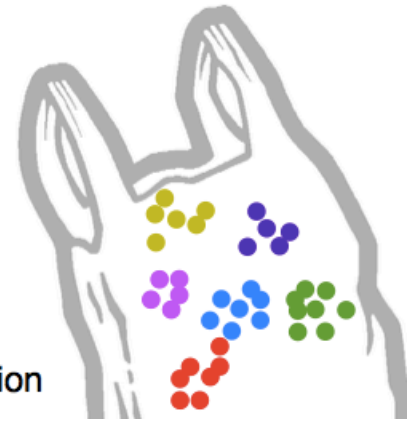
BoVW construction

1. Extract local descriptor densely
2. Quantise descriptors
3. Form histogram

2. **Discards spatial information**

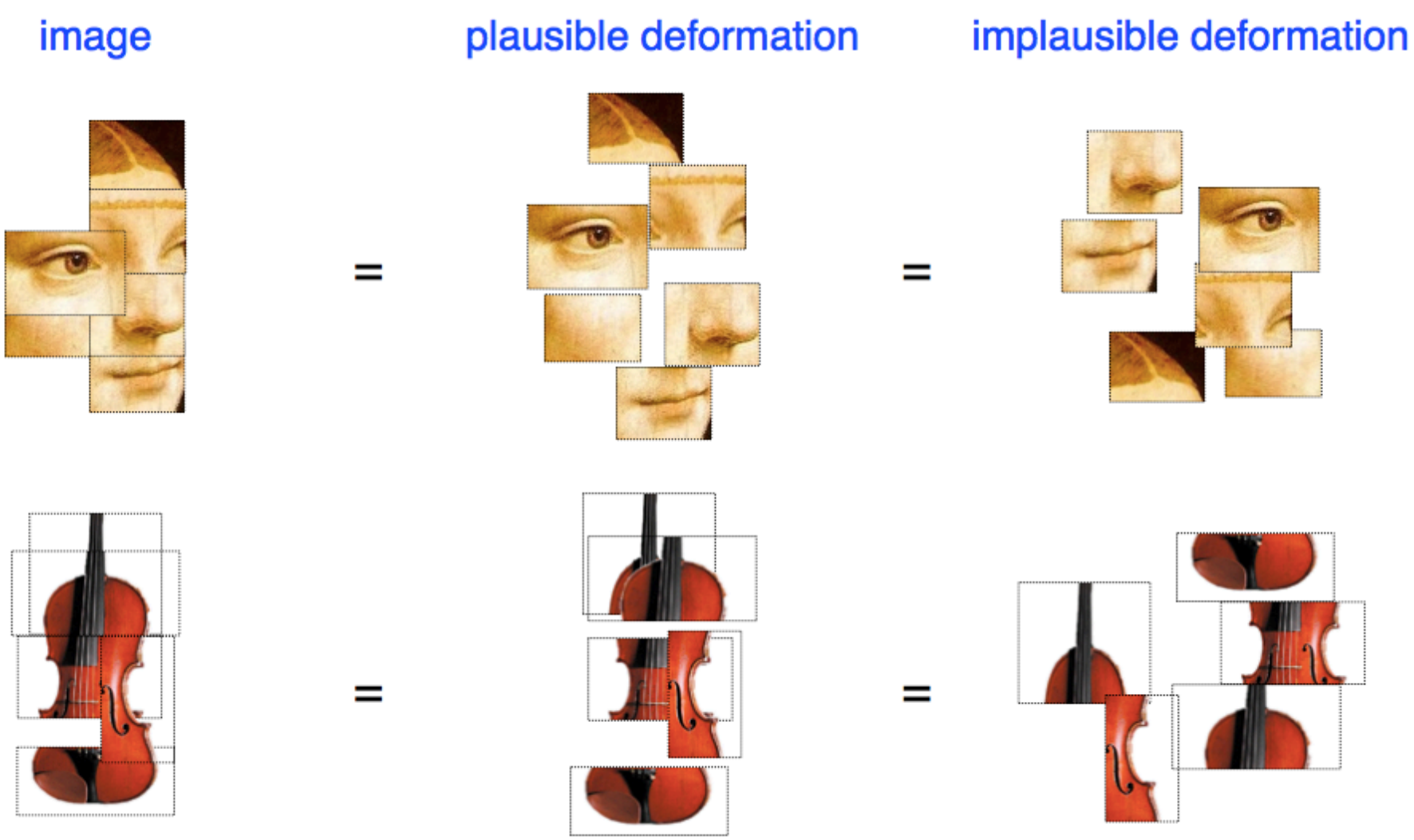


+ l^2 normalisation



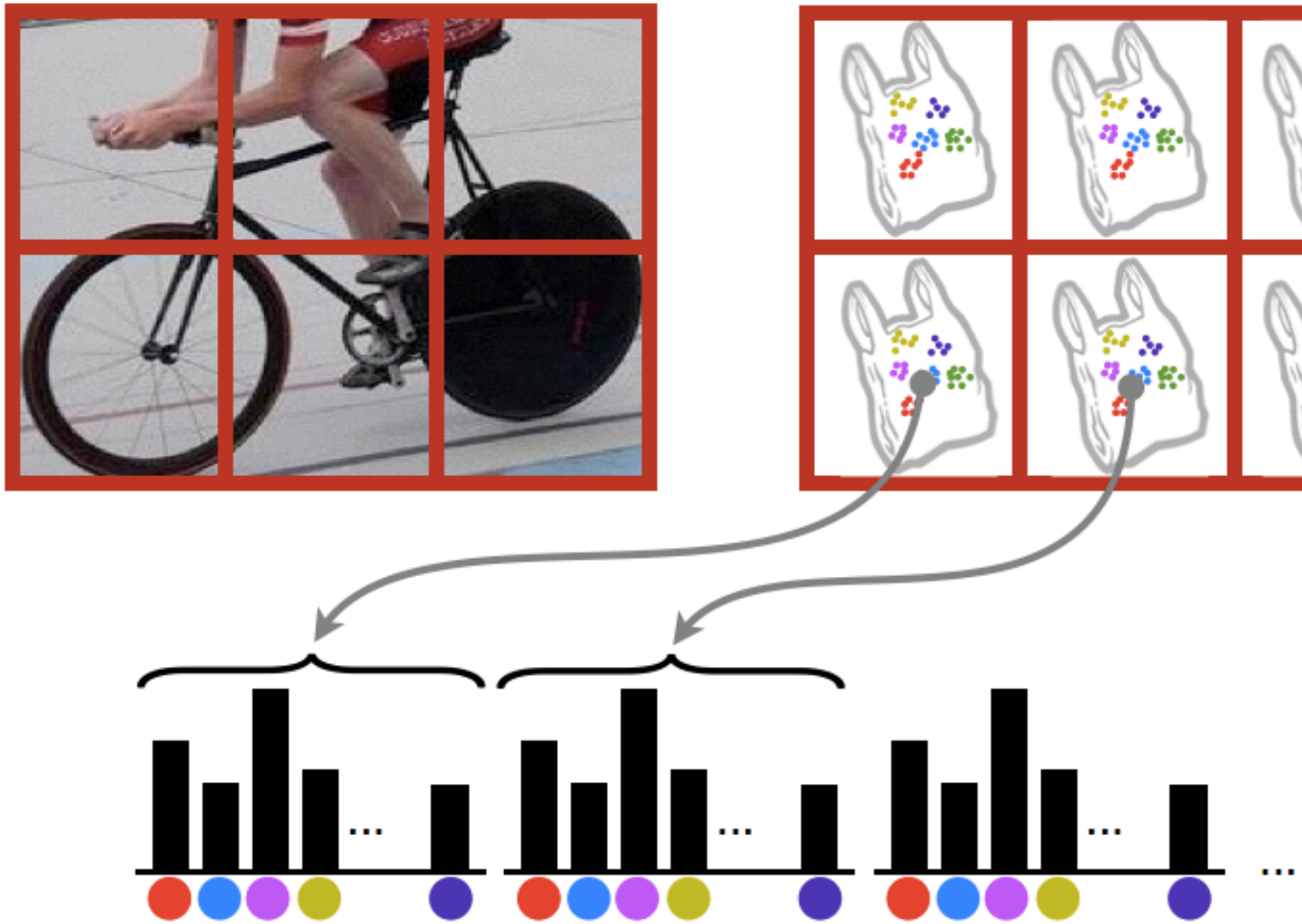
Slide credit: A. Vedaldi

Bag of words limitations



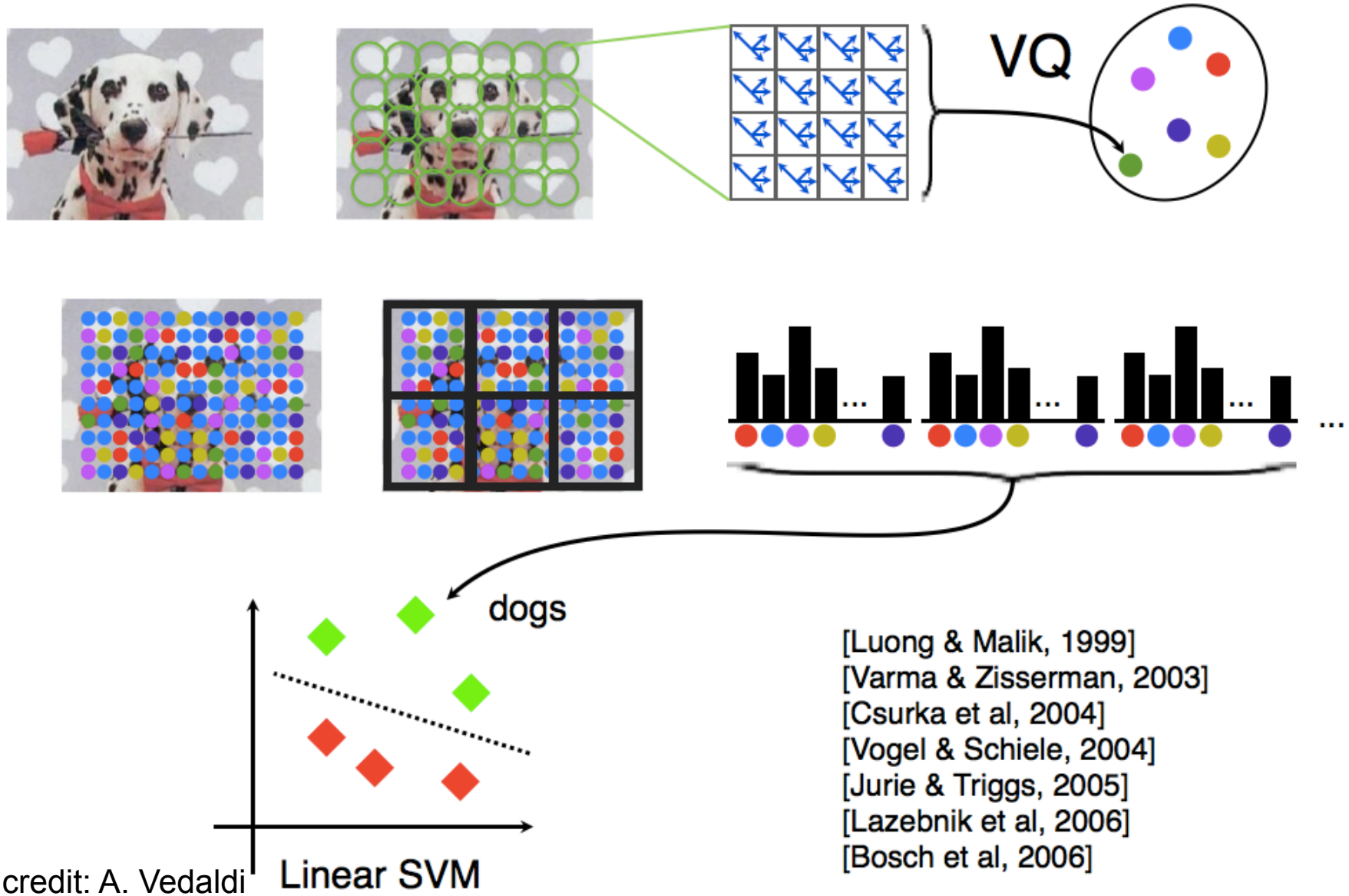
Slide credit: A. Vedaldi

Spatially-sensitive bag-of-words



Slide credit: A. Vedaldi

Image classification in a nutshell



[Luong & Malik, 1999]
[Varma & Zisserman, 2003]
[Csurka et al, 2004]
[Vogel & Schiele, 2004]
[Jurie & Triggs, 2005]
[Lazebnik et al, 2006]
[Bosch et al, 2006]

Slide credit: A. Vedaldi

Beyond Bags of Features: Spatial Pyramid Matching for Recognizing Natural Scene Categories

- Svetlana Lazebnik, Cordelia Schmid, Jean Ponce

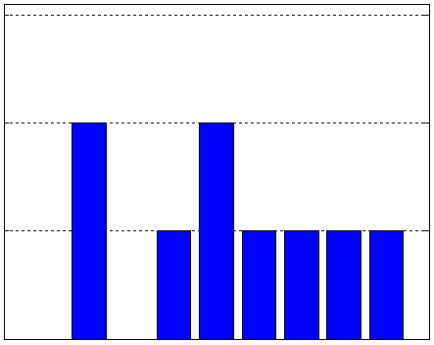
The Pyramid Match Kernel: Discriminative Classification with Sets of Image Features

- Kristen Grauman, Trevor Darrell

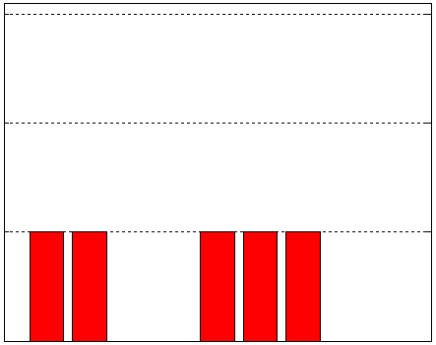
Intersection Kernel

Histogram intersection

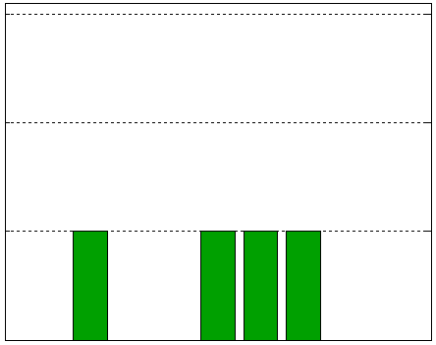
