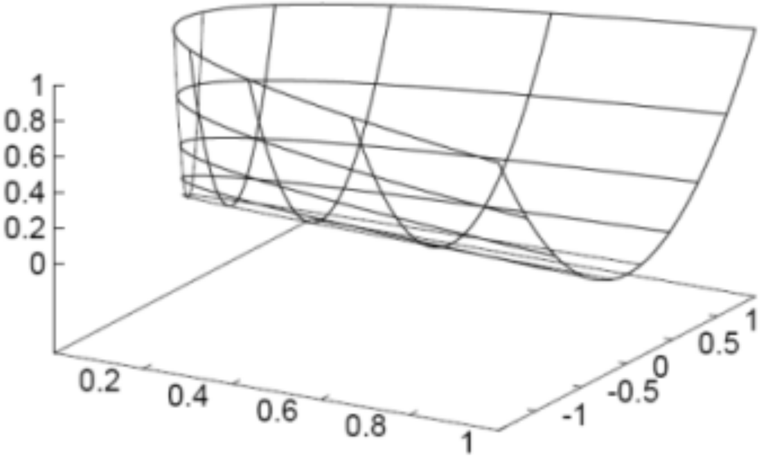
Introduction to Supervised Learning



Week 3:
Support Vector Machines

Iasonas Kokkinos

i.kokkinos@cs.ucl.ac.uk

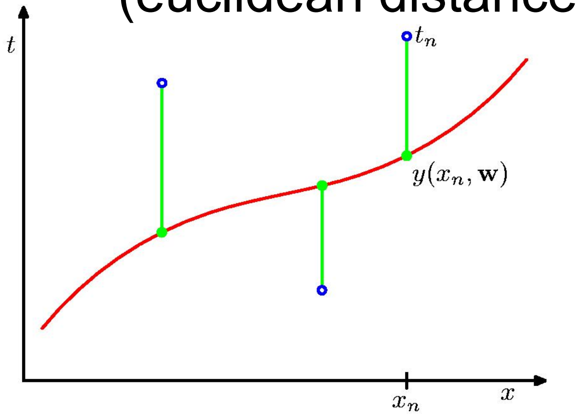
University College London

Lecture outline

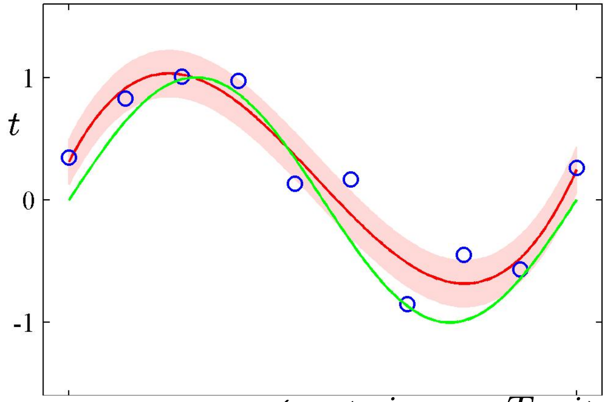
- Introduction to Support Vector Machines
  - Geometric margins
  - Training criterion & hinge loss
- Large margins and generalization
- Optimization
- Kernels
- Applications to vision

Our path so far (week 1-2)

Week 1 - regression: geometric (euclidean distance)

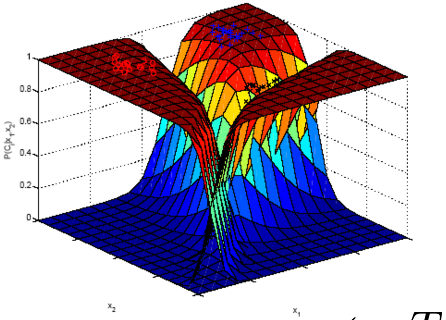


Week 2: probabilistic interpretation



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

Week 2: switch to classification

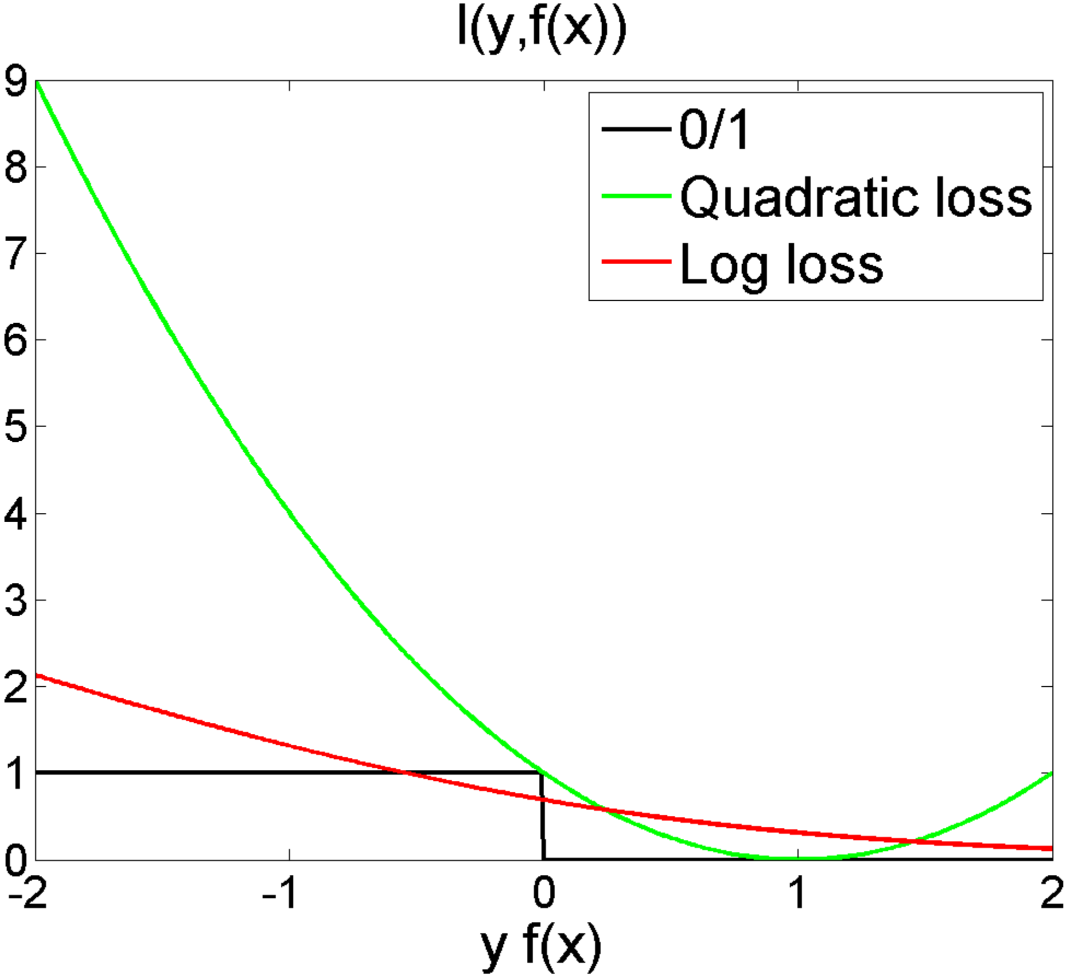


$$P(y^i|\mathbf{x}^i) = \frac{\exp(\mathbf{w}_c^T\mathbf{x})}{\sum_{c'=1}^{C} \exp(\mathbf{w}_{c'}^T\mathbf{x})}$$

?

geometry + classification?

Week 2: log loss vs. quadratic loss



Quadratic loss

$$l(y, f(x)) = (1 - yf(x))^2$$

Log loss

$$l(y, f(x)) = \log(1 + \exp(-yf(x)))$$

Do we need the logistic loss?

Week 2: Useful criterion for training classifiers

Maybe we can quickly hack an easy algorithm

Least squares: Gauss, 1795

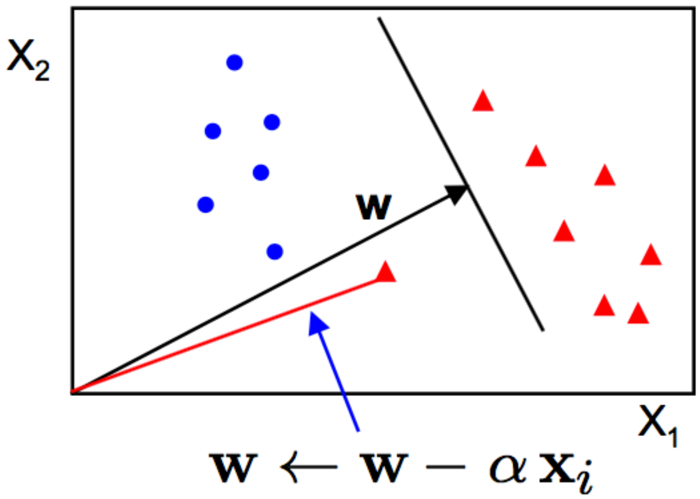
Logistic Regression: Cox, 1958

Perceptrons, Minsky & Papert, 1969

Perceptron algorithm

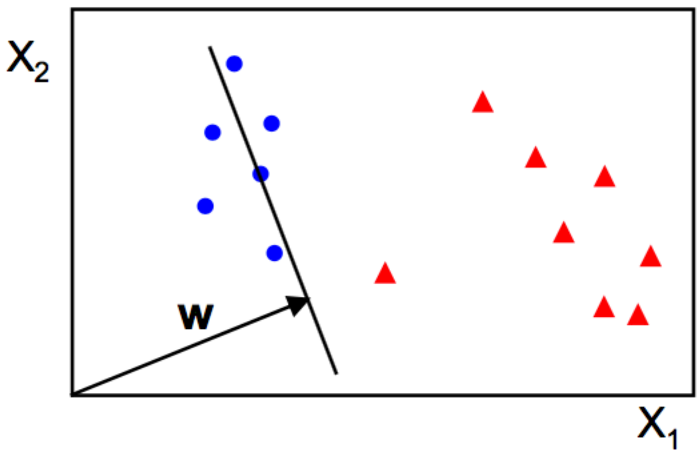
- Initialize **w** = 0
- Cycle though the data points { $\mathbf{x}_i$, y_i }
  - if $\mathbf{x}_i$ is misclassified then $\mathbf{w} \leftarrow \mathbf{w} + \alpha\,\mathrm{sign}(f(\mathbf{x}_i))\,\mathbf{x}_i$
- Until all the data is correctly classified

before update



$\mathbf{w} \leftarrow \mathbf{w} - \alpha\,\mathbf{x}_i$

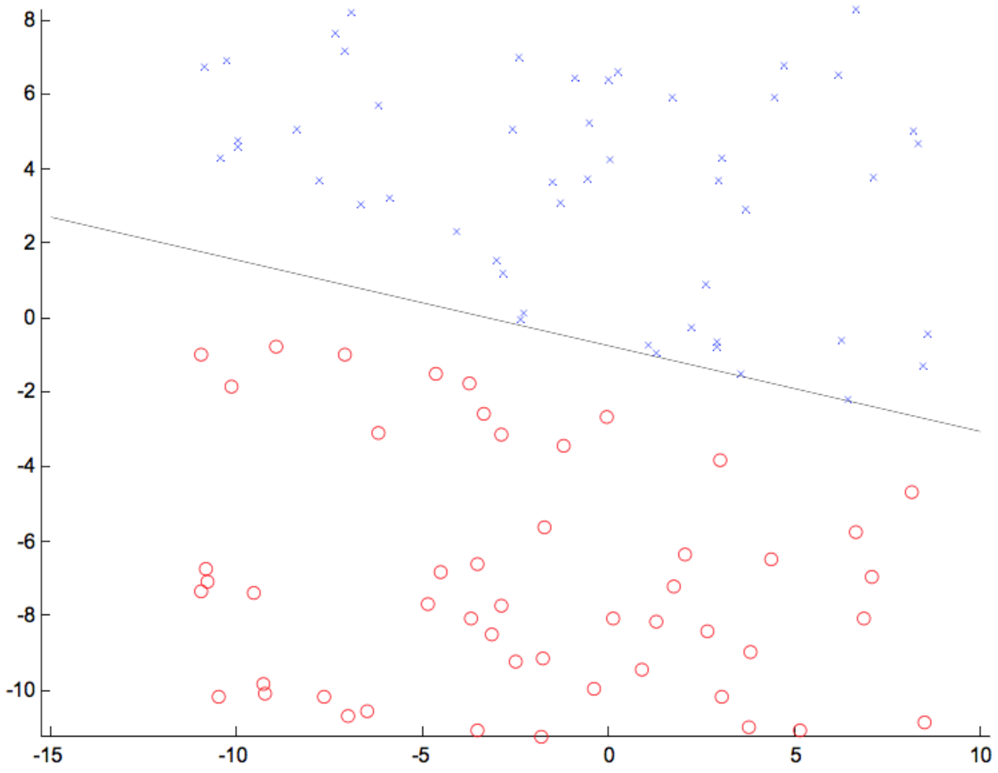
after update



after convergence $\mathbf{w} = \sum_i^N \alpha_i \mathbf{x}_i$

Perceptron algorithm (first ‘neural network’)

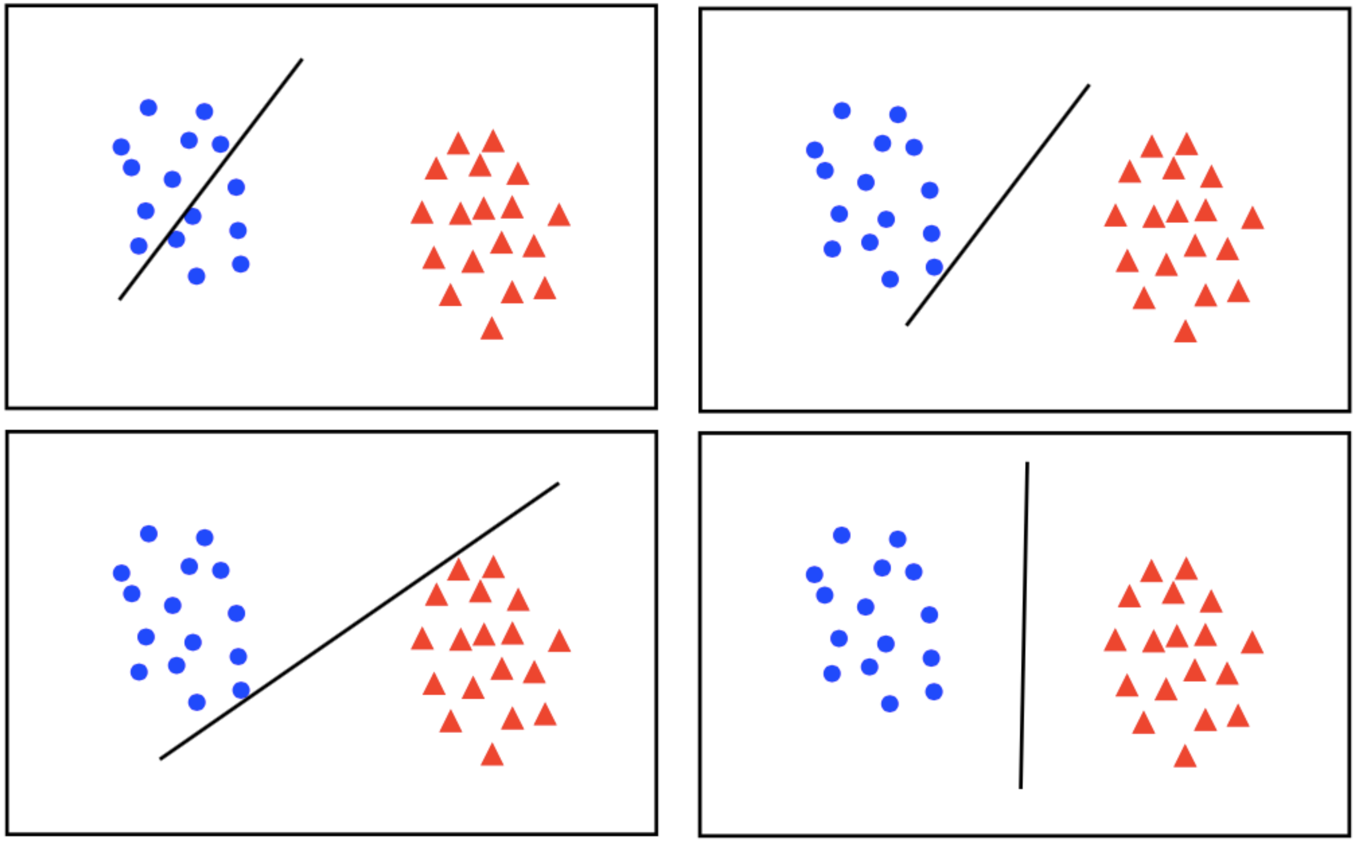
Perceptron example



- if the data is linearly separable, then the algorithm will converge
- convergence can be slow …
- separating line close to training data

This lecture: push it far away!