$$\mathcal{I}\left(H(\mathbf{X}), H(\mathbf{Y})\right) = \sum_{j=1}^{r} \min\left(H(\mathbf{X})_j, H(\mathbf{Y})_j\right)$$



$H(\mathbf{X})$



$H(\mathbf{Y})$



$\mathcal{I}\left(H(\mathbf{X}), H(\mathbf{Y})\right) = 4$

Scene category dataset

Fei-Fei & Perona (2005), Oliva & Torralba (2001)

`http://www-cvr.ai.uiuc.edu/ponce_grp/data`

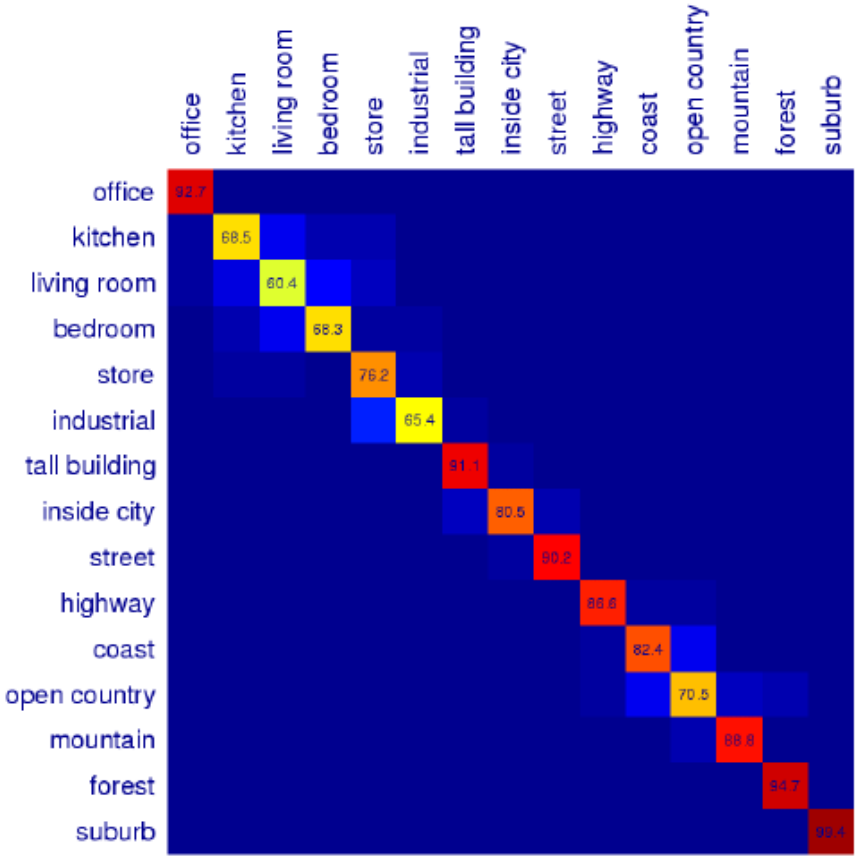
Multi-class classification results (100 training images per class)

	Weak features (vocabulary size: 16)		Strong features (vocabulary size: 200)	
Level	Single-level	Pyramid	Single-level	Pyramid
0 (1×1)	45.3 ±0.5		72.2 ±0.6	
1 (2×2)	53.6 ±0.3	56.2 ±0.6	77.9 ±0.6	79.0 ±0.5
2 (4×4)	61.7 ±0.6	64.7 ±0.7	79.4 ±0.3	**81.1** ±0.3
3 (8×8)	63.3 ±0.8	**66.8** ±0.6	77.2 ±0.4	80.7 ±0.3

Fei-Fei & Perona: 65.2%

Scene category retrieval

Scene category confusions



Difficult indoor images

kitchen

living room

bedroom

Beyond Sliding Windows: Object Localization by *Efficient Subwindow Search*

Christoph H. Lampert[†], Matthew B. Blaschko[†], & Thomas Hofmann[‡]



CVPR 2008 best paper award

Sliding window classifiers

0.1

Sliding window classifiers

-0.2

Sliding window classifiers

. . .

1.5

. . .

Sliding window classifiers

0.5

Sliding window classifiers

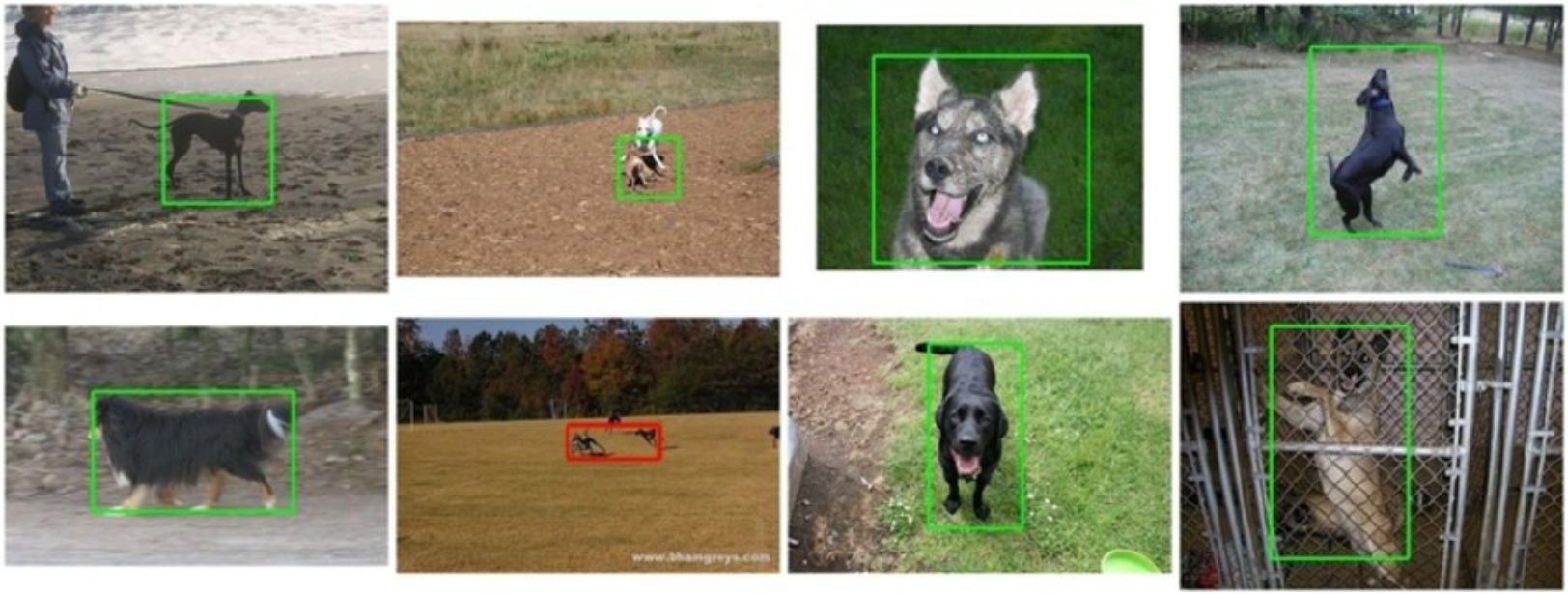
0.3

Sliding window classifiers

0.1
-0.2
-0.1
0.1
. . .
1.5
. . .
0.5
0.4
0.3

Efficient subwindow search: smart search for argmax (test-time)

- High localization quality: first place in 5 of 20 categories.
- High speed: $\approx 40ms$ per image (excl. feature extraction)



Example detections on VOC 2007 dog.

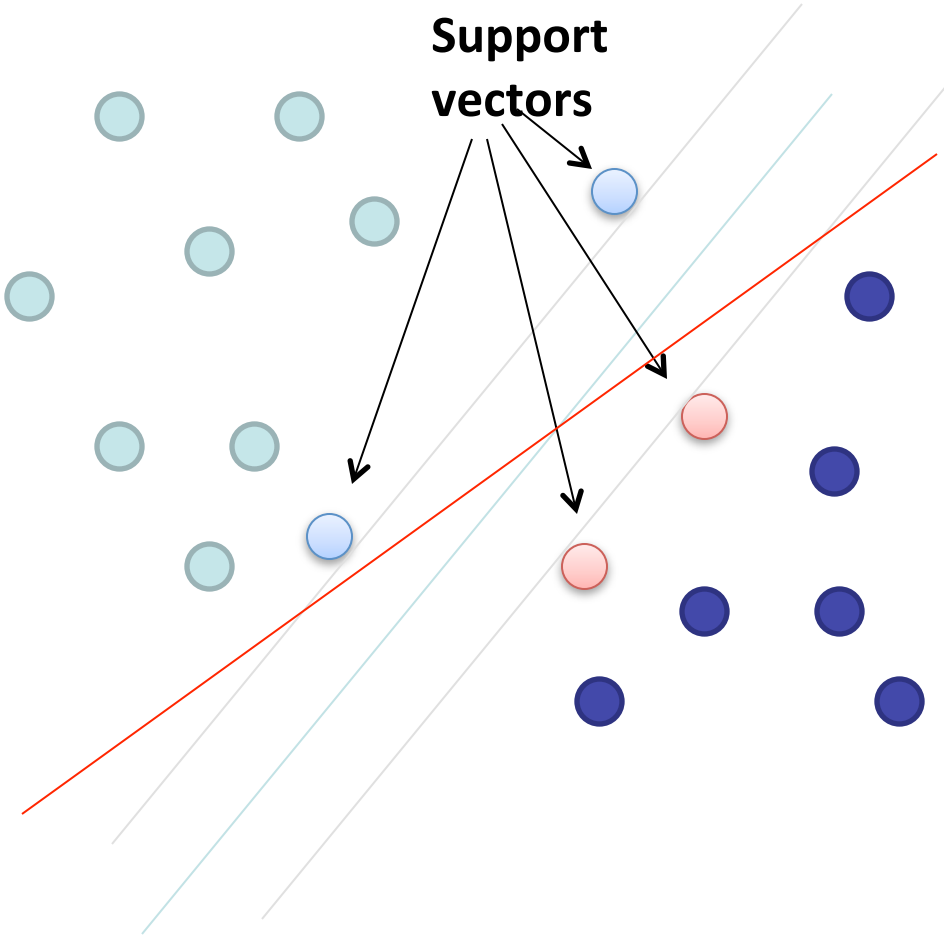
Outline

- Introduction
- Support Vector Machines
- Stochastic Subgradient Descent SVM - Pegasos
- Experiments

Introduction

- Support Vector Machines (SVMs) have become one of the most popular classification tools in the last decade
- Straightforward implementations could not handle large sets of training examples
- Recently methods for solving SVMs arose with linear computational complexity
- Pegasos: primal estimated subgradient approach for solving SVMs