Which classifier is best?



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

How can we quantify this?

How can we enforce this?

Functional Margins

Consider Logistic Regression:

$$P(y=1|\mathbf{x};\mathbf{w}) = g(\mathbf{w}^T\mathbf{x}) = \frac{1}{1+\exp(-\mathbf{w}^T\mathbf{x})}$$

Ideally:

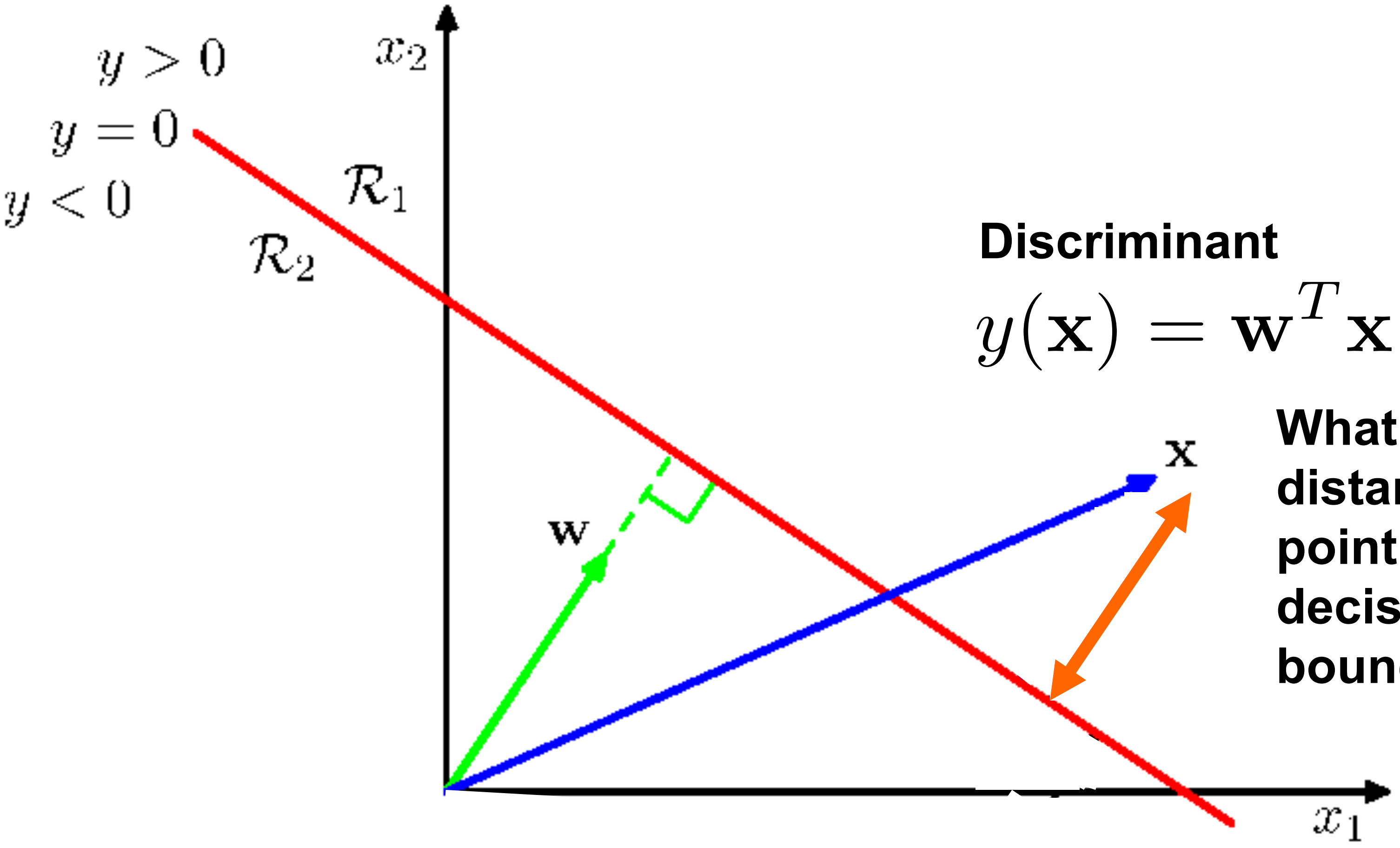
$$\begin{aligned} \mathbf{w}^T\mathbf{x}^i &\gg 0, \quad \text{if} \quad y^i = 1 \\ \mathbf{w}^T\mathbf{x}^i &\ll 0, \quad \text{if} \quad y^i = -1 \end{aligned}$$

Put together: $y^i(\mathbf{w}^T\mathbf{x}^i) \gg 0$

`functional margin’

Problem: scaling **w** changes functional margin, but not decision boundary

Geometric Margins



Discriminant

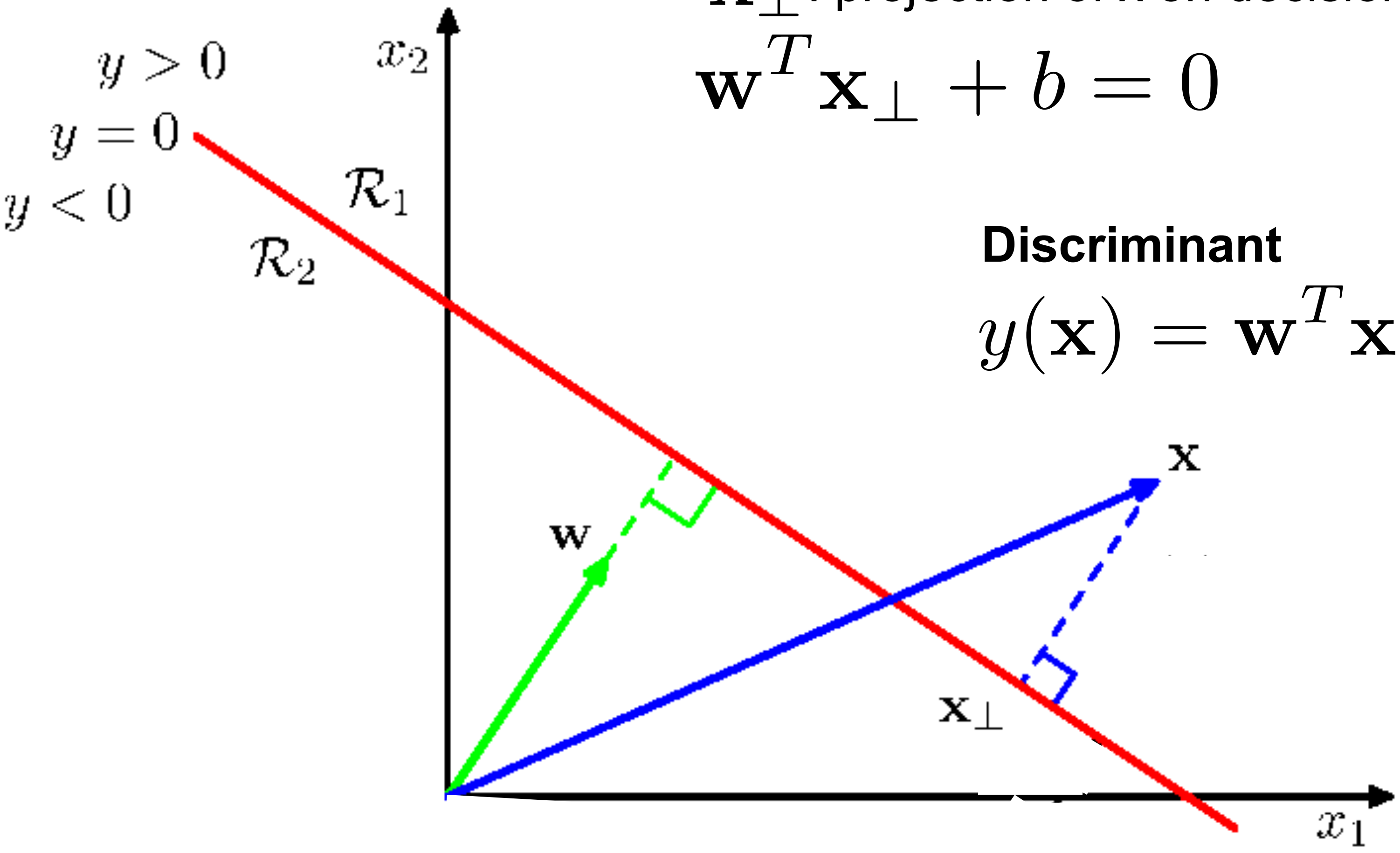
$$y(\mathbf{x}) = \mathbf{w}^T\mathbf{x} + b$$

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

Geometric Margins

$\mathbf{x}_\perp$: projection of x on decision boundary

$$\mathbf{w}^T\mathbf{x}_\perp + b = 0$$

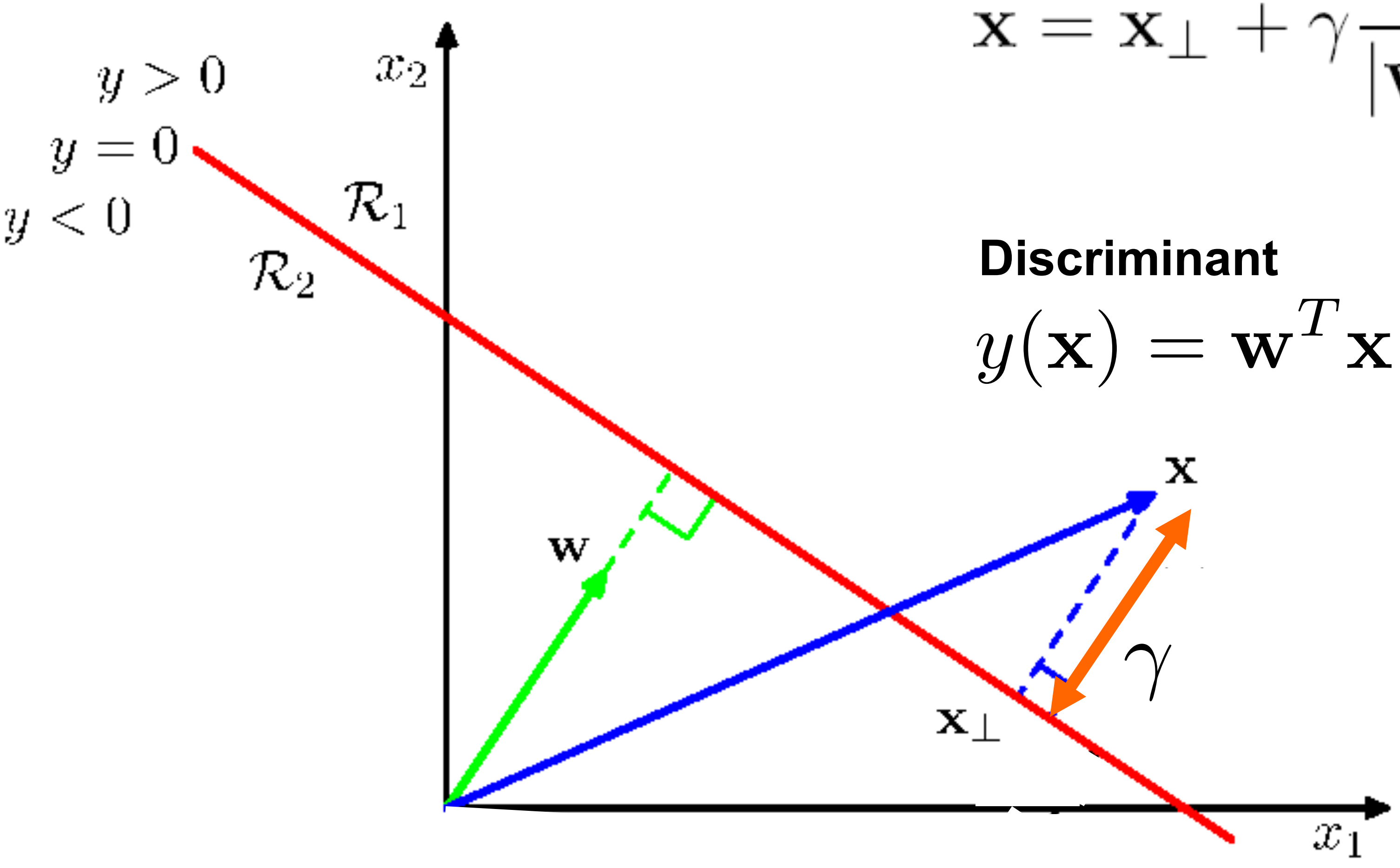


Discriminant

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

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

Point = projection + distance* direction

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

Note: γ is independent of |w|

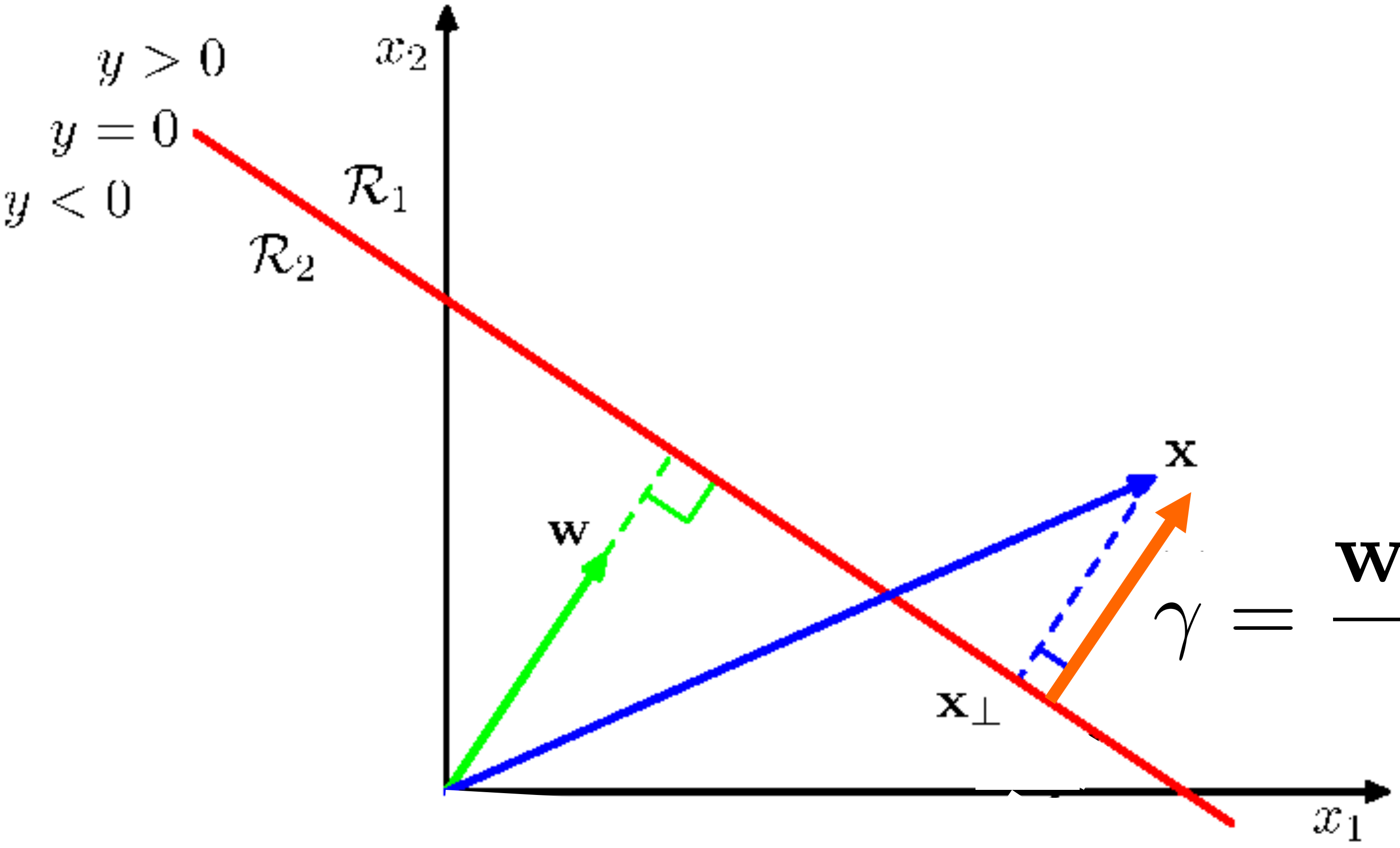
Multiply: $$\mathbf{w}^T \mathbf{x} = \mathbf{w}^T \mathbf{x}_{\perp} + \mathbf{w}^T \gamma \frac{\mathbf{w}}{|\mathbf{w}|}$$

Rewrite ($\mathbf{w}^T \mathbf{x}_{\perp} + b = 0$) :

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

Solve for γ: $$\gamma = \frac{\mathbf{w}^T \mathbf{x} + b}{|\mathbf{w}|} = \frac{\mathbf{w}^T}{|\mathbf{w}|} \mathbf{x} + \frac{b}{|\mathbf{w}|}$$