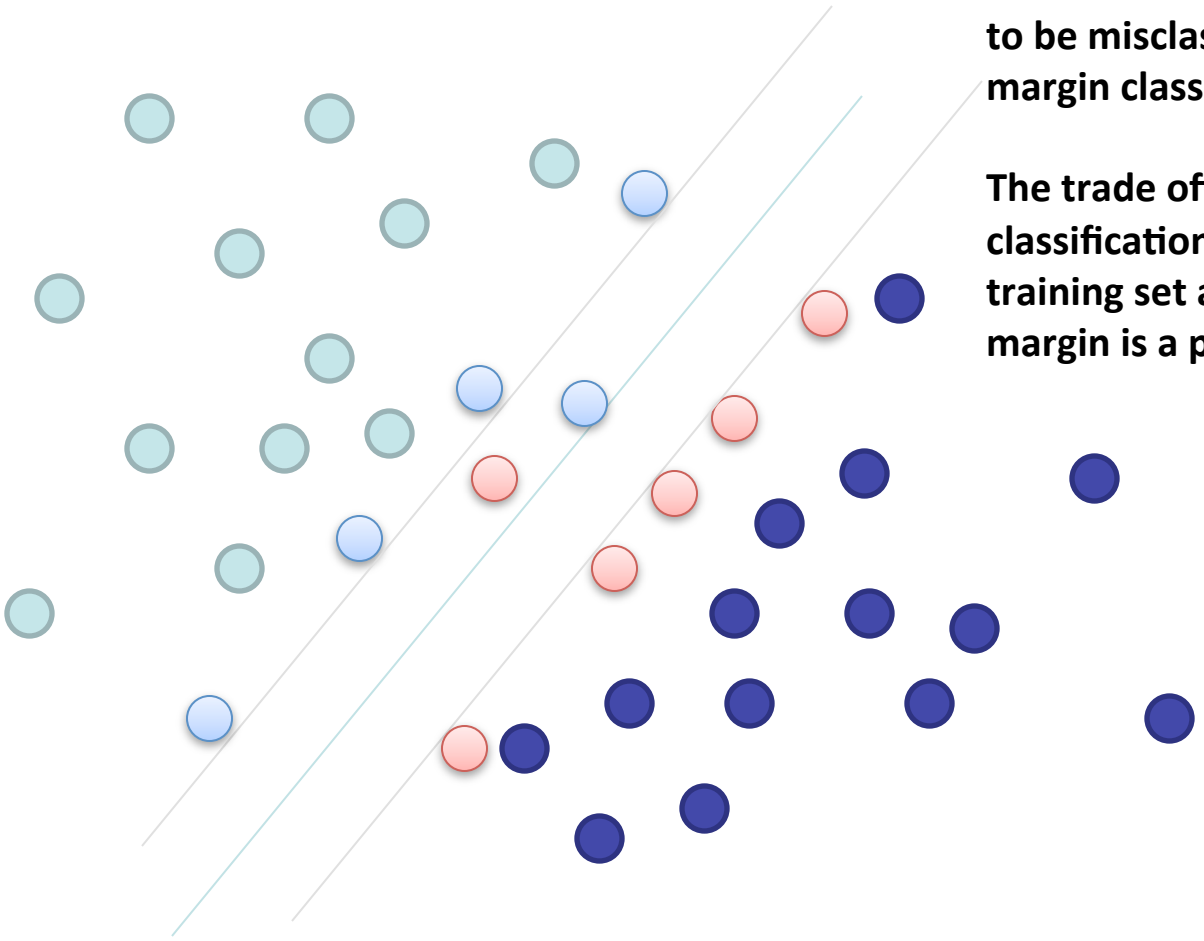
Hard Margin SVM



Many possible hyperplanes perfectly separate the two classes (e.g. red line).

Only one hyperplane with the maximum margin (blue line).

Soft Margin SVM



Allow a small number of examples to be misclassified to find a large margin classifier.

The trade off between the classification accuracy on the training set and the size of the margin is a parameter for the SVM

Problem setting

- Let $S = \{(x_1,y_1),..., (x_m, y_m)\}$ be the set of input-output pairs, where $x_i \in R^N$ and $y_i \in \{-1,1\}$.
- Find the hyperplane with the normal vector $w \in R^N$ and offset $b \in R$ that has good classification accuracy on the training set S and has a large margin.
- Classify a new example x as $sign(w'x - b)$

Optimization problem

- Regularized hinge loss:

Expected hinge loss on the training set

$$min_w \ \lambda/2 \ w'w \ + \ 1/m \ \Sigma_i(1 \ - \ y_i(w'x_i - b))_+$$

Trade off between margin and loss

Size of the margin

Positive for correctly classified examples, else negative

$(1 - z)_+ := \max\{0, 1-z\}$ (hinge loss)

First summand is a quadratic function, the sum is a piecewise linear function. The whole objective: piecewise quadratic.

Perceptron

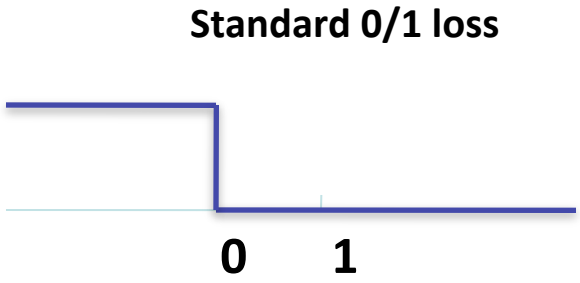
- We ignore the offset parameter b from now on ($b = 0$)
- Regularized Hinge Loss (SVM):

  $min_w \ \lambda/2 \ \ w'w \ + \ 1/m \ \Sigma_i (1 \ - \ \ y_i(w'x_i))_+$
- Perceptron

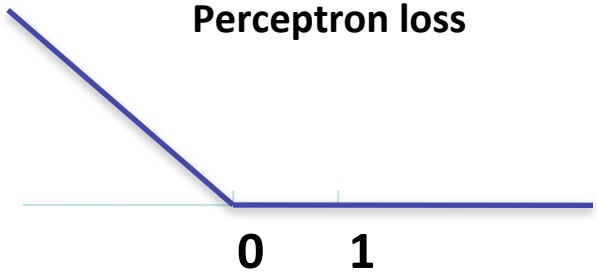
  $min_w \ \ 1/m \ \Sigma_i (-y_i(w'x_i))_+$

Loss functions

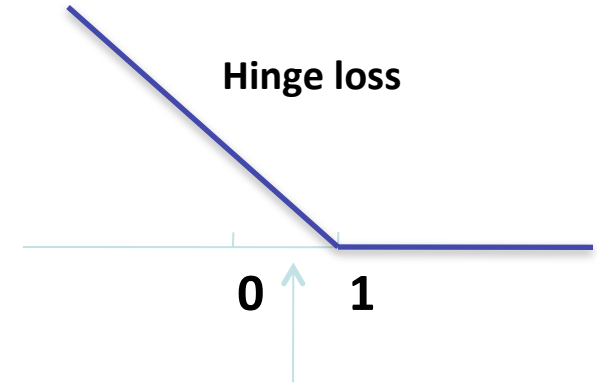
Penalizes all incorrectly classified examples with the same amount



Penalizes incorrectly classified examples x proportionally to the size of |w'x|

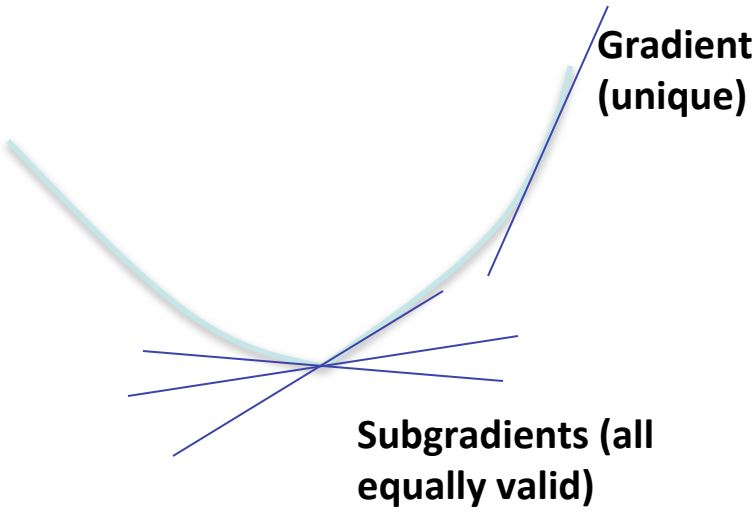


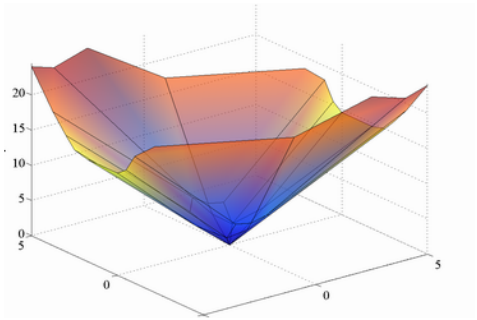
Penalizes incorrectly classified examples and correctly classified examples that lie within the margin



Stochastic Subgradient Descent

- Gradient descent optimization in perceptron (smooth objective)
- Subgradient descent in pegasos (non differentiable objective)





Stochastic Subgradient

- Subgradient in perceptron: $1/m\ \Sigma -y_i x_i$ for all misclassified examples
- Subgradient in SVM: $\lambda w + 1/m\ \Sigma_i (1 - y_i x_i)$ for all misclassified examples
- For every point w the subgradient is a function of the training sample S. We can estimate it from a smaller random subset of S of size k, A, (stochastic part) and speed up computations.
- Stochastic subgradient in SVM:
  $\lambda w + 1/k\ \Sigma_{(x,y)\ \epsilon\ A\ \&\&\ \text{misclassified}} (1 - yx)$

Pegasos – the algorithm

Input: S, λ, T, k

Initialize: Choose w_1 randomly so that $\|w_1\| \leq \frac{1}{\sqrt{\lambda}}$

For $t = 1,2, \ldots, T$

Choose A_t, a random subset of S of size k

Set $B_t = \{(x, y) \in A : y w_t' x < 1\}$

Set $\mu_t = \frac{1}{\lambda t}$

Set $w_{t+\frac{1}{2}} = (1 - \mu_t \lambda) w_t + \frac{\mu_t}{k} \sum_{(x,y) \in B_t} yx$

Set $w_{t+1} = \min\left\{1, \frac{1}{\left(\sqrt{\lambda} \left\| w_{t+\frac{1}{2}} \right\|\right)}\right\} w_{t+\frac{1}{2}}$

Output w_{T+1}

Subsample

Learning rate

Subgradient step

Subgradient is zero on other training points

Projection into a ball (rescaling)

What's new in pegasos?