Geometric Margins

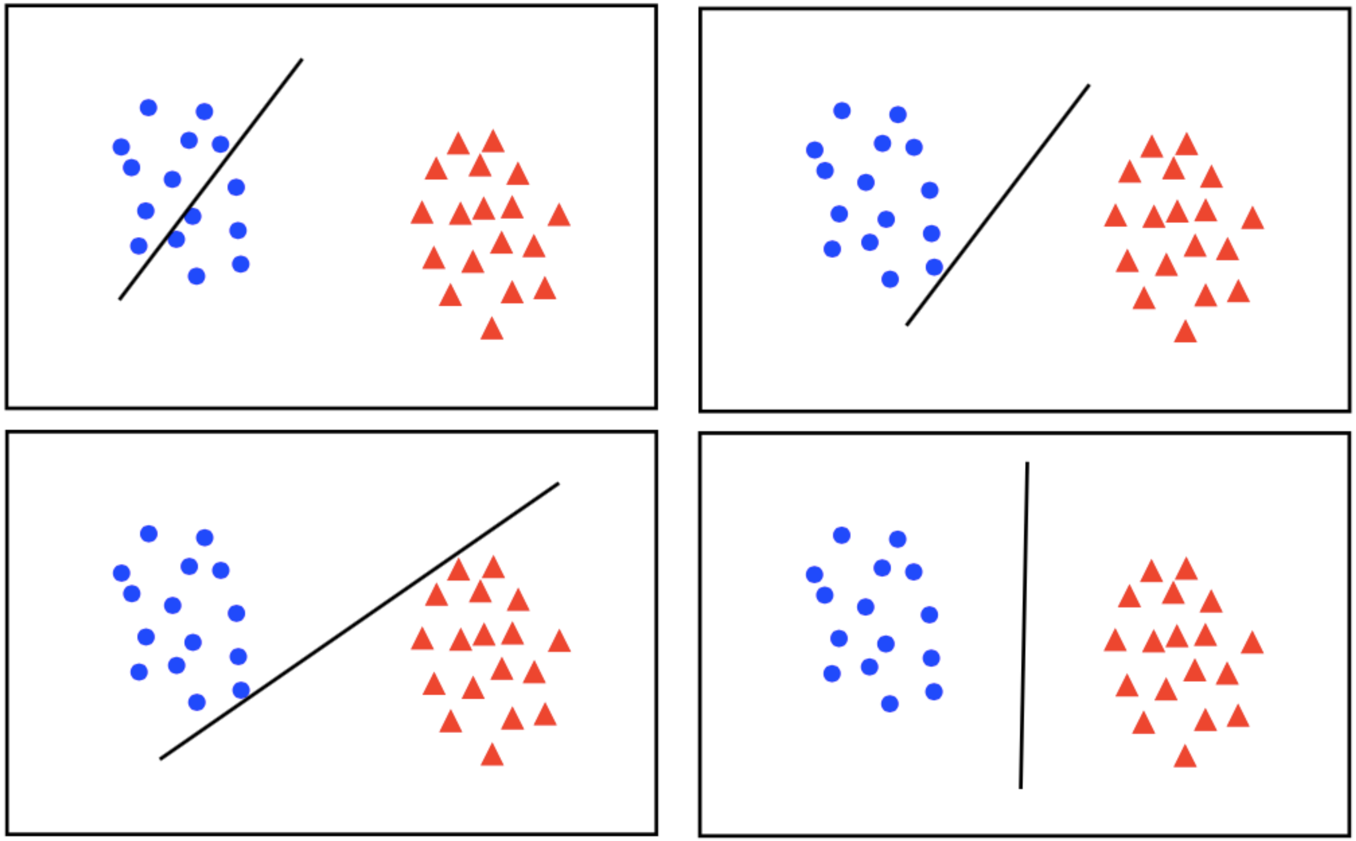


$$\gamma = \frac{\mathbf{w}^T\mathbf{x} + b}{|\mathbf{w}|}$$

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)

Which classifier is best?



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

How can we quantify this? (large margins!)

How can we enforce this?

Lecture outline

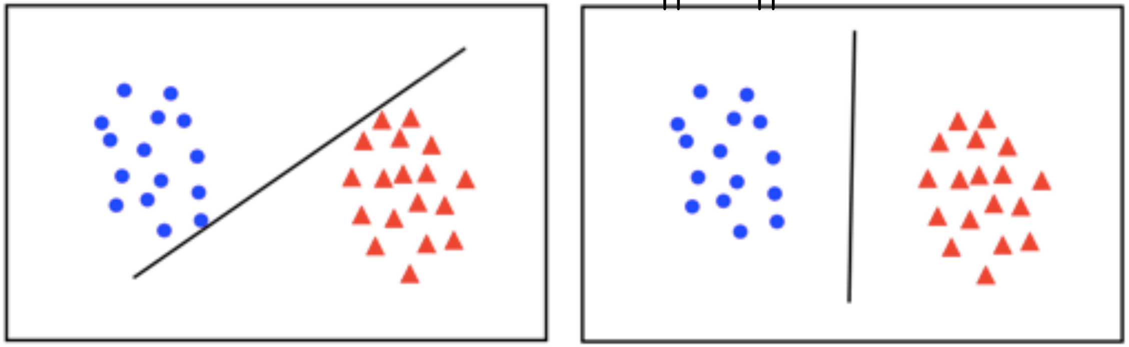
- Introduction to Support Vector Machines
  - Geometric margins
  - Training criterion & hinge loss
- Large margins and generalization
- Optimization
- Kernels
- Applications to vision

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: $\gamma^i = y^i \dfrac{\mathbf{w}^T \mathbf{x}^i + b}{\|\mathbf{w}\|}$



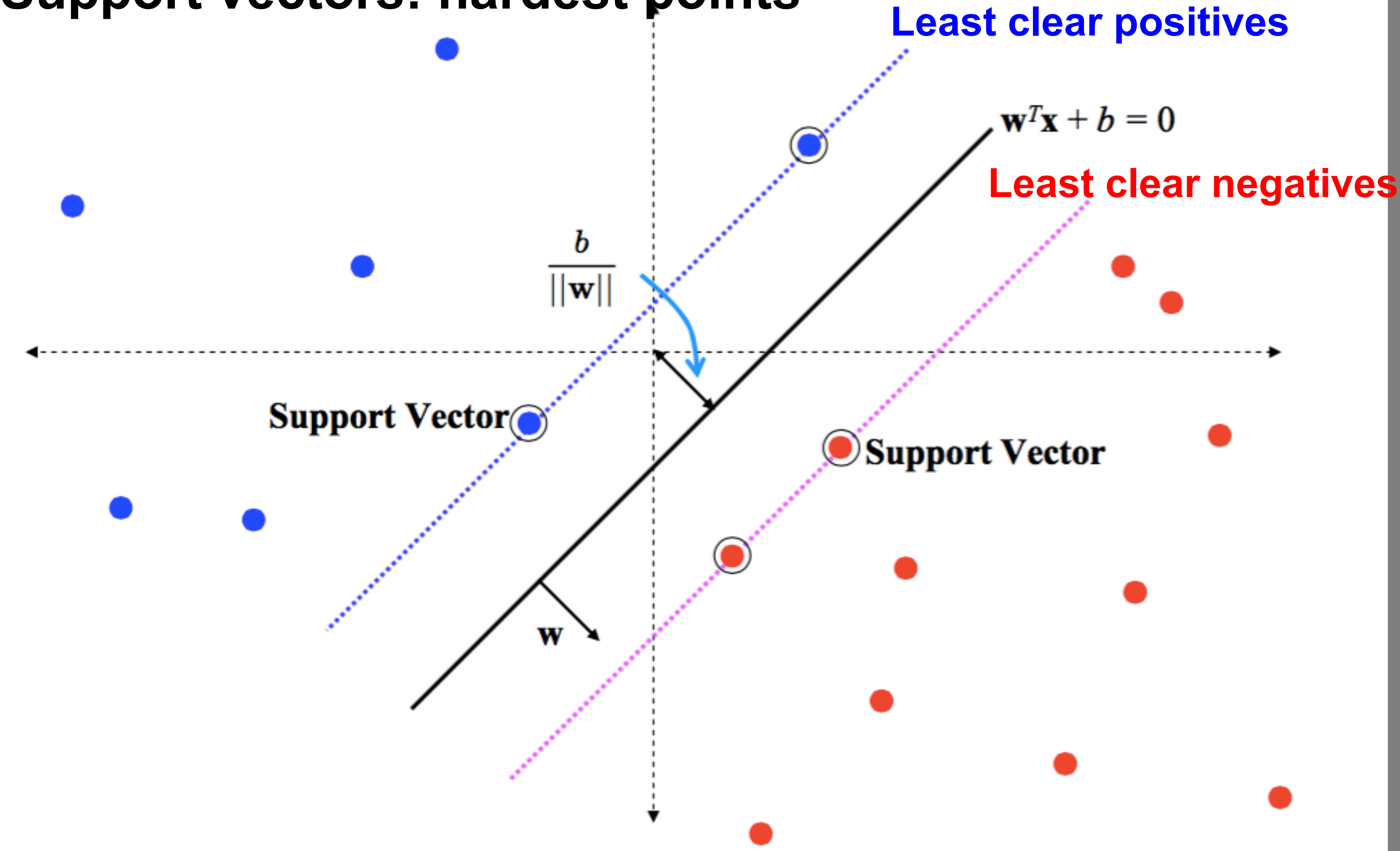
Should we be optimizing the mean, max, min margin?

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

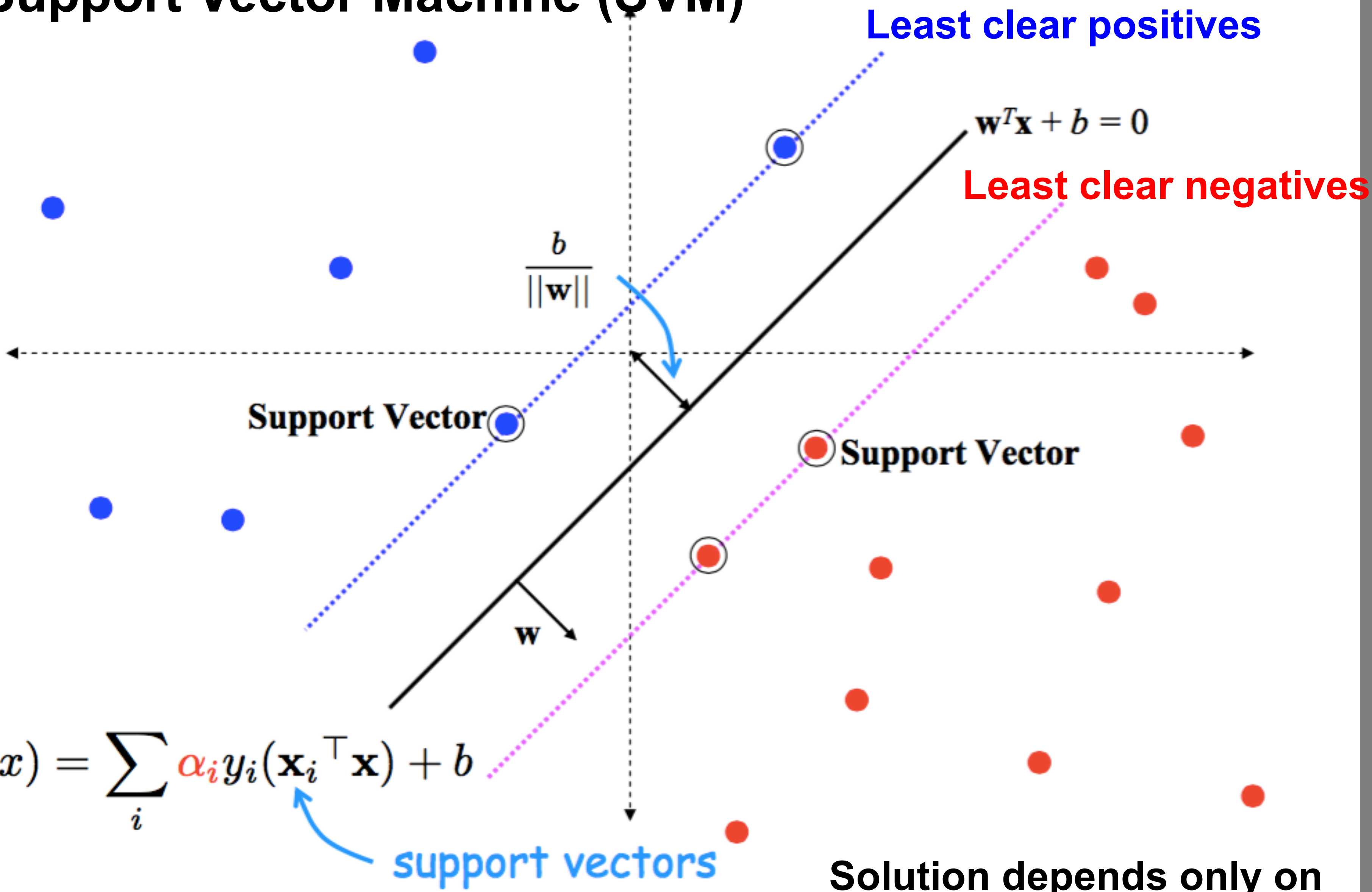
1) Take points that do not lie clearly on the correct side

2) Make sure they do

Support vectors: hardest points



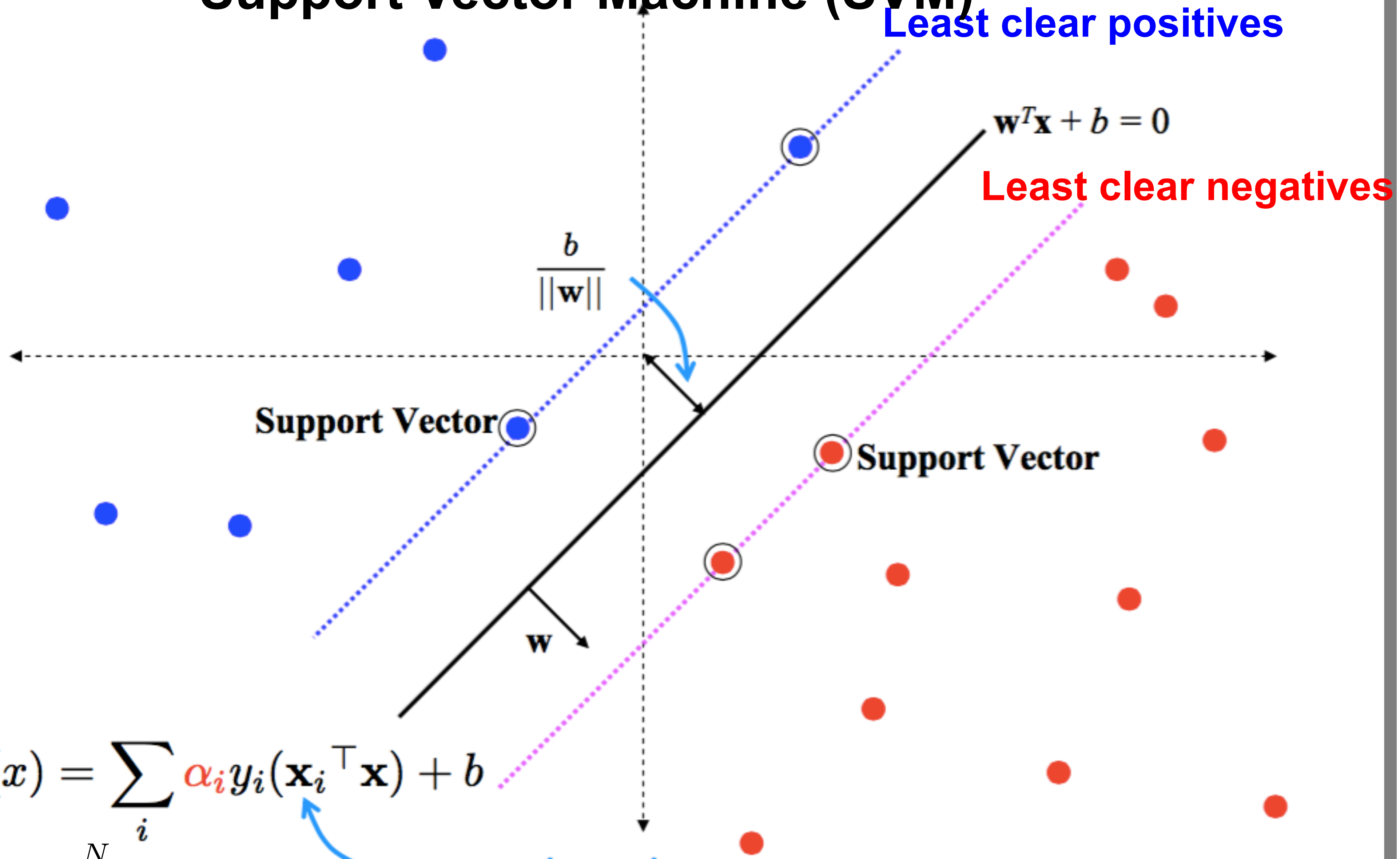
Support Vector Machine (SVM)



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

Solution depends only on hardest points

Support Vector Machine (SVM)



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

support vectors

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

Solution depends only on ‘Support Vectors’

SVM, sketch of derivation

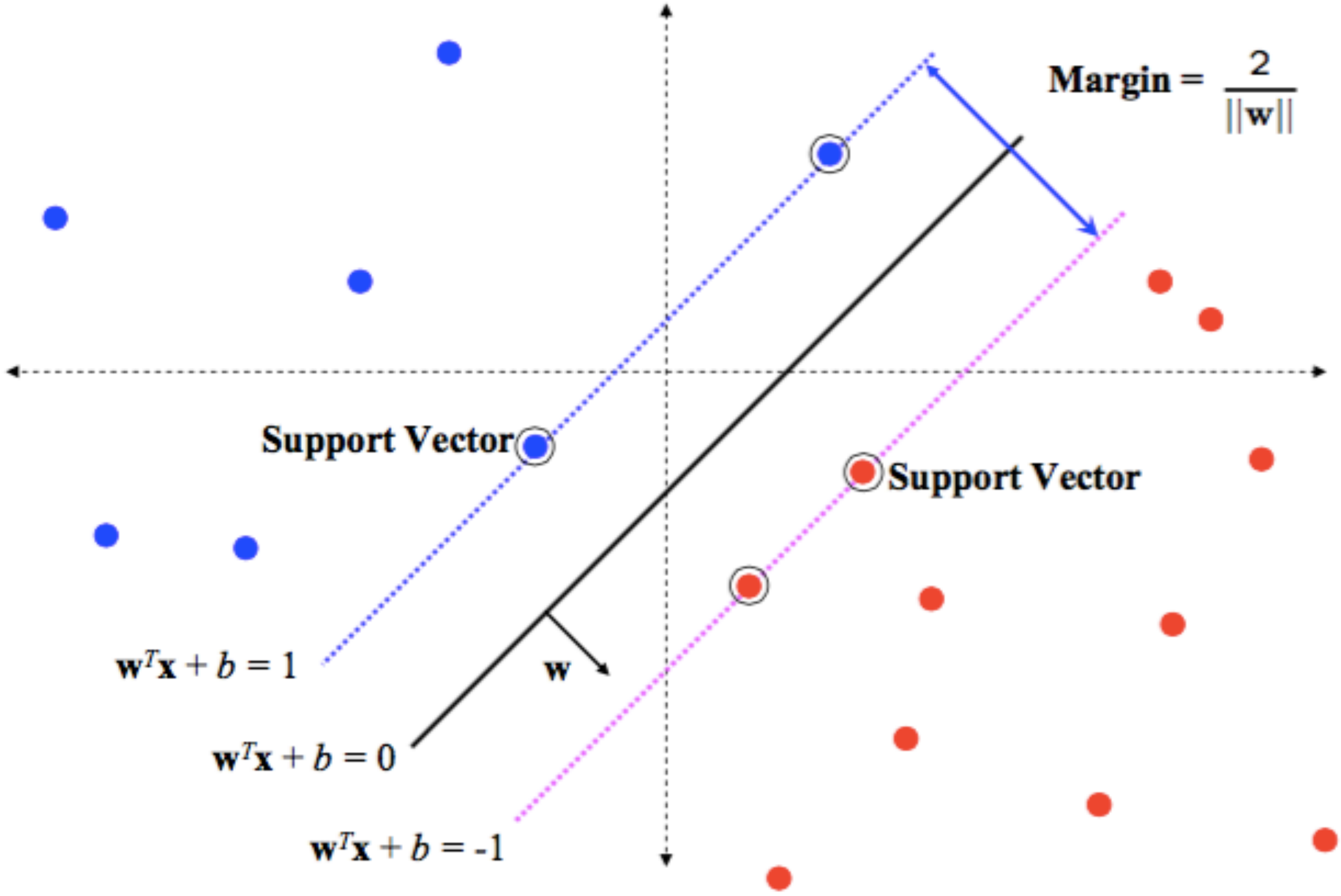
- Since $\mathbf{w}^\top \mathbf{x} + b = 0$ and $c(\mathbf{w}^\top \mathbf{x} + b) = 0$ define the same plane, we have the freedom to choose the normalization

- Choose normalization such that $\mathbf{w}^\top \mathbf{x}_+ + b = +1$ and $\mathbf{w}^\top \mathbf{x}_- + b = -1$ for the positive and negative support vectors respectively

- Then the margin is given by

$$\frac{\mathbf{w}^\top \left(\mathbf{x}_+ - \mathbf{x}_-\right)}{||\mathbf{w}||} = \frac{2}{||\mathbf{w}||}$$

Support Vector Machine (SVM)


Margin = $\frac{2}{||\mathbf{w}||}$
Support Vector
Support Vector
$\mathbf{w}^T\mathbf{x} + b = 1$
$\mathbf{w}$
$\mathbf{w}^T\mathbf{x} + b = 0$
$\mathbf{w}^T\mathbf{x} + b = -1$

Representer theorem

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

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

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

Equivalently:

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

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

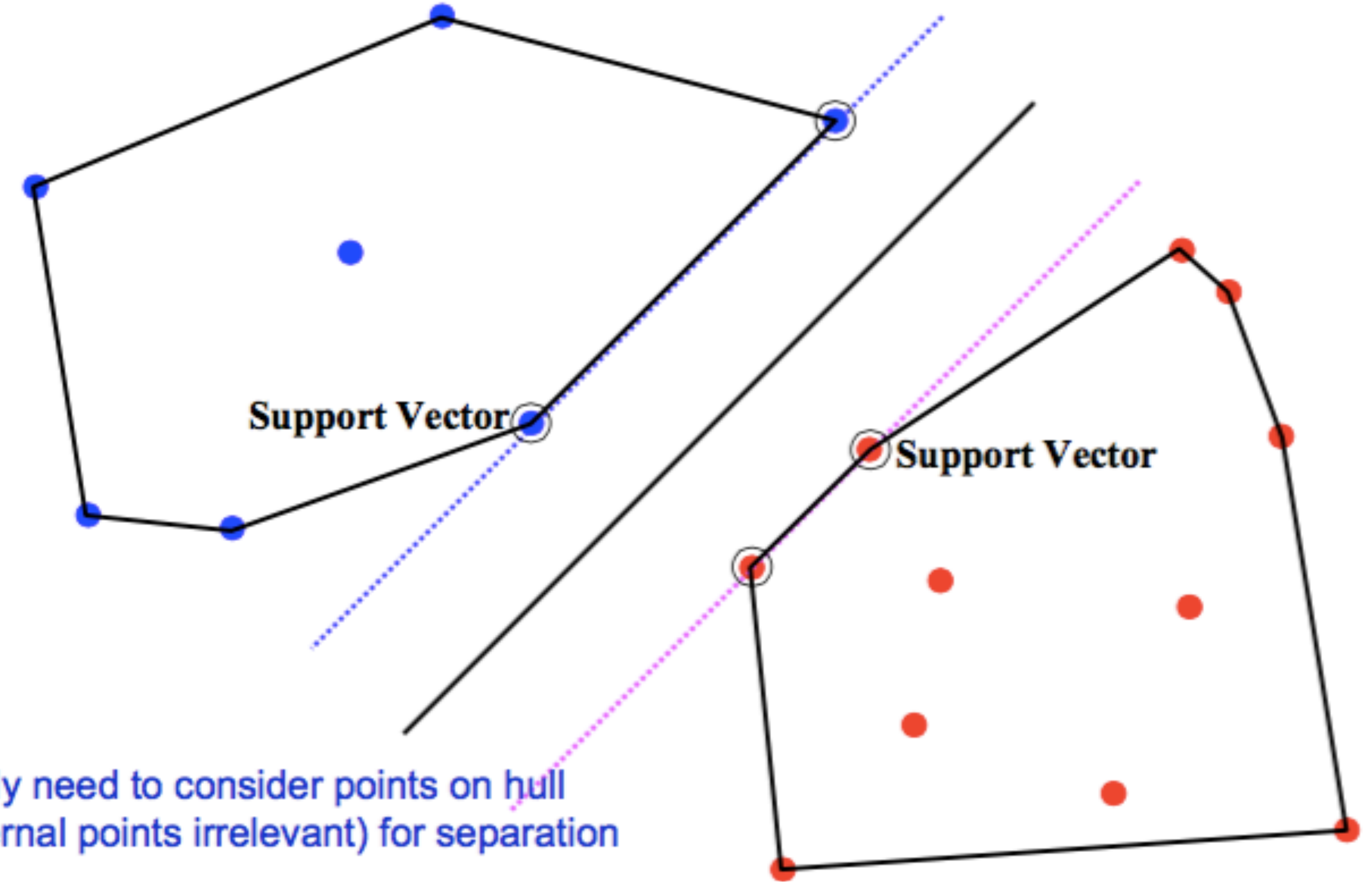
Representer Theorem: we can prove that the minimum is a linear combination of the training points

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

Geometric algorithm

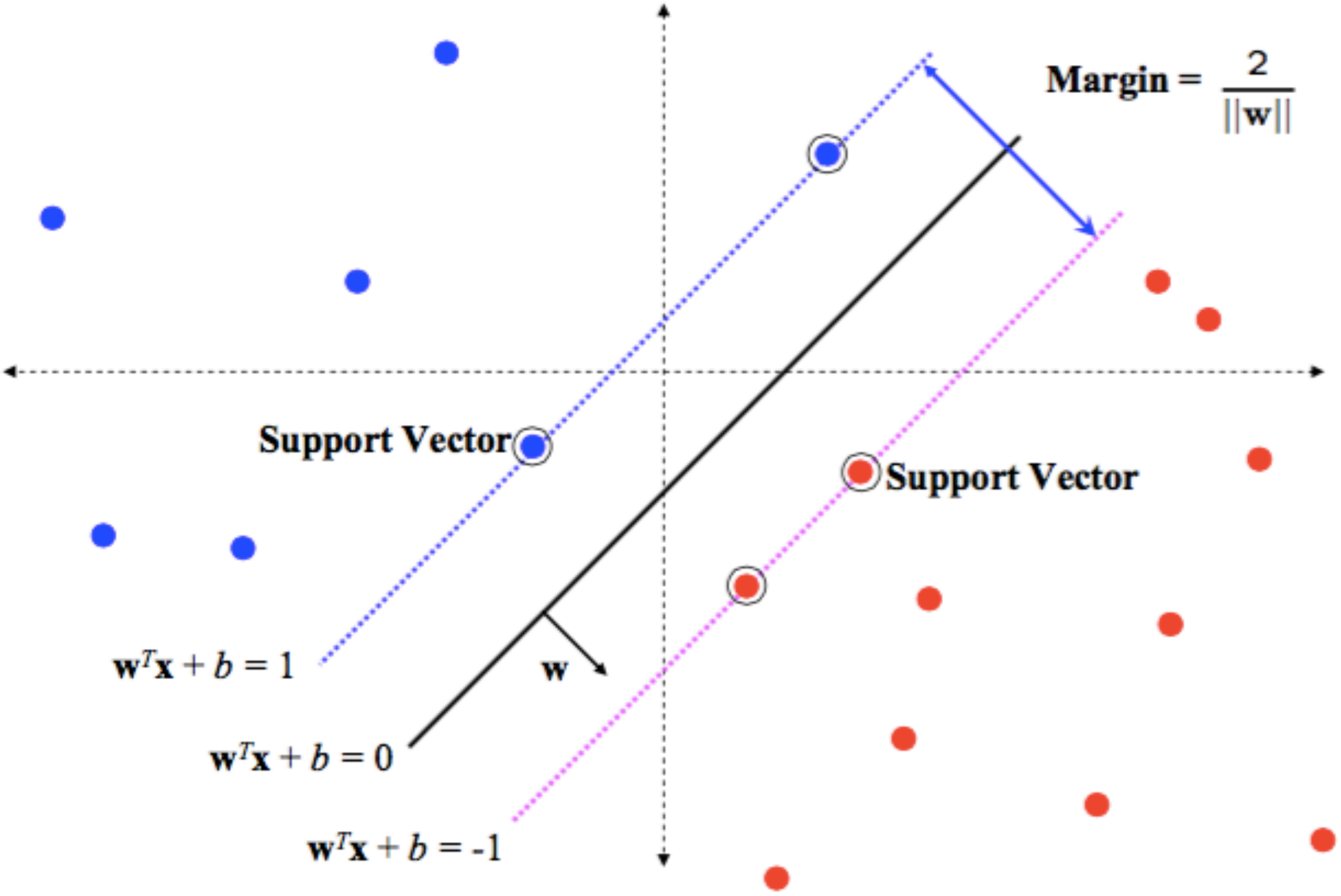
- Compute the convex hull of the positive points, and the convex hull of the negative points
- For each pair of points, one on positive hull and the other on the negative hull, compute the margin
- Choose the largest margin

Intuitive justification of theorem



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

Support Vector Machine (SVM)


Margin = $\frac{2}{||\mathbf{w}||}$
Support Vector
Support Vector
$\mathbf{w}^T\mathbf{x} + b = 1$
$\mathbf{w}$
$\mathbf{w}^T\mathbf{x} + b = 0$
$\mathbf{w}^T\mathbf{x} + b = -1$

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

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

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

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

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

Primal vs dual

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

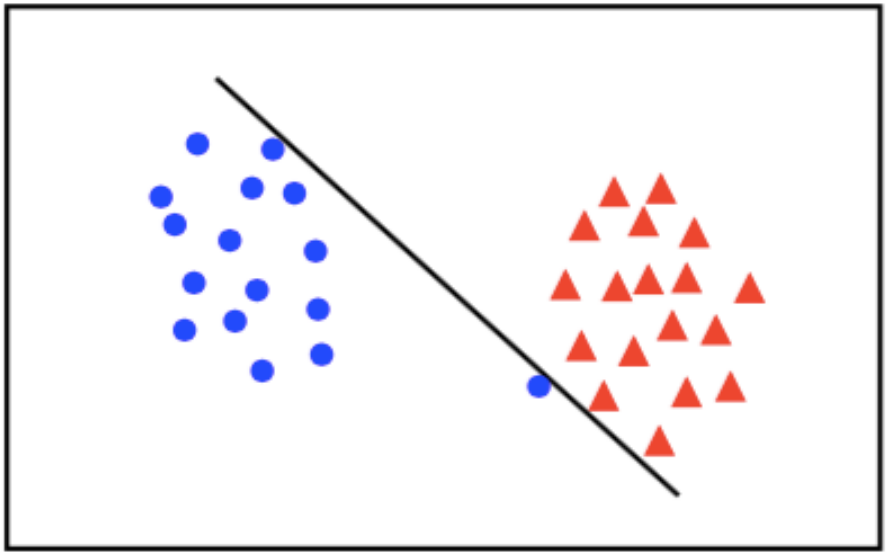
Dual:
$$\min_{\boldsymbol{\alpha}} \sum_{i=1}^{N}\sum_{j=1}^{N} \alpha^i\alpha^j y^i y^j \langle \mathbf{x}^i, \mathbf{x}^j \rangle \qquad \boldsymbol{\alpha} \in \mathbb{R}^N \to O(N^3)$$
$$\text{s.t.}: \;\; y^i\left(\sum_{j=1}^{N} \alpha^j y^j \langle \mathbf{x}^j, \mathbf{x}^i \rangle + b\right) \geq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

Dual can be faster if N<D!