- Sub-gradient descent technique 50 years old
- Soft Margin SVM 14 years old
- Typically the gradient descent methods suffer from slow convergence
- Authors of Pegasos proved that aggressive decrease in learning rate μ_t still leads to convergence.
  - Previous works: μ_t = 1/(λ√t))
  - pegasos: μ_t = 1/ (λt)
- Proved that the solution always lies in a ball of radius 1/√λ

SVM Light

- A popular SVM solver with superlinear computational complexity
- Solves a large quadratic program
- Solution can be expressed in terms a small subset of training vectors, called support vectors
- Active set method to find the support vectors
- Solve a series of smaller quadratic problems
- **Highly accurate** solutions
- Algorithm and implementation by Thorsten Joachims

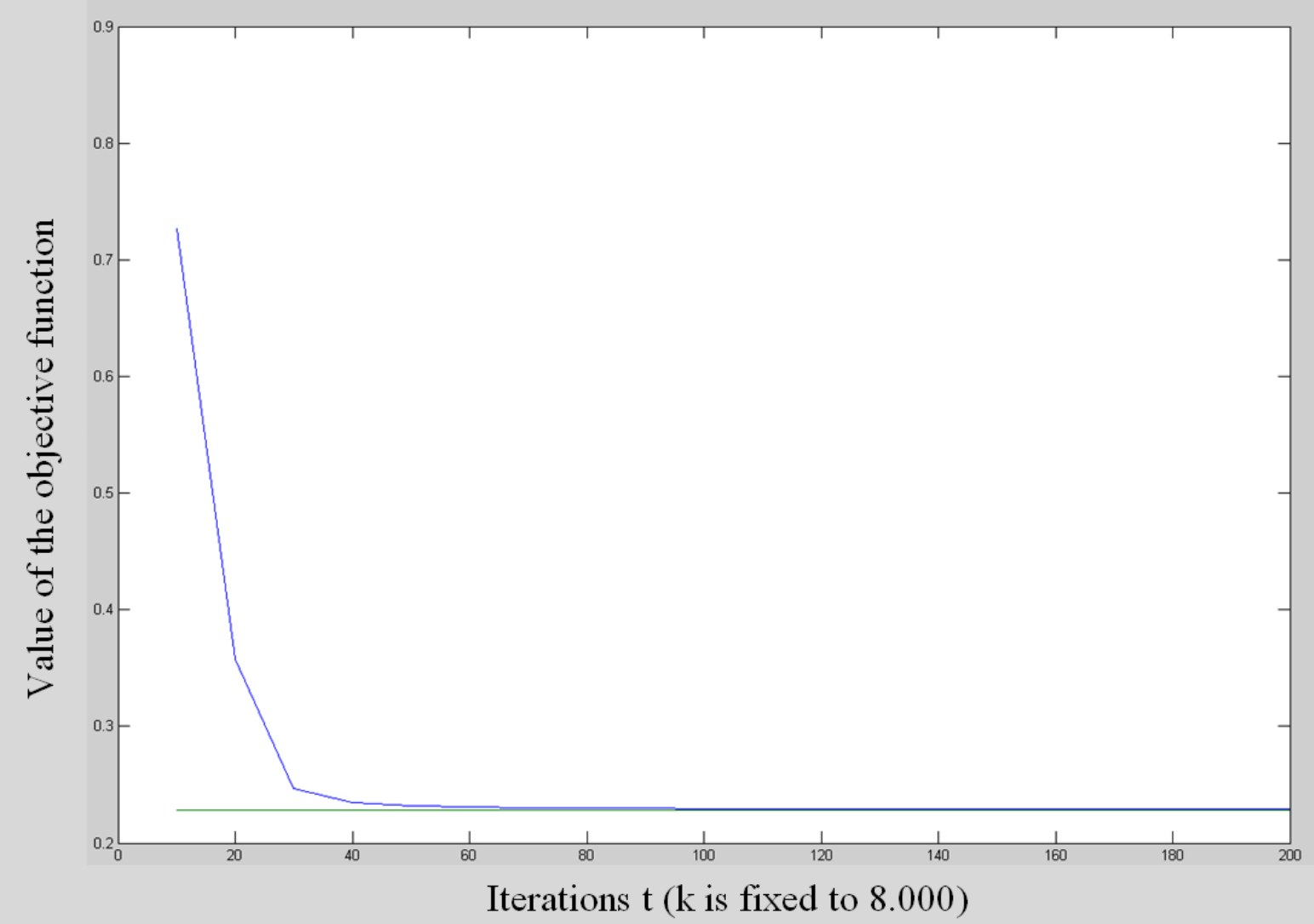
T. Joachims, Making large-Scale SVM Learning Practical. Advances in Kernel Methods - Support Vector Learning, B. Schölkopf and C. Burges and A. Smola (ed.), MIT Press, 1999

Experiments

- Data
  - Reuters RCV2
  - Roughly 800.000 news articles
  - Already preprocessed to bag of word vectors, publicly available
  - Number of features roughly 50.000
  - Sparse vectors
  - category **CCAT**, which consists of 381.327 news

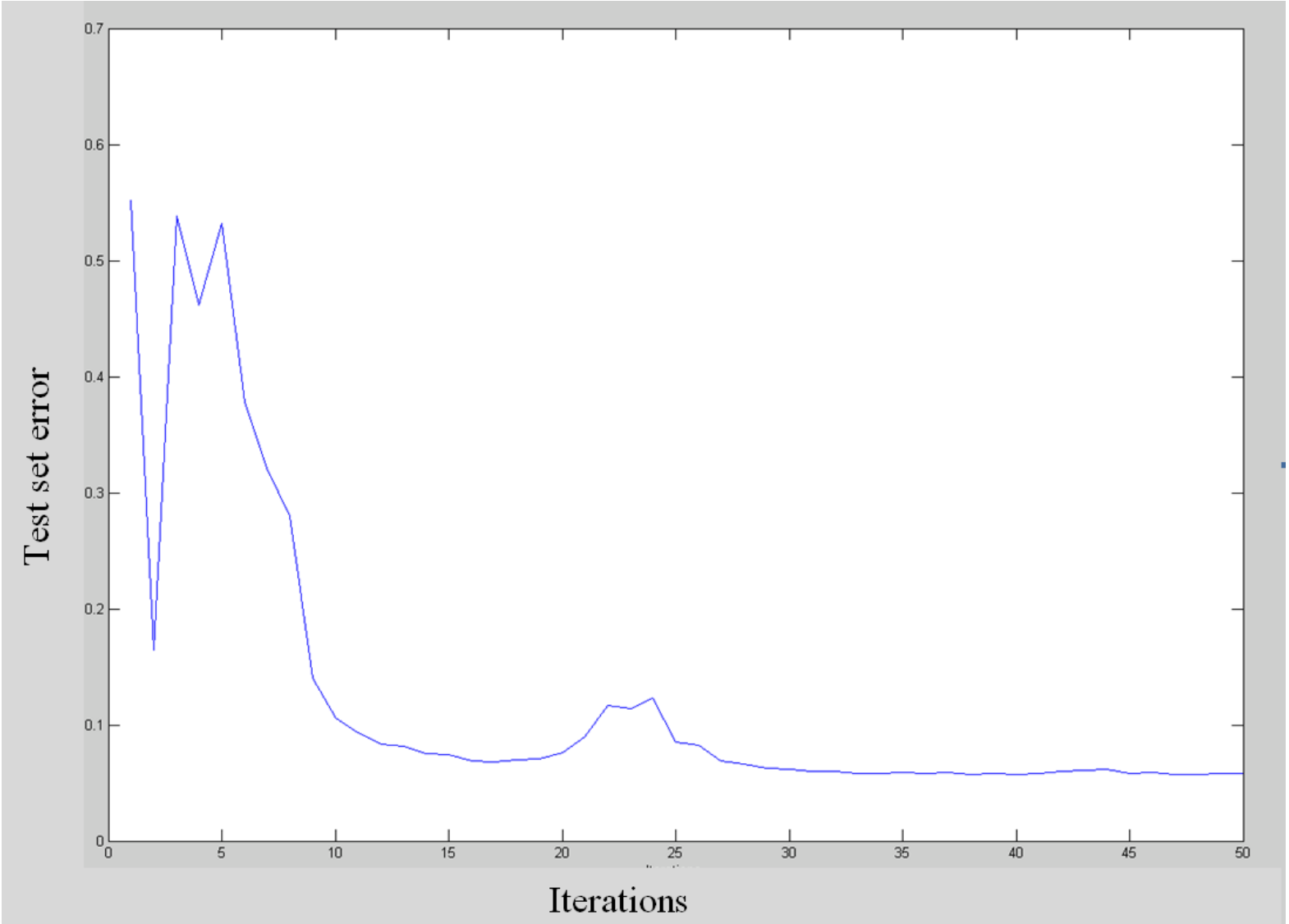
Quick convergence to suboptimal solutions