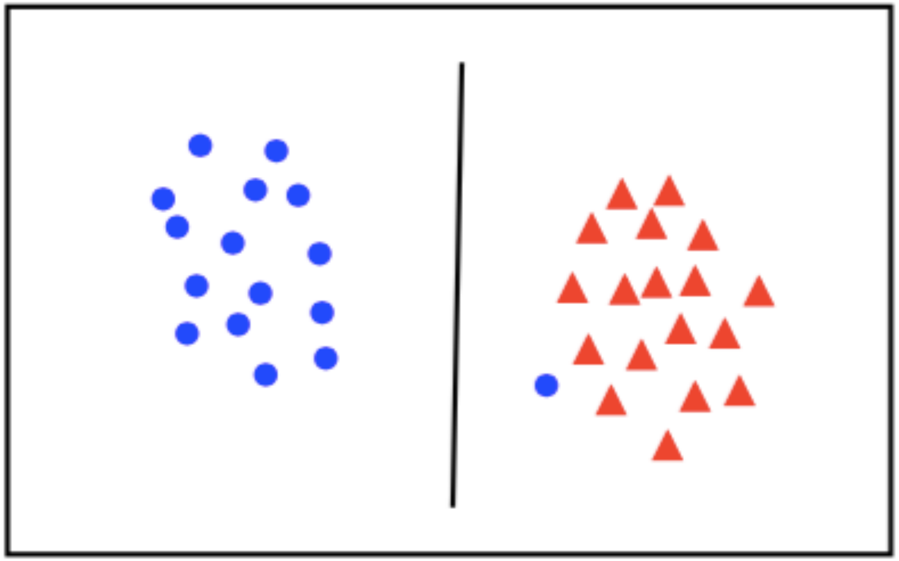
Primal and dual classifier forms:
$$f(\mathbf{x}) = \langle \mathbf{w}, \mathbf{x} \rangle + b = \sum_{i=1}^{N} \alpha^i y^i \langle \mathbf{x}^i, \mathbf{x} \rangle + b$$

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

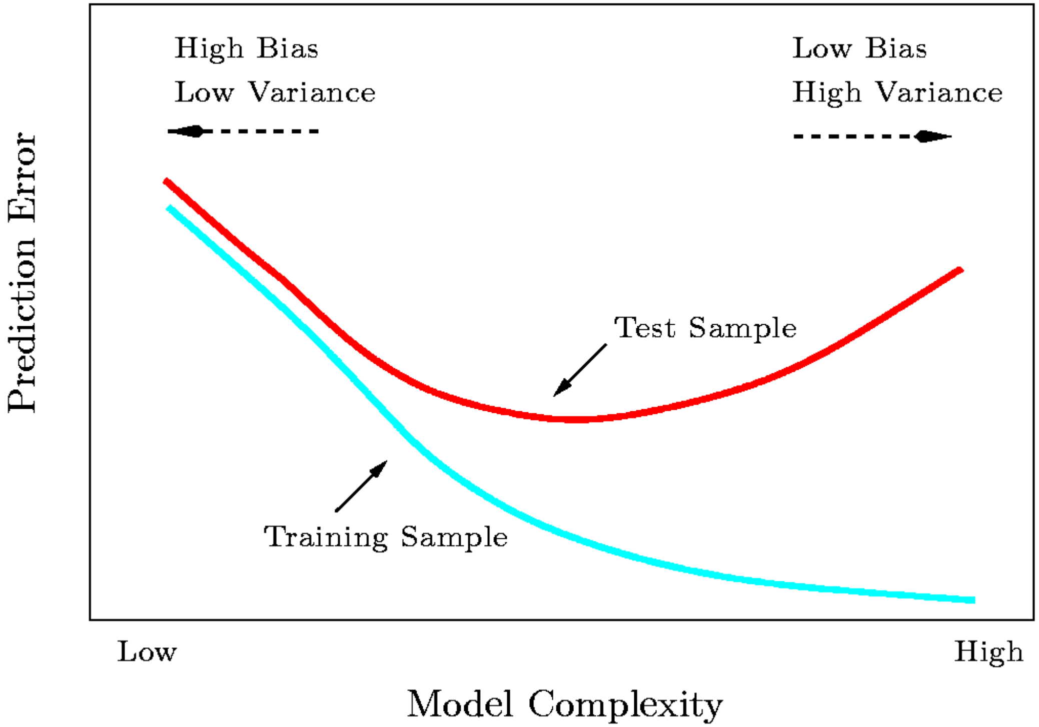
Tuning the model's complexity

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

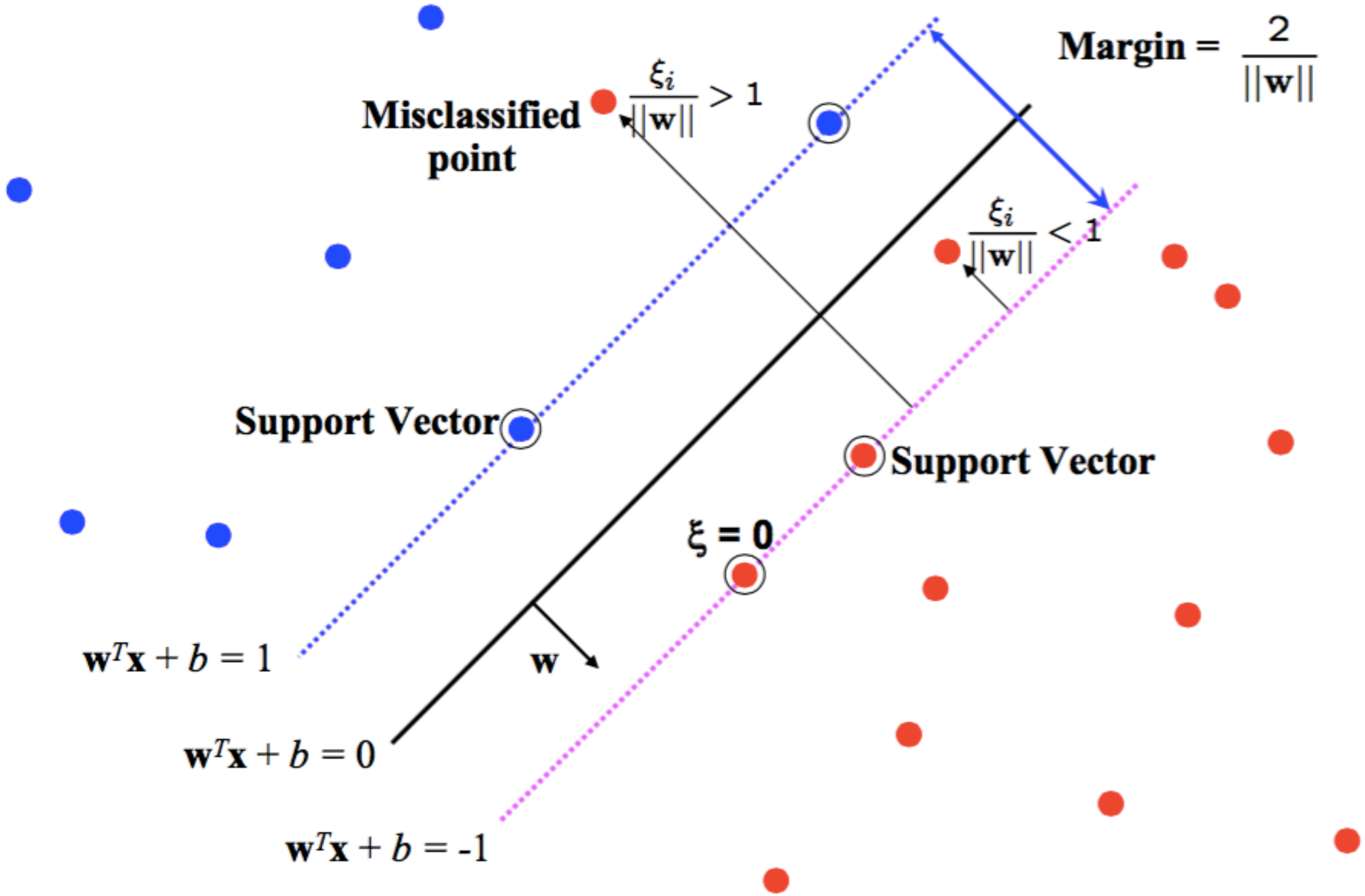
but can "overtrain" and have poor performance on the test set ("variance")

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


Margin = $\frac{2}{||\mathbf{w}||}$
Misclassified point
$\frac{\xi_i}{||\mathbf{w}||} > 1$
$\frac{\xi_i}{||\mathbf{w}||} < 1$
Support Vector
Support Vector
ξ = 0
$\mathbf{w}$
$\mathbf{w}^T\mathbf{x} + b = 1$
$\mathbf{w}^T\mathbf{x} + b = 0$
$\mathbf{w}^T\mathbf{x} + b = \text{-}1$

Objective for non-separable data

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

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

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

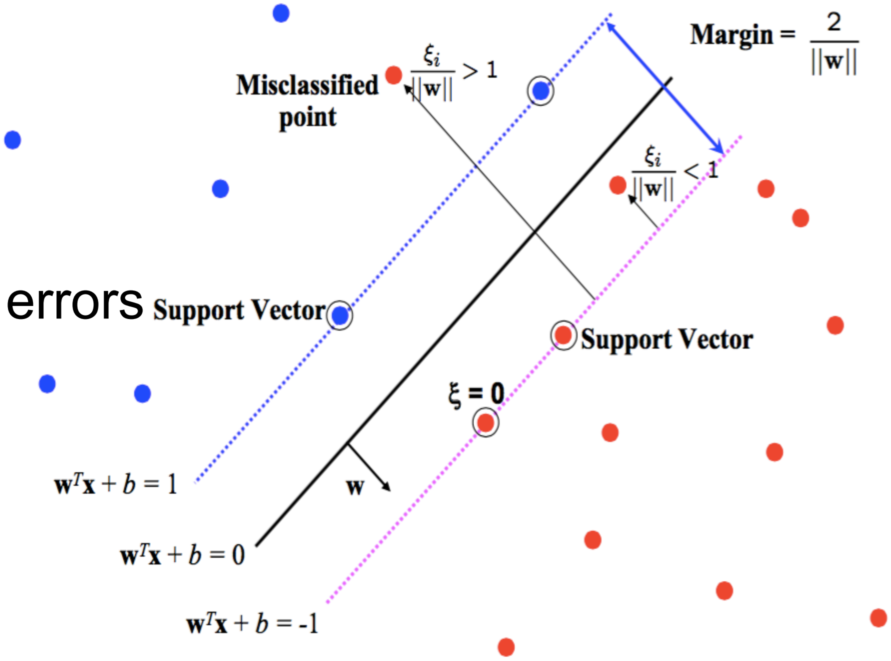
newcomers

misclassification when ξ>1

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

C: hyperparameter

(cross-validation!)



Margin = $\frac{2}{||\mathbf{w}||}$
Misclassified point
$\frac{\xi_i}{||\mathbf{w}||} > 1$
$\frac{\xi_i}{||\mathbf{w}||} < 1$
Support Vector
Support Vector
ξ = 0
$\mathbf{w}^T\mathbf{x} + b = 1$
$\mathbf{w}$
$\mathbf{w}^T\mathbf{x} + b = 0$
$\mathbf{w}^T\mathbf{x} + b = -1$

Primal problem: for $\mathbf{w} \in \mathbb{R}^d$

$$\min_{\mathbf{w}\in\mathbb{R}^d, \xi_i \in \mathbb{R}^+} ||\mathbf{w}||^2 + C\sum_i^N \xi_i \text{ subject to } y_i\left(\mathbf{w}^\top \mathbf{x}_i + b\right) \geq 1 - \xi_i \text{ for } i = 1 \dots N$$

The constraint $y_i\left(\mathbf{w}^\top \mathbf{x}_i + b\right) \geq 1 - \xi_i$, can be written more concisely as

$$y_i f(\mathbf{x}_i) \geq 1 - \xi_i$$

which is equivalent to

$$\xi_i = [1 - y_i f(\mathbf{x}_i)]_+$$

where $[.]_+$ indicates the positive part. Hence the optimization problem is equivalent to

$$\min_{\mathbf{w}\in\mathbb{R}^d} \underbrace{||\mathbf{w}||^2}_{\text{regularization}} + C\sum_i^N \underbrace{[1 - y_i f(\mathbf{x}_i)]_+}_{\text{loss function}}$$

Loss function

Optimization problem:

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

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

$$\xi^i \geq 0$$

Rewrite constraint: $y^i h_{\mathbf{w},b}(\mathbf{x}) \geq 1 - \xi^i$

Compact form: $\xi^i = [1 - y^i h_{\mathbf{w},b}(\mathbf{x})]_+$

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

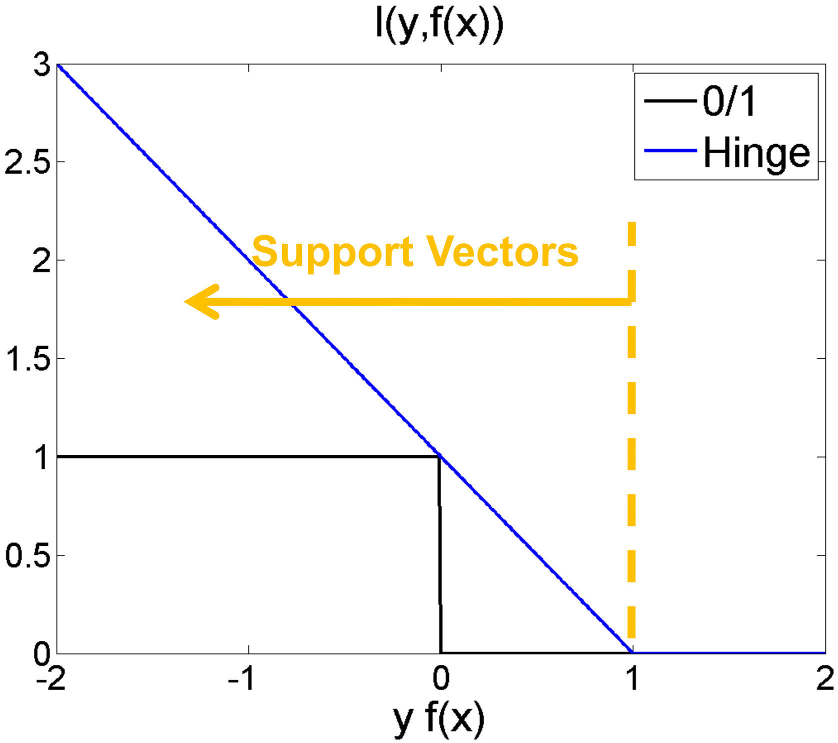
What if we plug that in the optimization objective?

Loss function

Optimization problem:
$$
\begin{aligned}
L(\mathbf{w}) &= \frac{1}{2}||\mathbf{w}||^2 + C\sum_{i=1}^{N} \max(0, 1 - y^i h_{\mathbf{w},b}(x^i)) \\
&\propto \lambda||\mathbf{w}||^2 + \sum_{i=1}^{N} \underbrace{\max(0, 1 - y^i h_{\mathbf{w},b}(x^i))}_{l(y^i,x^i)}
\end{aligned}
$$

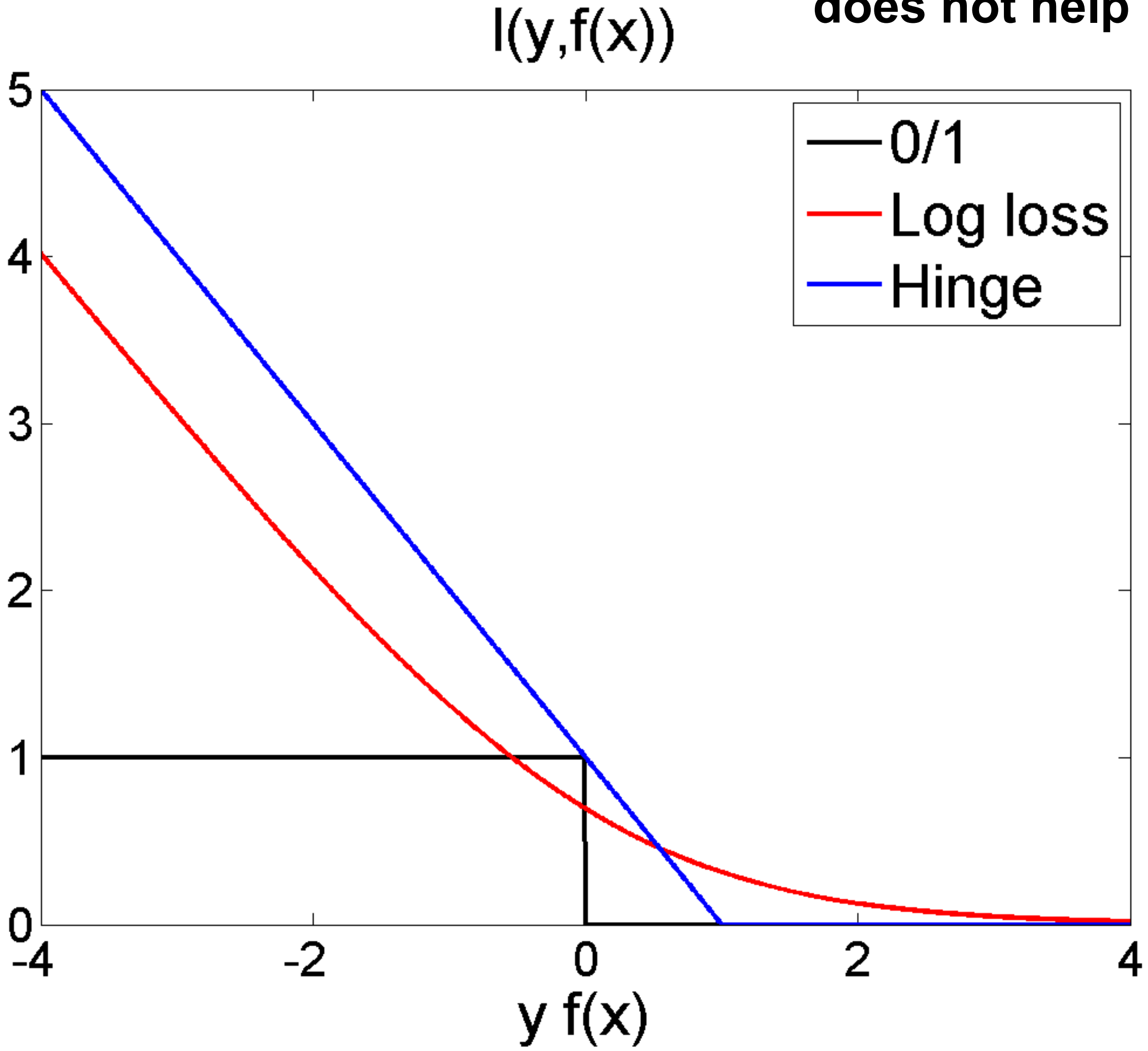
regularizer **additive loss**

Hinge loss:

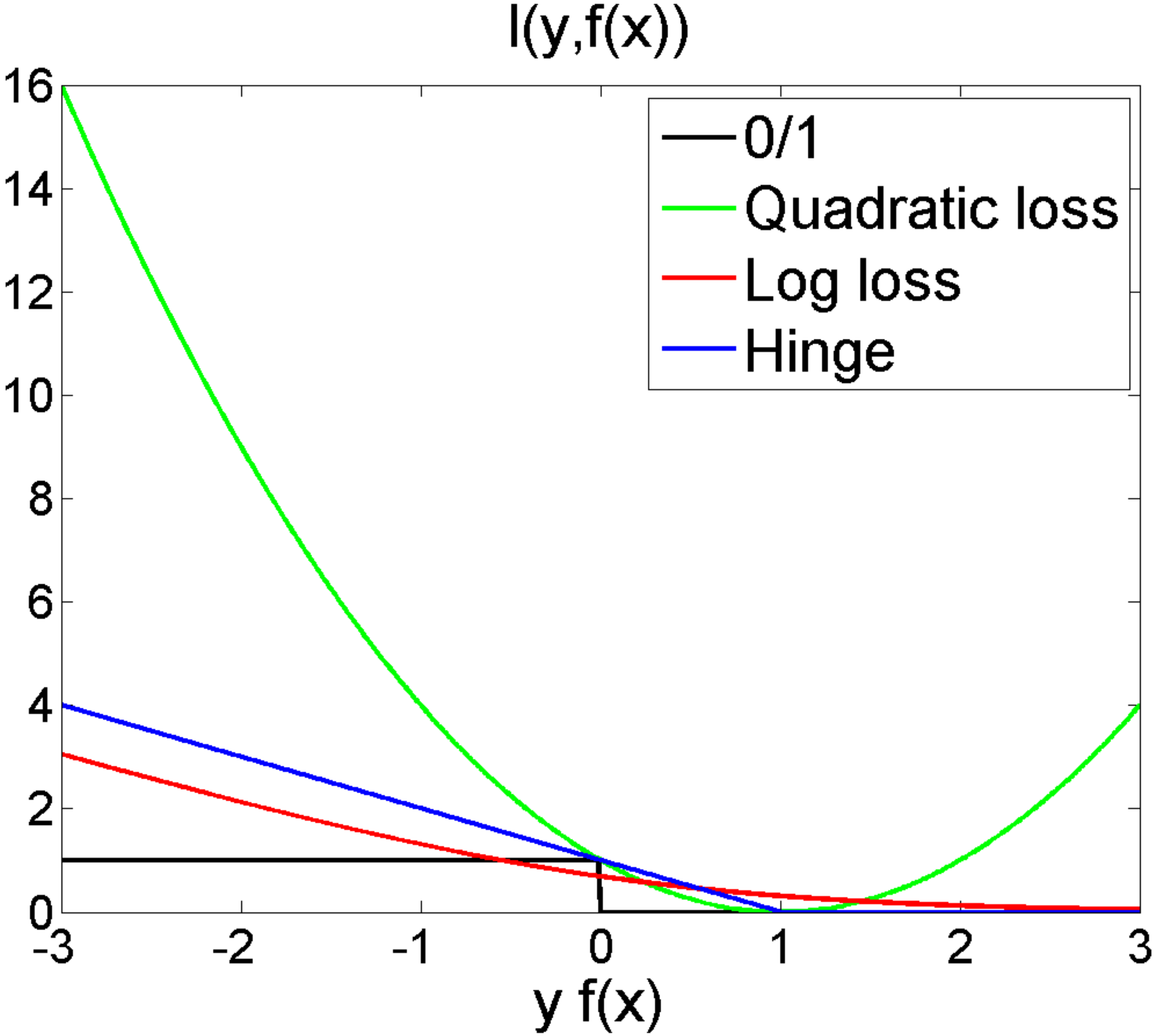


Hinge loss vs log-loss

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

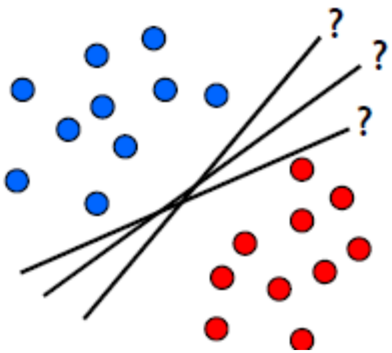
Kernels

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

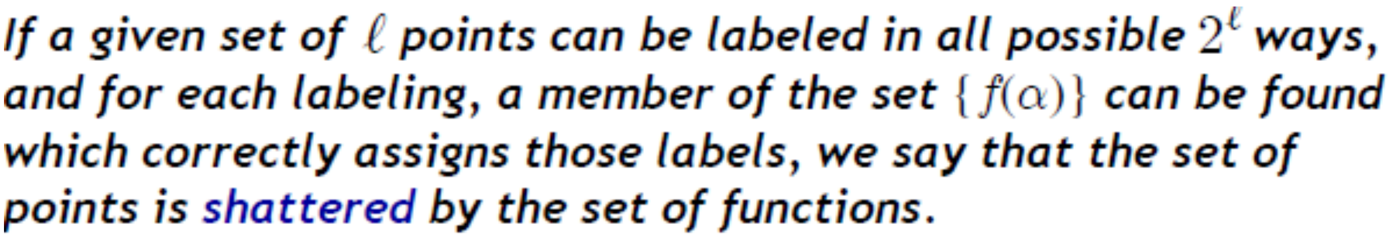
N: number of training examples

p^*: probability that the bound is correct

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

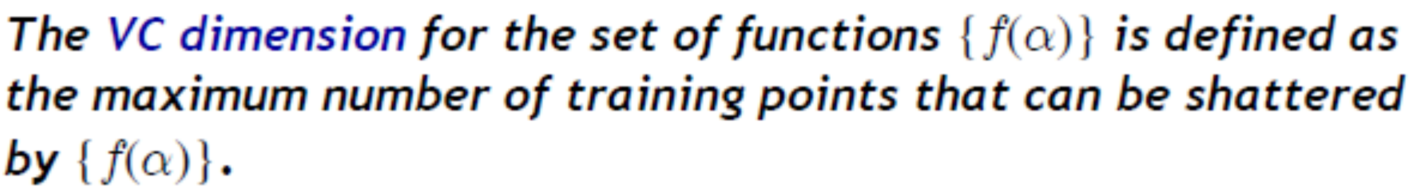
Vapnik Chervonenkis (VC) Dimension

- Shattering:



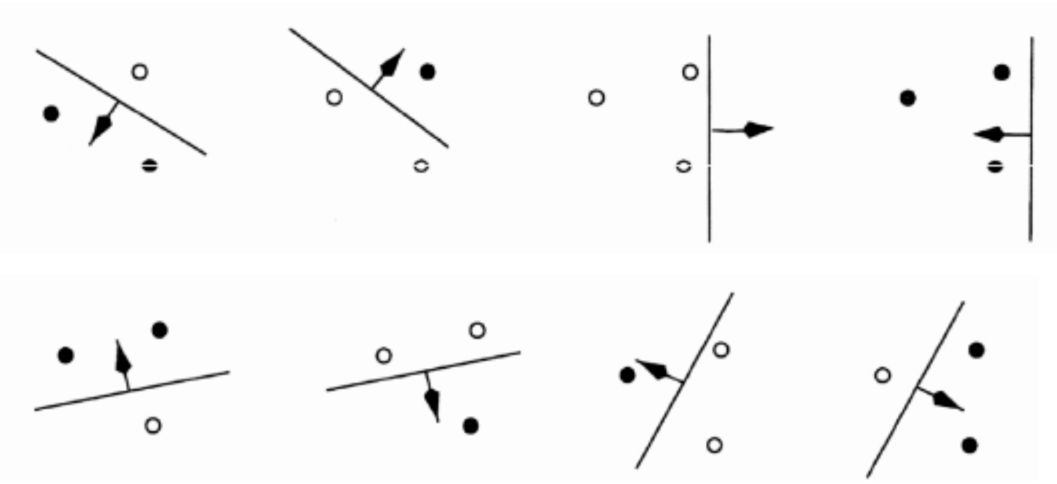
If a given set of ℓ points can be labeled in all possible 2^ℓ ways, and for each labeling, a member of the set $\{f(\alpha)\}$ can be found which correctly assigns those labels, we say that the set of points is shattered by the set of functions.

- VC dimension

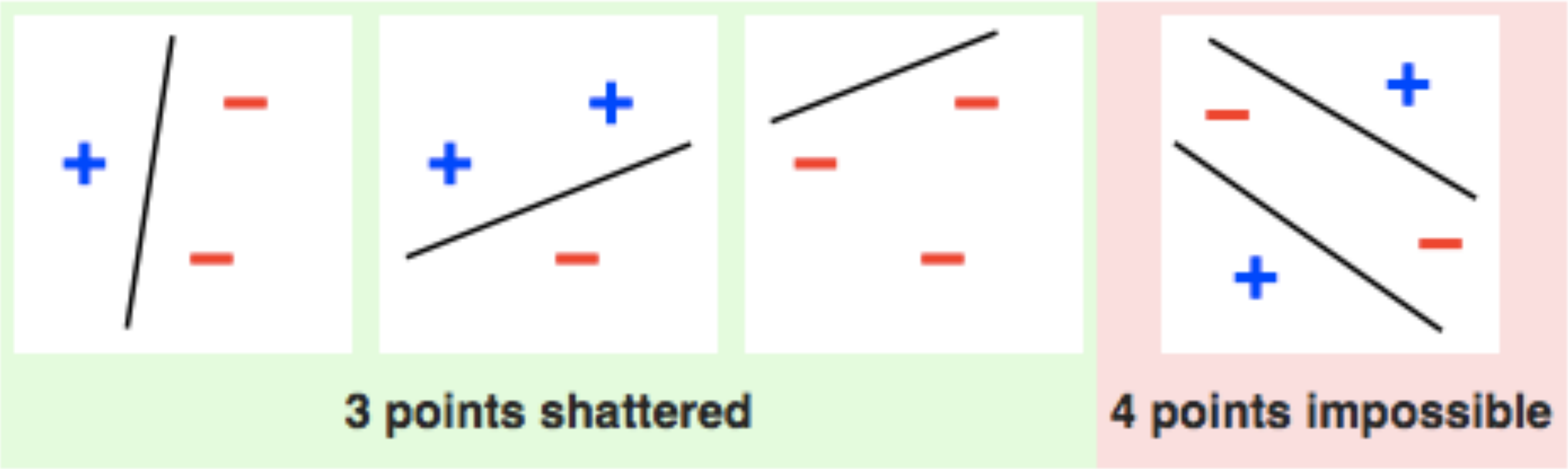


The VC dimension for the set of functions $\{f(\alpha)\}$ is defined as the maximum number of training points that can be shattered by $\{f(\alpha)\}$.