- 200 iterations took 9.2 CPU seconds, the objective value was 0.3% higher than the optimal solution
- 560 iterations to get a 0.1% accurate solution



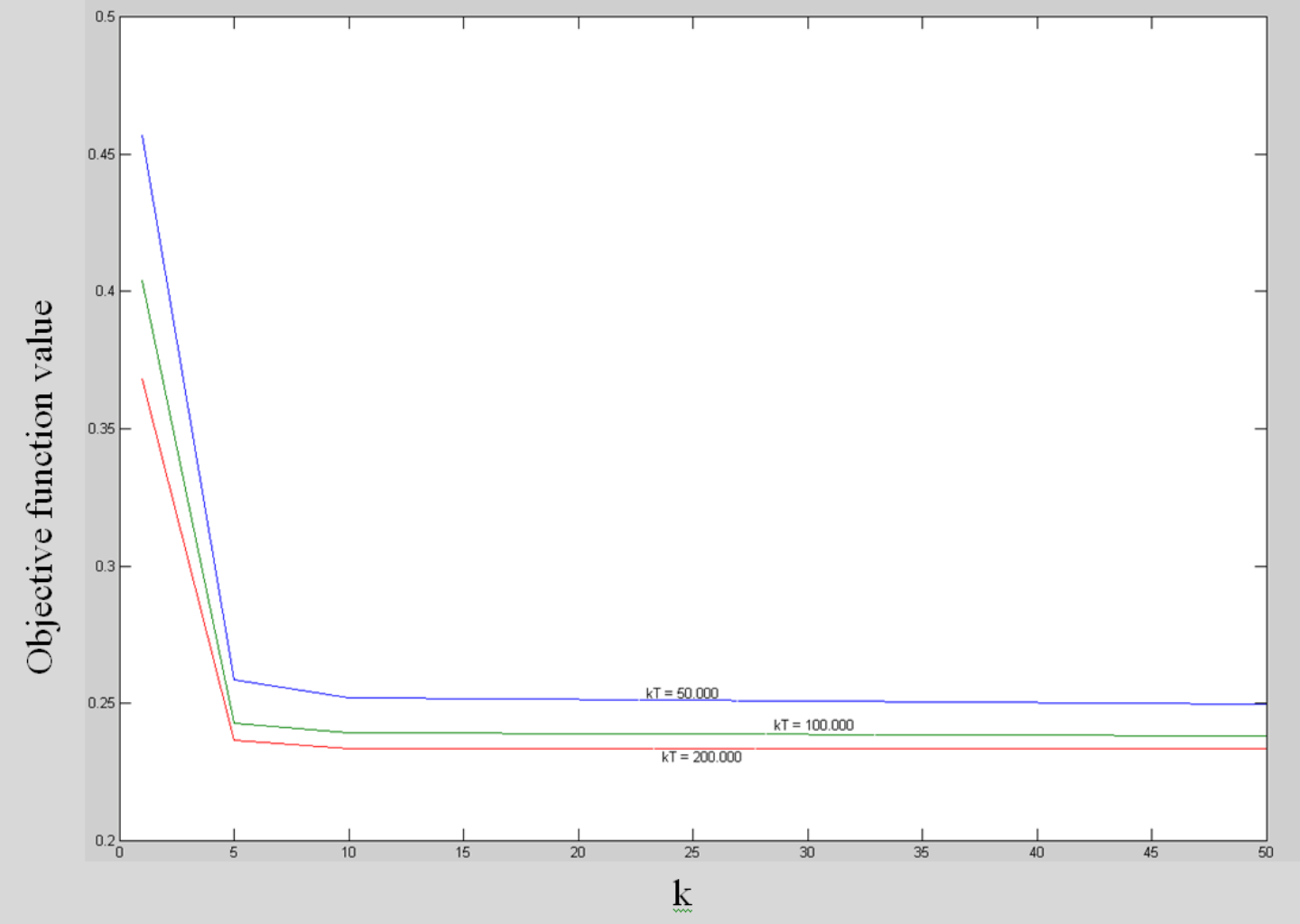
Test set error

- **Optimizing the objective value to a high precission is often not necessary**
- **The lowest error on the test set is achieved much earlier**



Parameters k and T

- **The product of kT determines how close to the optimal value we get**
- **If kT is fixed the k does not play a significant role**



Appendix

Lecture outline

Recap

Large margins and generalization

Optimization

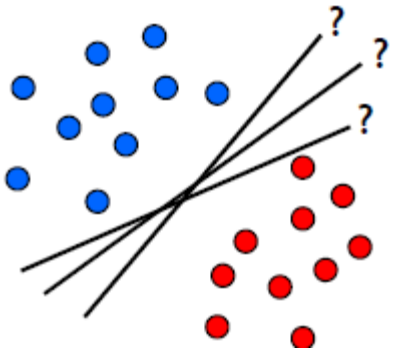
Kernels

Applications to vision

Generalization Error

- What is model complexity?
  - Number of parameters, magnitude of discriminant w?
  - Analyze complexity of hypothesis class

- Linear classifiers:
  - Different decision boundaries
    - Different generalization performance
  - Test error > training error
  - Which line gives smallest test error?



Learning Theory

- V. Vapnik, 1968
  - Mainstream Statistics: Large-sample analysis (`in the limit')
  - Pattern Recognition: Small sample properties

- Distribution-free bounds on worst performance

Empirical and Actual risk

- **Empirical risk**
  - Measured on the training/validation set

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

- **Actual risk (= Expected risk)**
  - Expectation of the error on *all* data.

$$R(\alpha) = \int L(y_i, f(\mathbf{x}; \alpha)) dP_{X,Y}(\mathbf{x}, y)$$

  - $P_{X,Y}(\mathbf{x}, y)$ is the probability distribution of $(\mathbf{x}, y)$. It is fixed, but typically unknown.

Actual and Empirical Risk

- Idea
  - Compute an upper bound on the actual risk based on the empirical risk

$$R(\alpha) \leq R_{emp}(\alpha) + \epsilon(N, p^*, h)$$

  - where

    N: number of training examples

    p^*: probability that the bound is correct

    h: capacity of the learning machine ("VC-dimension")

- With probability $(1\text{-}\eta)$, the following bound holds

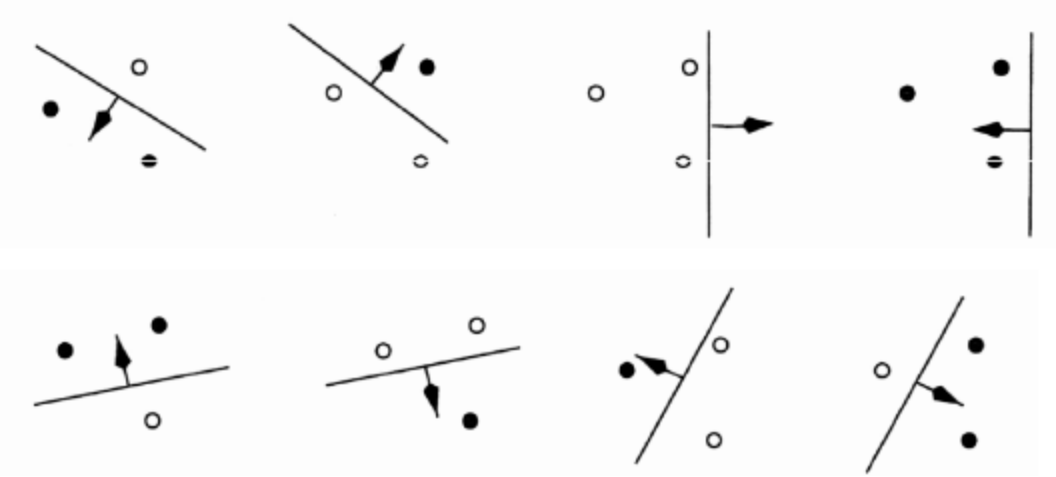
$$R(\alpha) \leq R_{emp}(\alpha) + \sqrt{\frac{h(\log(2N/h) + 1) - \log(\eta/4)}{N}}$$

Vapnik Chervonenkis (VC) Dimension

- Shattering: *If a given set of ℓ points can be labeled in all possible 2^ℓ ways, and for each labeling, a member of the set $\{f(\alpha)\}$ can be found which correctly assigns those labels, we say that the set of points is shattered by the set of functions.*

- VC dimension *The VC dimension for the set of functions $\{f(\alpha)\}$ is defined as the maximum number of training points that can be shattered by $\{f(\alpha)\}$.*

- Example



Large Margins & VC Dimension

- Vapnik: *The class of optimal linear separators has VC dimension h bounded from above as*

$$h \leq \min\left\{\left\lceil \frac{D^2}{\rho^2} \right\rceil, m_0\right\} + 1$$

where ρ is the margin, D is the diameter of the smallest sphere that can enclose all of the training examples, and m_0 is the dimensionality.

- If we maximize the margins, feature dimensionality does not matter

Number of parameters does not matter by itself!

Margin does!

Appendix: Duality

- Constrained optimization problem:

$$\begin{aligned} \min_{w} \quad & f(w) \\ s.t. \quad & h_i(w) = 0, \quad i = 1 \dots l \\ & g_i(w) \leq 0, \quad i = 1 \dots m \end{aligned}$$

- Equivalent to unconstrained problem:

$$\min_{w} \quad f_{uc}(w) = f(w) + \sum_{i=1}^{l} I_0(h_i(w)) + \sum_{i=1}^{m} I_+(g_i(w))$$

$$I_0(x) = \begin{cases} 0, & x = 0 \\ \infty, & x \neq 0 \end{cases}, \qquad I_+(x) = \begin{cases} 0, & x \leq 0 \\ \infty, & x > 0 \end{cases}$$

- Soften constraint terms $I_0(x_i) \rightarrow \lambda_i x_i \qquad I_+(x_i) \rightarrow \mu_i x_i, \mu > 0$

Lagrangian

- Replace hard constraints with soft ones

$$\min_w \quad f_{uc}(w) = f(w) + \sum_{i=1}^{l} I_0(h_i(w)) + \sum_{i=1}^{m} I_+(g_i(w))$$

$$L(w, \lambda, \mu) = f(w) + \sum_{i=1}^{l} \lambda_i h_i(w) + \sum_{i=1}^{m} \mu_i g_i(w), \quad \mu_i > 0 \forall i$$

- Observe that

$$f_{uc}(w) = \max_{\lambda, \mu : \mu_i > 0} L(w, \lambda, \mu)$$

- At an optimum:

$$f(w^*) = \min_w \max_{\lambda, \mu : \mu_i > 0} L(w, \lambda, \mu)$$

Lagrange Dual Function

- Form $$\theta(\lambda, \mu) = \inf_{w} L(w, \lambda, \mu)$$

- θ : lower bound on optimal value of the original problem

$$\begin{aligned} L(w^*, \lambda, \mu) &= f(w^*) + \sum_{i=1}^{l} \lambda_i h_i(w^*) + \sum_{i=1}^{m} \mu_i g_i(w^*) = \\ &\overset{w^*:feasible}{=} f(w^*) + \sum_{i=1}^{l} \lambda_i 0 + \sum_{i=1}^{m} \mu_i \underbrace{g_i(w^*)}_{<0} = \\ &\overset{\mu_i \geq 0}{\leq} f(w^*) \end{aligned}$$

- Therefore: $$\theta(\lambda, \mu) = \inf_{w} L(w, \lambda, \mu) \leq L(w^*, \lambda, \mu) \leq f(w^*)$$

Dual Problem

- Maximize the lower bound on the cost of the primal

$$\max_{\lambda,\mu} \quad \theta(\lambda,\mu)$$
$$s.t. \quad \mu_i > 0 \quad \forall i$$

- In general:

$$\begin{aligned} d^* &= \max_{\lambda,\mu} \theta(\lambda,\mu) \\ &= \max_{\lambda,\mu:\mu_i>0} \min_{w} L(w,\lambda,\mu) \\ &\leq \min_{w} \max_{\lambda,\mu:\mu_i>0} L(w,\lambda,\mu) \\ &= \min_{w} f_{uc}(w) = p^* \end{aligned}$$

- For convex cost and convex constraints (SVM case): $d^* = p^*$

Complementary Slackness

- Assume $d^* = p^*$
- There exists a feasible solution w^*, λ^*, μ^* to the primal and dual problems, such that $f(w^*) = \theta(\lambda^*, \mu^*)$
- We will have

$$\begin{aligned} f(w^*) &= \theta(\lambda^*, \mu^*) \\ &= \inf_w f(w) + \sum_{i=1}^{l} \lambda_i^* h_i(w) + \sum_{i=1}^{m} \mu_i^* f_i(w) \\ &\leq f(w^*) + \sum_{i=1}^{M} \lambda_i^* h_i(w^*) + \sum_{i=1}^{m} \mu_i^* f_i(w^*) \\ &\leq f(w^*) \end{aligned}$$

- This means $\mu_i^* f_i(w^*) = 0, \quad \forall i$

Karush-Kuhn Tucker (KKT) Conditions

- Solution of the primal problem:
  - minimum of the Lagrangan w.r.t. the primal variables
  - therefore $$\nabla f(w^*) + \sum_{i=1}^{l} \lambda_i \nabla h_i(w^*) + \sum_{i=1}^{m} \nabla f_i(w^*) = 0$$

- Putting all constraints together: KKT conditions

$$\begin{aligned} h_i(w^*) &= 0 \\ f_i(w^*) &\leq 0 \\ \mu_i f_i(w^*) &= 0 \\ \mu_i &\geq 0 \\ \nabla f(w^*) + \sum_{i=1}^{l} \lambda_i \nabla h_i(w^*) + \sum_{i=1}^{m} \nabla f_i(w^*) &= 0 \end{aligned}$$

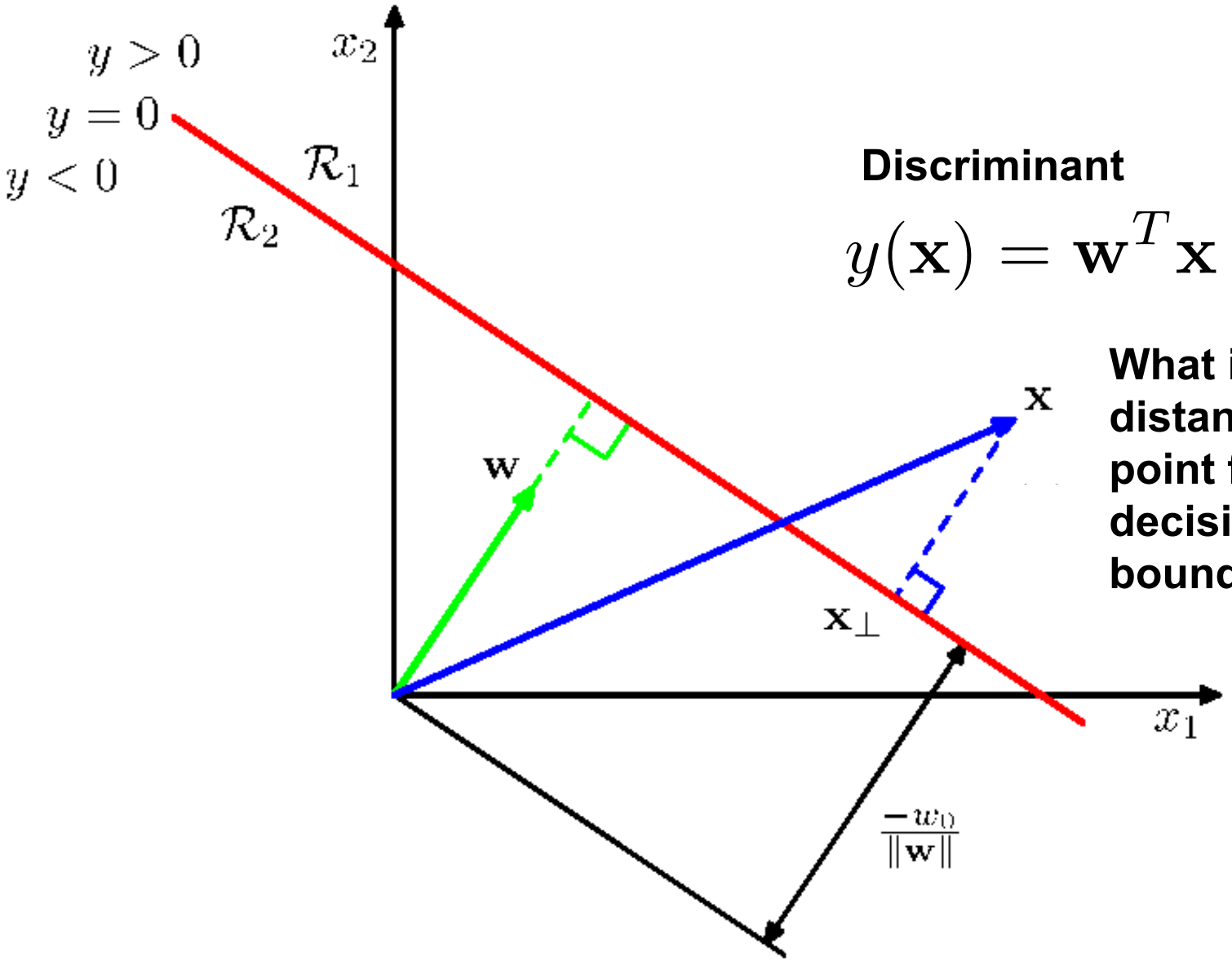
Update

References: on moodle

Online resource: David Barber’s book

http://web4.cs.ucl.ac.uk/staff/D.Barber/pmwiki/pmwiki.php?n=Brml.HomePage

Geometric Margins



Discriminant

$$y(\mathbf{x}) = \mathbf{w}^T\mathbf{x} + w_0$$

What is the distance of this point from the decision boundary?