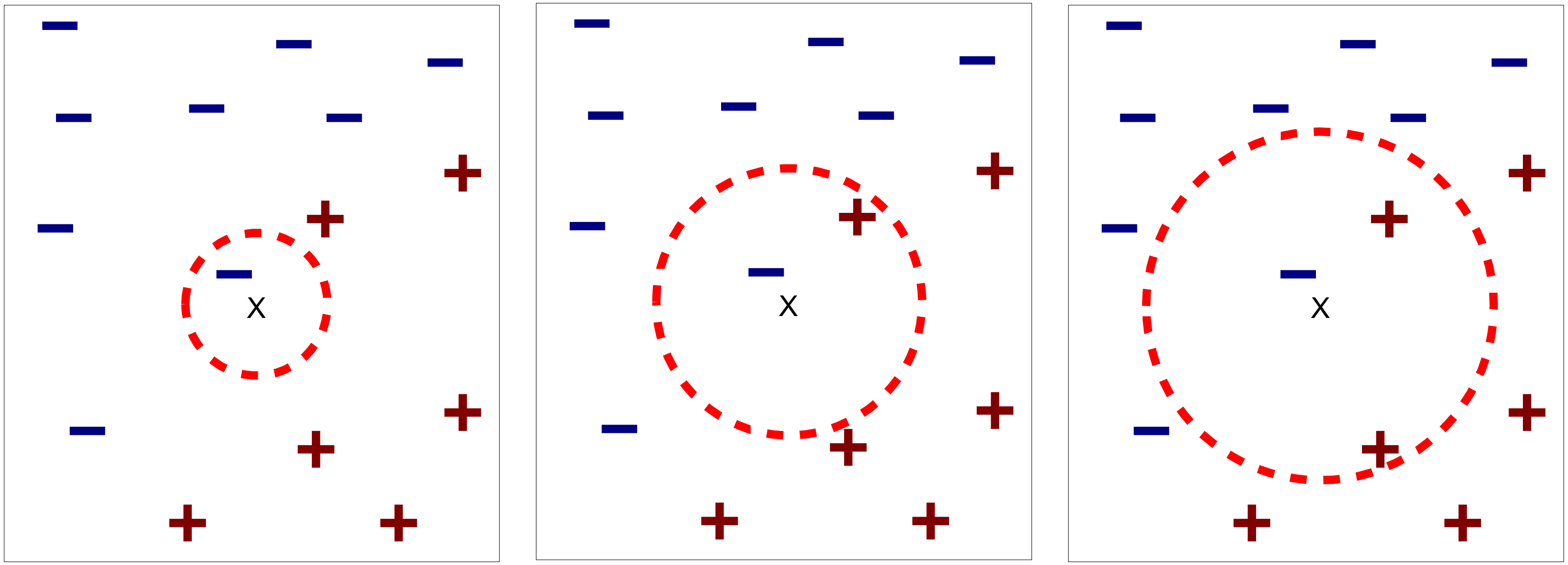
- Example



Arbitrary linear classifier in N-dimensions: VC-dim= N+1



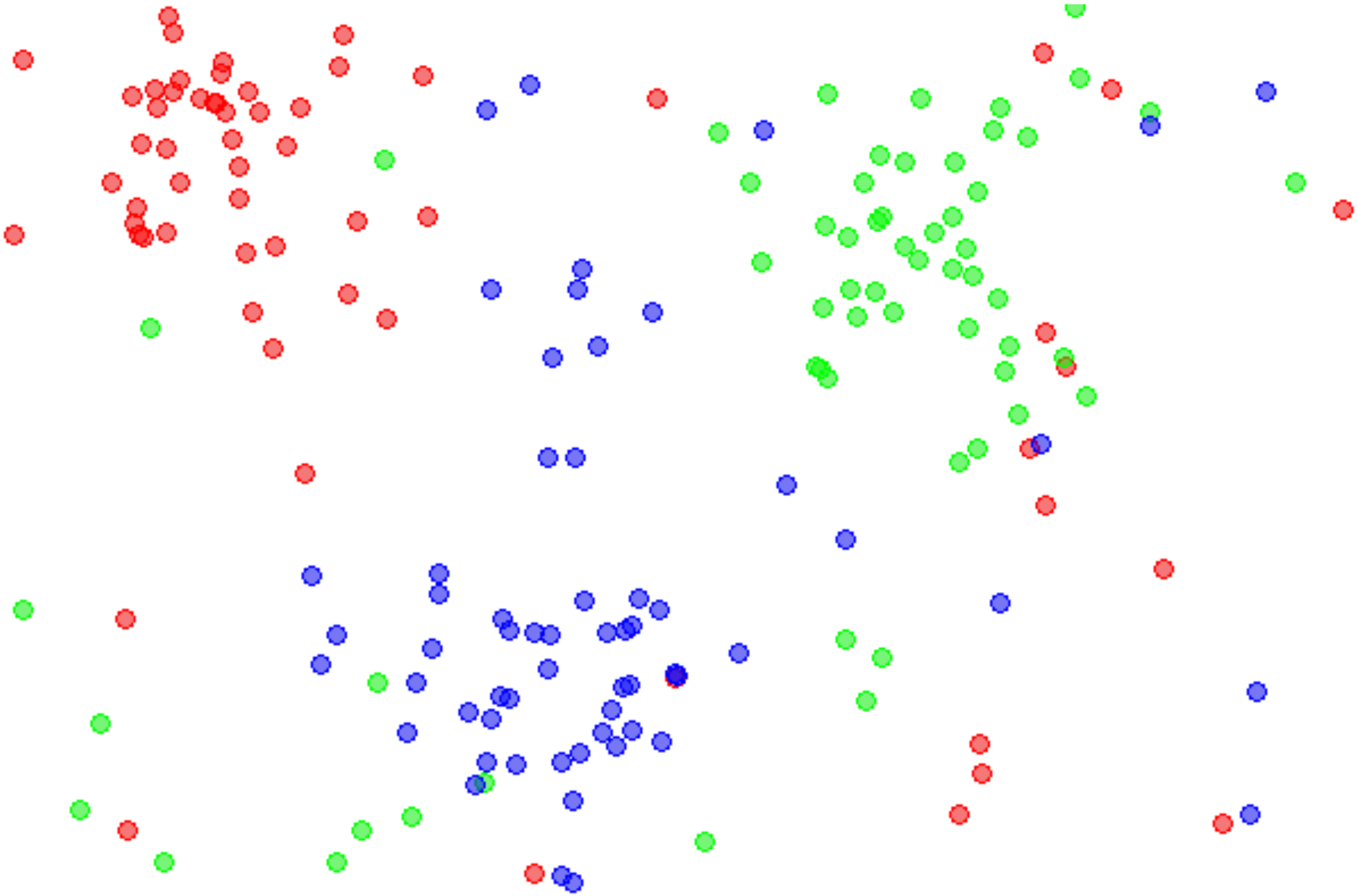
Reminder: K-nearest neighbor classifier



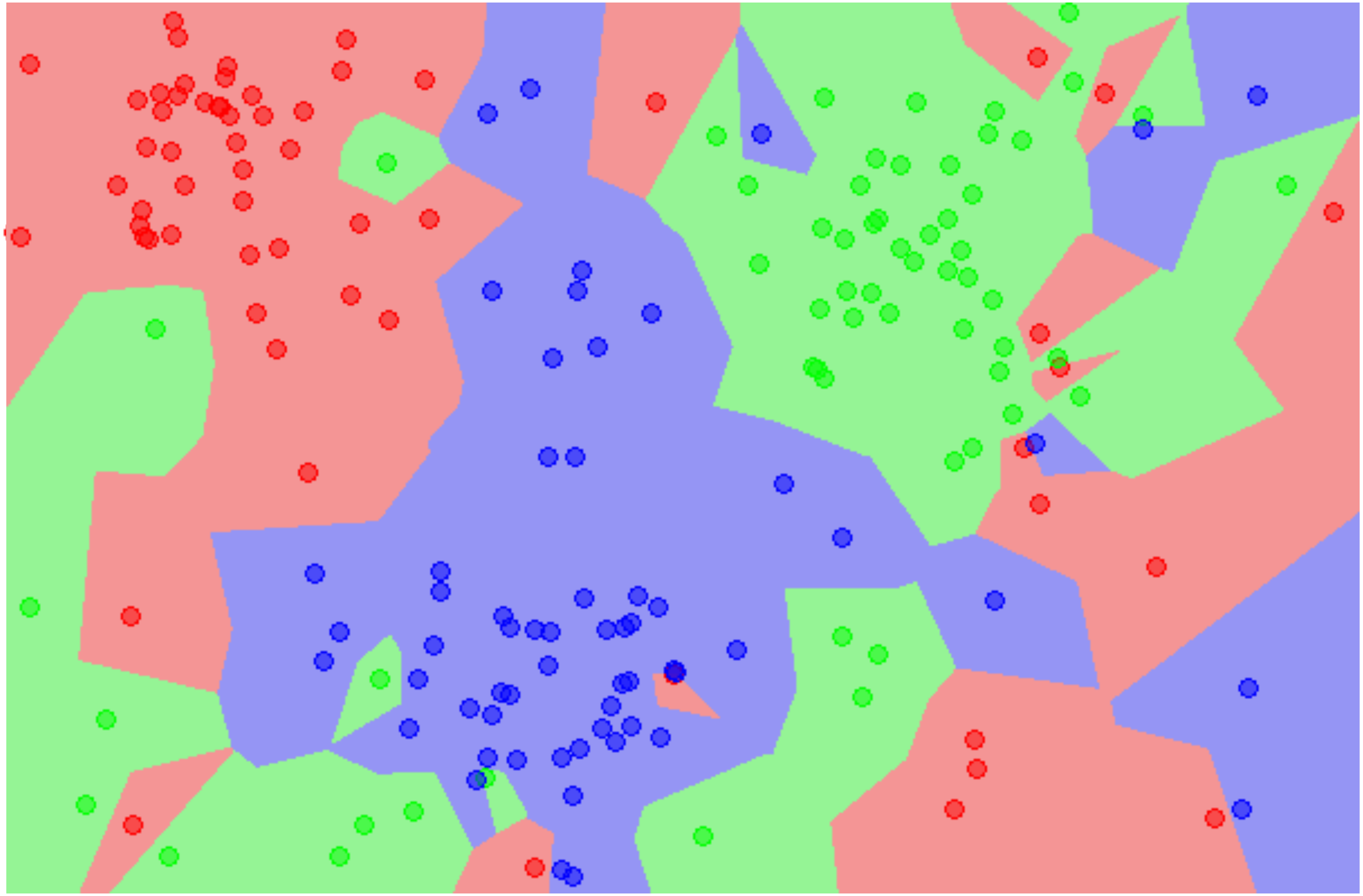
(a) 1-nearest neighbor (b) 2-nearest neighbor (c) 3-nearest neighbor

- Compute distance to other training records
- Identify *K* nearest neighbors
- Take majority vote

Training data for NN classifier (in R^2)



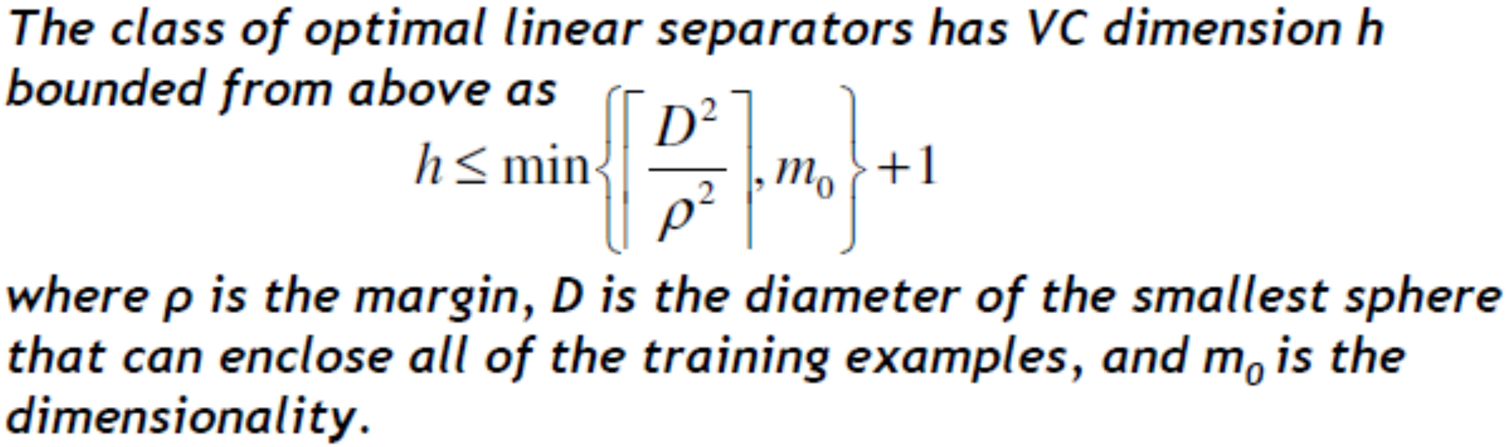
1-nn classifier prediction (in R^2)



What is the VC dimension of this classifier?

Large Margins & VC Dimension

- Vapnik:



The class of optimal linear separators has VC dimension h bounded from above as

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

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

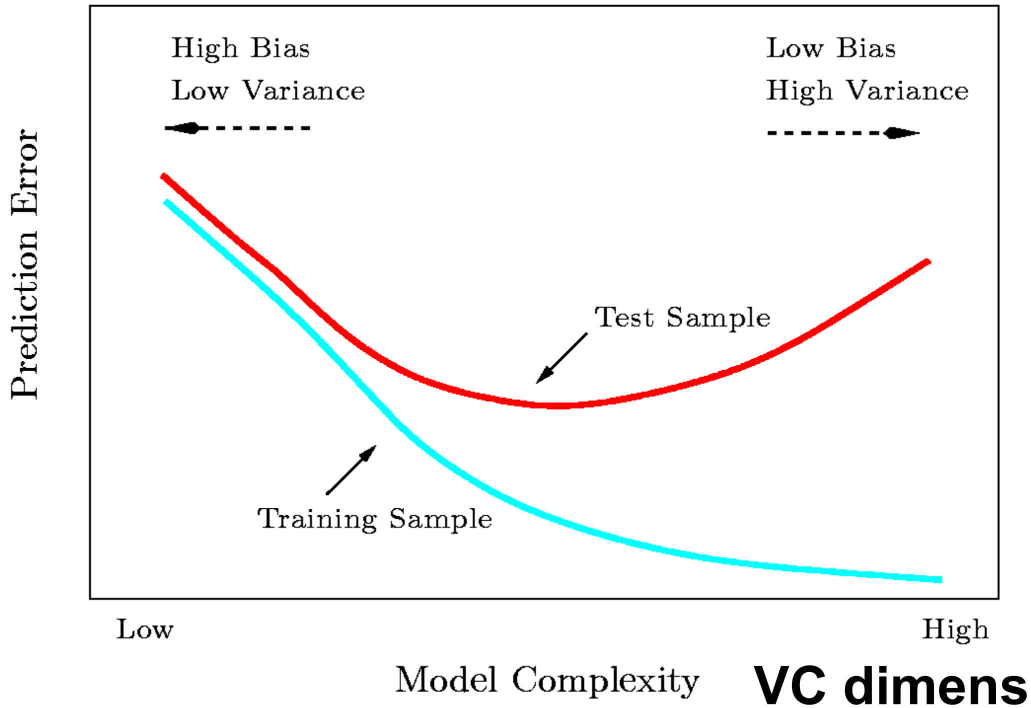
Tuning the model’s complexity

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

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

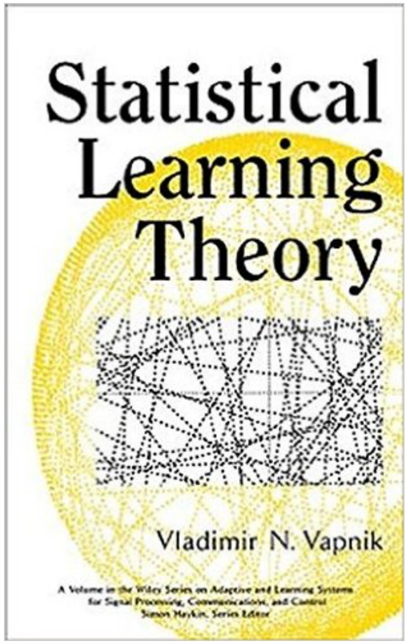
- With probability $(1\text{-}\eta)$, the following bound holds

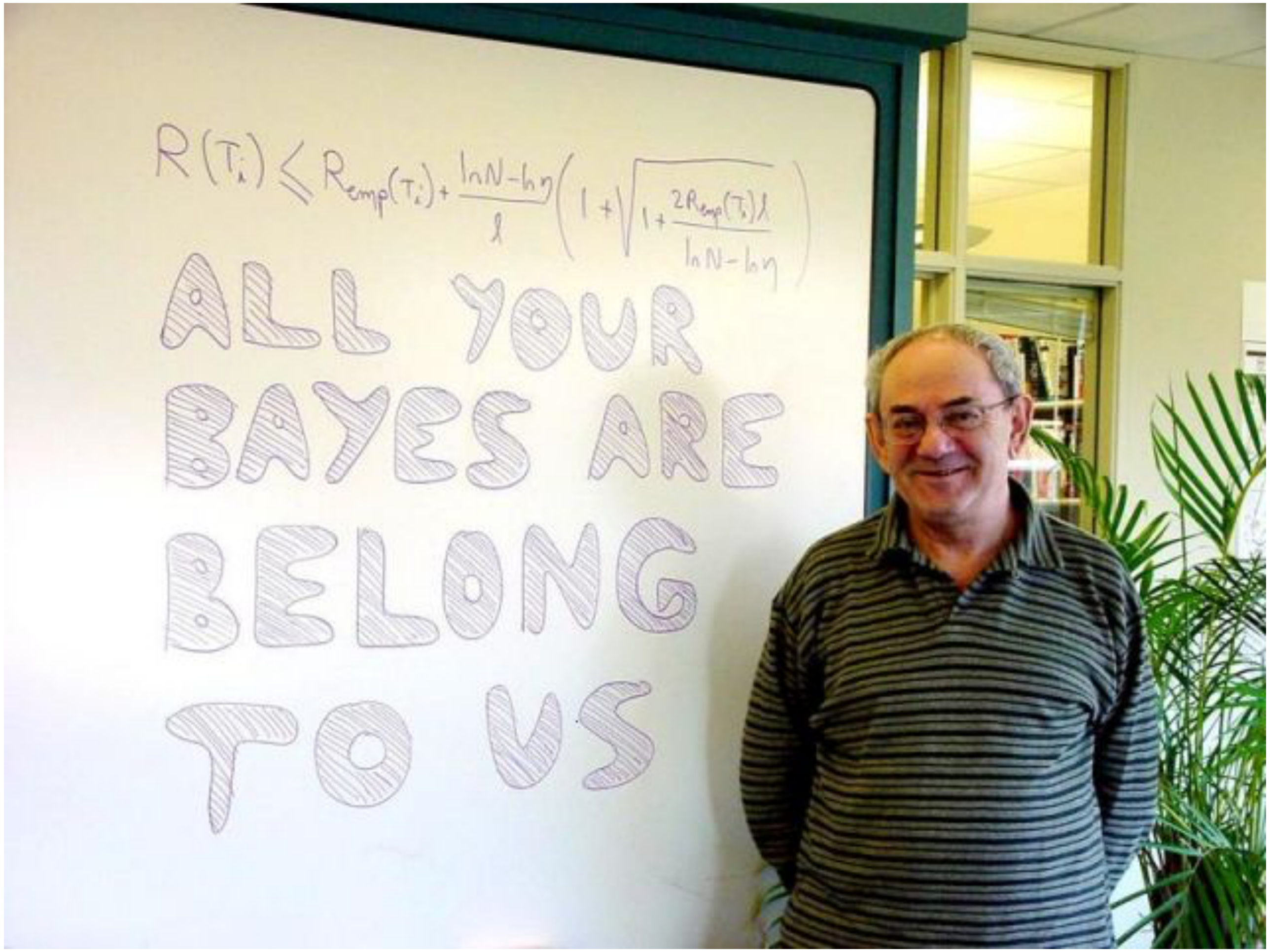
$$R(\alpha) \leq R_{emp}(\alpha) + \underbrace{\sqrt{\frac{h(\log(2N/h)+1) - \log(\eta/4)}{N}}}_{\text{“VC confidence”}}$$