Lecture outline

- Recap
- Large margins and generalization
- Optimization
- Kernels
- Applications to vision
  - Strong models
  - Loose models

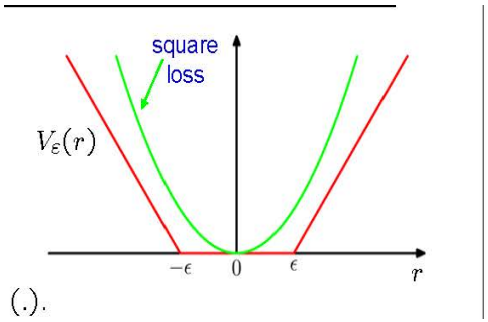
Support Vector Regression

- Combines loss function and flatness as a single objective.
- Support vector machines developed by Vapnik and coworkers for optical character recognitions in Russia in the 1960s.
- First use for regression in 1997.
- Besides regression has become popular as classifier to divide design space into feasible domain (where constraints are satisfied) and infeasible domain (where they are not).

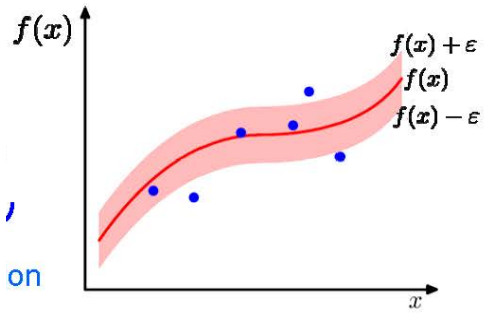
Epsilon-insensitive loss function

- Support vector regression can use any loss function, but the one most often associated with it is epsilon-insensitive.
- It is less sensitive to one bad data point.

$$V_\varepsilon(r) = \begin{cases} 0 & if\ |r| < \varepsilon \\ |r| - \varepsilon & otherwise \end{cases}$$



cost is zero inside epsilon "tube"



Figures from Gunn's Support Vector Machines for Classification and Regression

SVM Recall

Two-class classification problem using linear model:

$$y(x) = w^T \phi(x) + b$$

Regularized Error Function

In linear regression, we minimize the error function:

$$\frac{1}{2}\sum_{n=1}^{N}\{y_n - t_n\}^2 + \frac{\lambda}{2}\|w\|^2$$

Replace the quadratic error function by Є-insensitive error function:

$$C\sum_{n=1}^{N} E_\varepsilon(y(x_n) - t_n) + \frac{1}{2}\|w\|^2$$

An example of Є-insensitive error function:

$$L_\epsilon(y) = \begin{cases} 0 & \text{for} \quad |f(\mathbf{x}) - y| < \varepsilon \\ |f(\mathbf{x}) - y| - \varepsilon & \text{otherwise} \end{cases}$$

Slack Variables

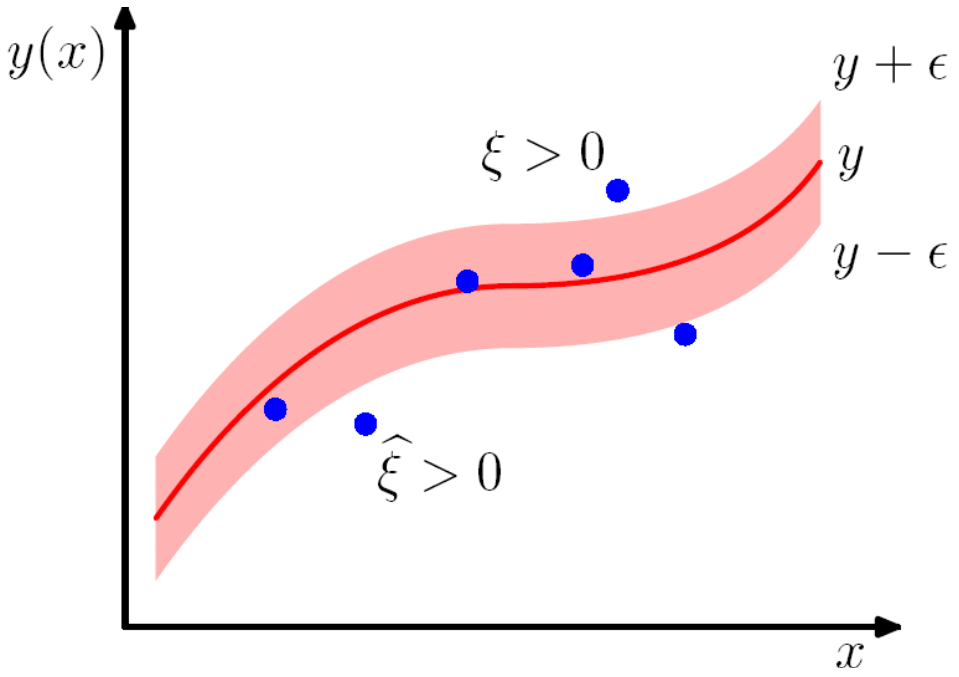
For a target point to lie inside the tube:

$$y_n - \in \leq t_n \leq y_n + \in$$

Introduce slack variables to allow points to lie outside the tube:

$$t_n \leq y(x_n) + \in + \xi_n$$

$$t_n \geq y(x_n) - \in - \xi_n^-$$



Error Function for Support Vector Regression

Minimize:

$$C\sum_{n=1}^{N}(\xi_n + \xi_n^-) + \frac{1}{2}\|w\|^2$$

Subject to:

$$\xi_n \geq 0 \qquad \textbf{and} \qquad t_n \leq y(x_n) + \in + \xi_n$$

$$\xi_n^- \geq 0 \qquad t_n \geq y(x_n) - \in - \xi_n^-$$

Lagrangian

Minimize:

$$L = C\sum_{n=1}^{N}(\xi_n + \xi_n^-) + \frac{1}{2}\|w\|^2 - \sum_{n=1}^{N}(\mu_n\xi_n + \mu_n^-\xi_n^-) - \sum_{n=1}^{N}a_n(\in+\xi_n + y_n - t_n) - \sum_{n=1}^{N}a_n^-(\in+\xi_n^- - y_n + t_n)$$

$$\frac{\partial L}{\partial w} = 0 \Rightarrow w = \sum_{n=1}^{N}(a_n - a_n^-)\phi(x_n)$$

$$\frac{\partial L}{\partial b} = 0 \Rightarrow \sum_{n=1}^{N}(a_n - a_n^-) = 0$$

$$\frac{\partial L}{\partial \xi_n} = 0 \Rightarrow a_n + \mu_n = C$$

$$\frac{\partial L}{\partial \xi_n^-} = 0 \Rightarrow a_n^- + \mu_n^- = C$$

Dual Form of Lagrangian

Maximize:

$$W(a,a^-) = -\frac{1}{2}\sum_{n=1}^{N}\sum_{m=1}^{N}(a_n - a_n^-)(a_m - a_m^-)k(x_n,x_m) - \in\sum_{n=1}^{N}(a_n + a_n^-) + \sum_{n=1}^{N}(a_n - a_n^-)t_n$$

$$0 \le a_n \le C$$

$$0 \le a_n^- \le C$$

Prediction can be made using:

$$y(x) = \sum_{n=1}^{N}(a_n - a_n^-)k(x,x_n) + b$$

How to determine b?

Karush-Kuhn-Tucker (KKT) conditions:

$$a_n(\in+\xi_n + y_n - t_n) = 0$$

$$a_n^-(\in+\xi_n^- - y_n + t_n) = 0$$

$$(C - a_n)\xi_n = 0$$

$$(C - a_n^-)\xi_n^- = 0$$

Support vectors are points that lie on the boundary or outside the tube

$$b = t_n - \in - w^T\phi(x_n) = t_n - \in - \sum_{m=1}^{N}(a_m - a_m^-)k(x_n, x_m)$$