VC dimension

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

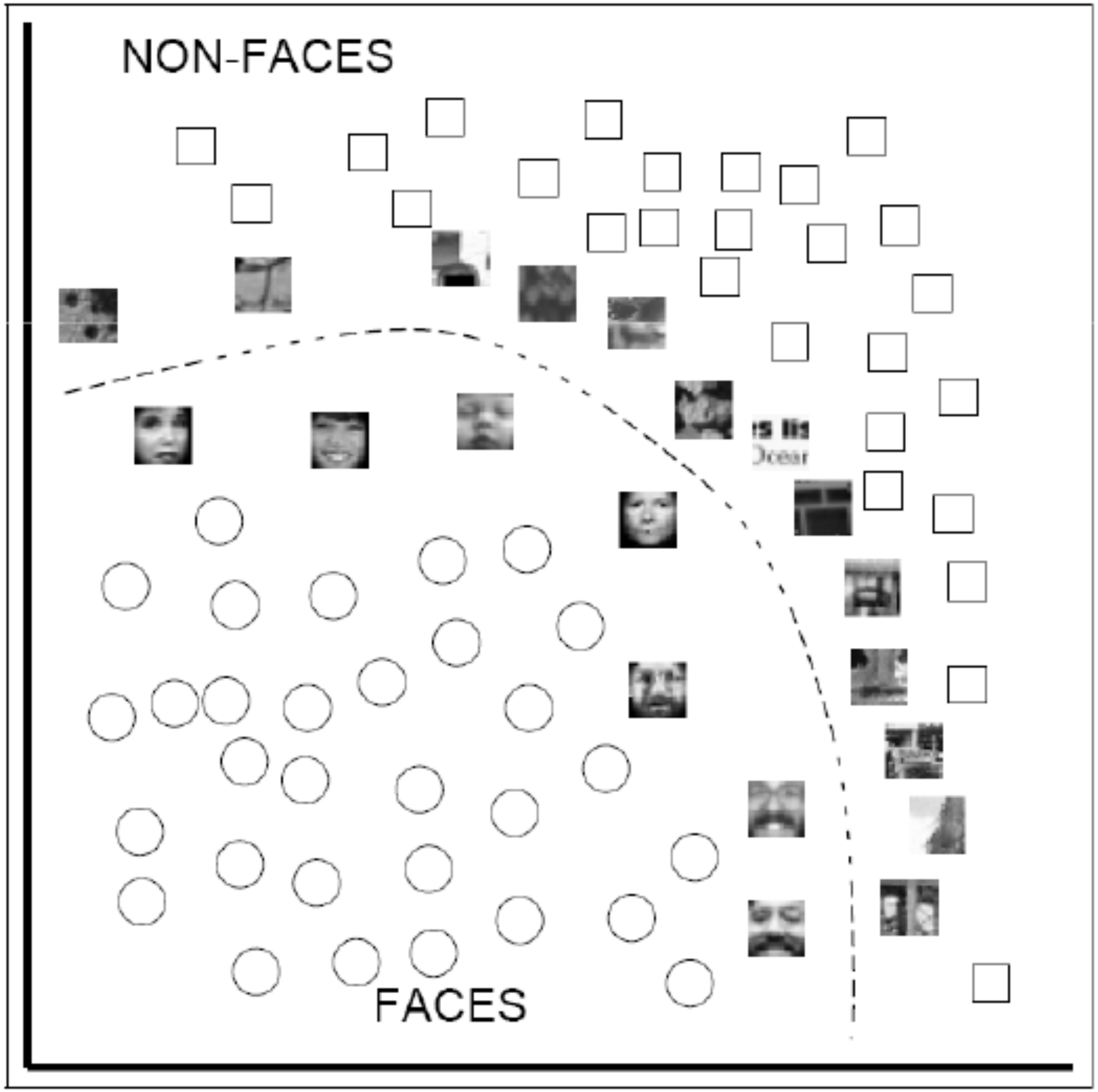

Statistical Learning Theory
Vladimir N. Vapnik
A Volume in the Wiley Series on Adaptive and Learning Systems for Signal Processing, Communications, and Control
Simon Haykin, Series Editor


$R(T_i) \leqslant R_{emp}(T_i) + \frac{\ln N - \ln \eta}{l}\left(1+\sqrt{1+\frac{2R_{emp}(T_i)l}{\ln N - \ln \eta}}\right)$
ALL YOUR BAYES ARE BELONG TO US

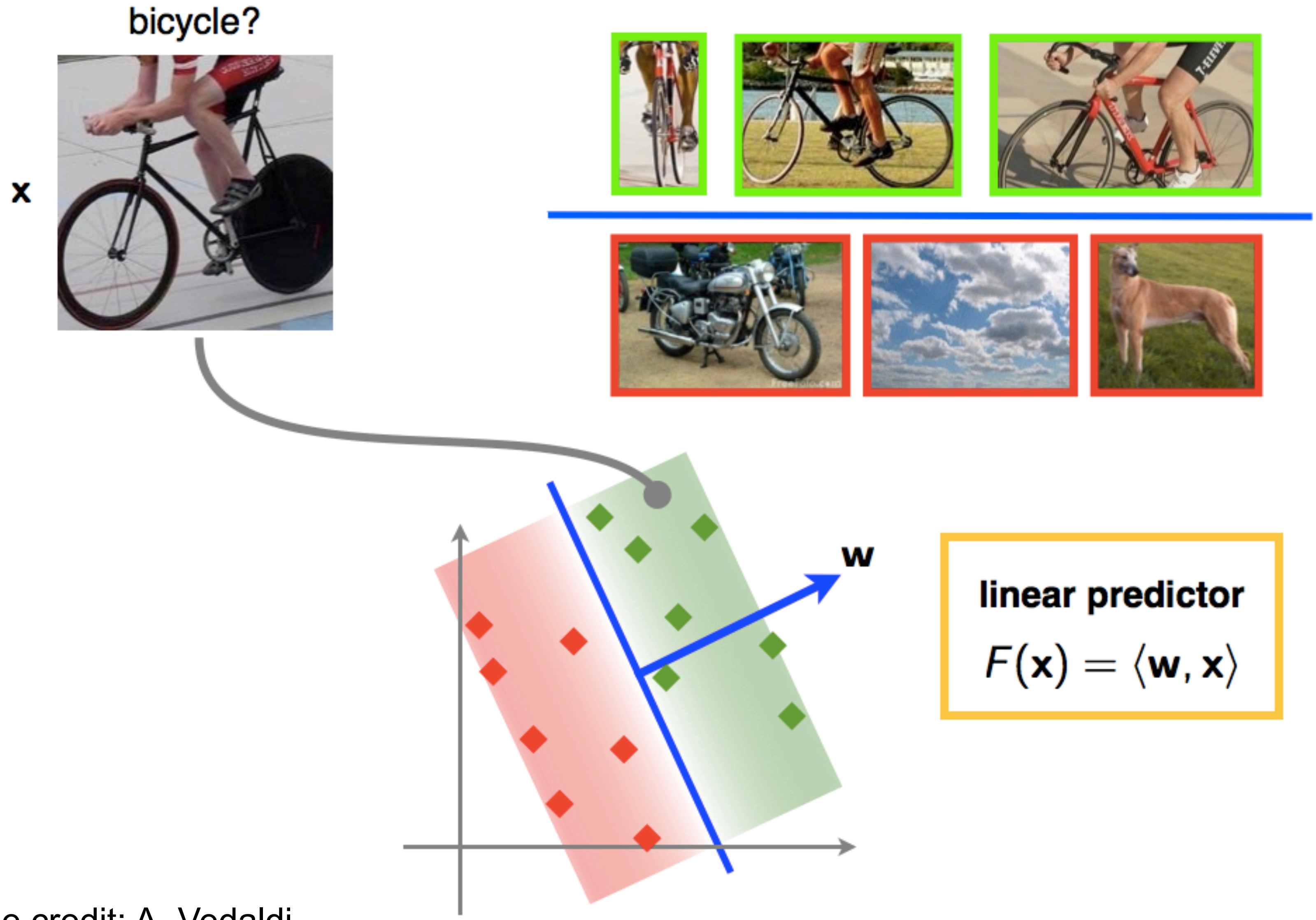
“There’s nothing more practical than a good theory”

Statistical Learning Theory
Vladimir N. Vapnik
A Volume in the Wiley Series on Adaptive and Learning Systems for Signal Processing, Communications, and Control
Simon Haykin, Series Editor
Nello Cristianini
John Shawe-Taylor
An Introduction to
Support Vector Machines
and other kernel-based learning methods
John Shawe-Taylor
and Nello Cristianini
Kernel Methods
for Pattern Analysis
Cambridge
LEARNING TO CLASSIFY TEXT USING SUPPORT VECTOR MACHINES
Methods, Theory and Algorithms
Copyrighted Material
Learning with Kernels
Support Vector Machines, Regularization, Optimization, and Beyond
Bernhard Schölkopf and Alexander J. Smola
Copyrighted Material
LARGE-SCALE KERNEL MACHINES
edited by Léon Bottou, Olivier Chapelle, Dennis DeCoste, and Jason Weston
Yunqian Ma · Guodong Guo Editors
Support Vector Machines Applications
Springer

Support vectors for Faces (P&P 98)



SVMs in computer vision

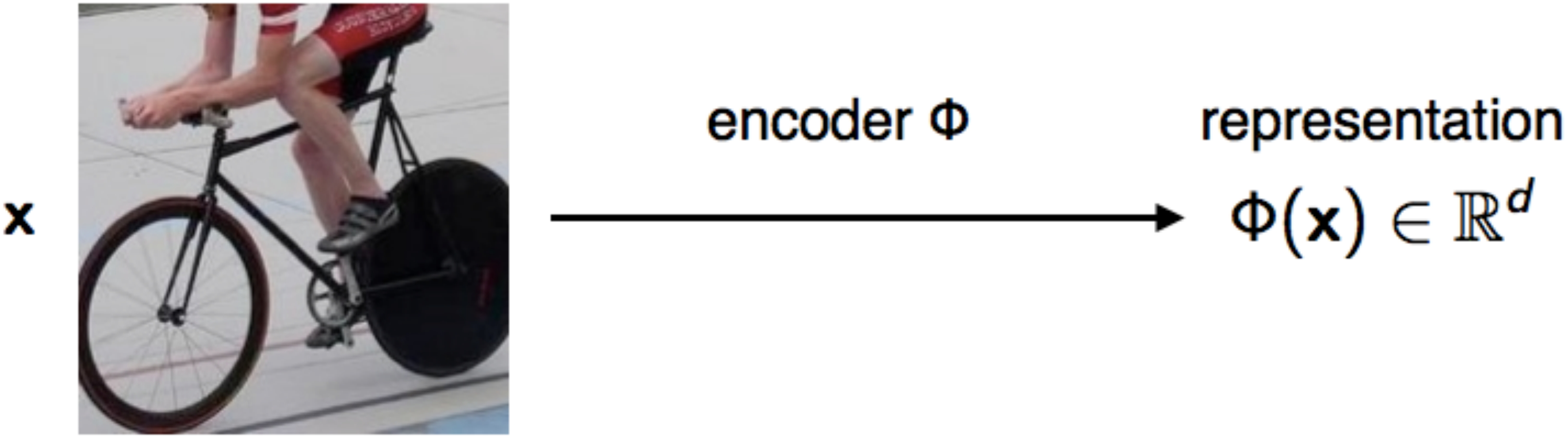


linear predictor

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

Slide credit: A. Vedaldi

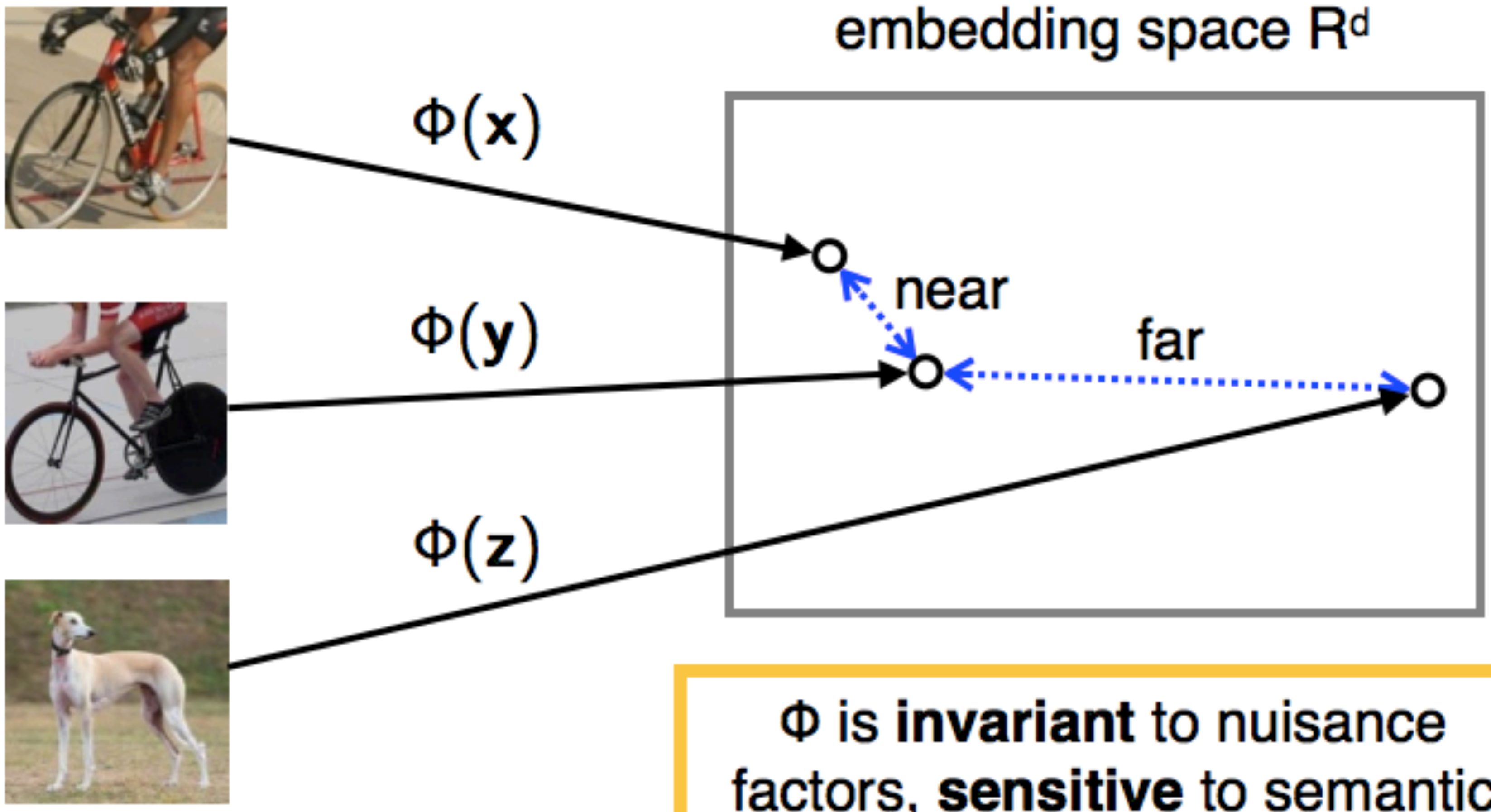
Image features



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

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

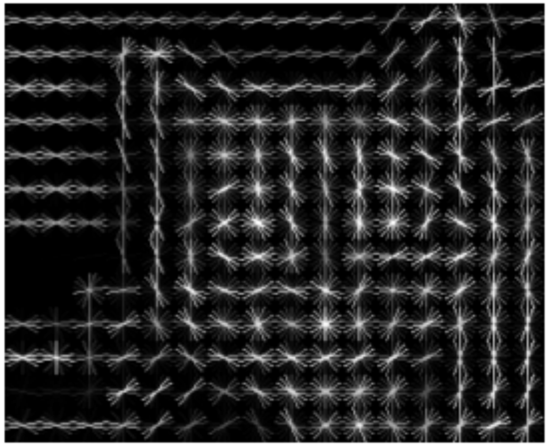
Desirable feature properties

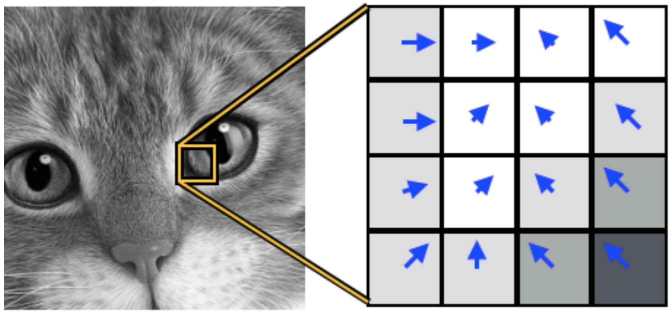


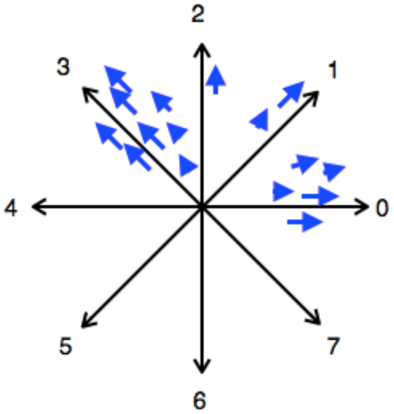
Φ is **invariant** to nuisance factors, **sensitive** to semantic variations

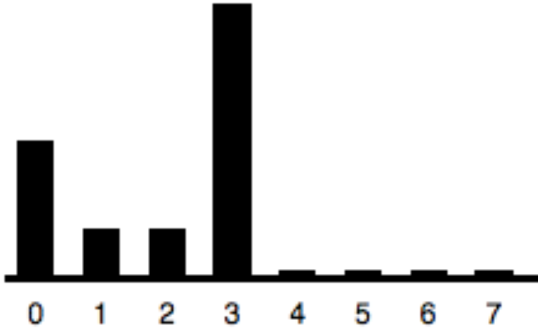
Slide credit: A. Vedaldi

Histogram of Gradient (HOG)/SIFT Features



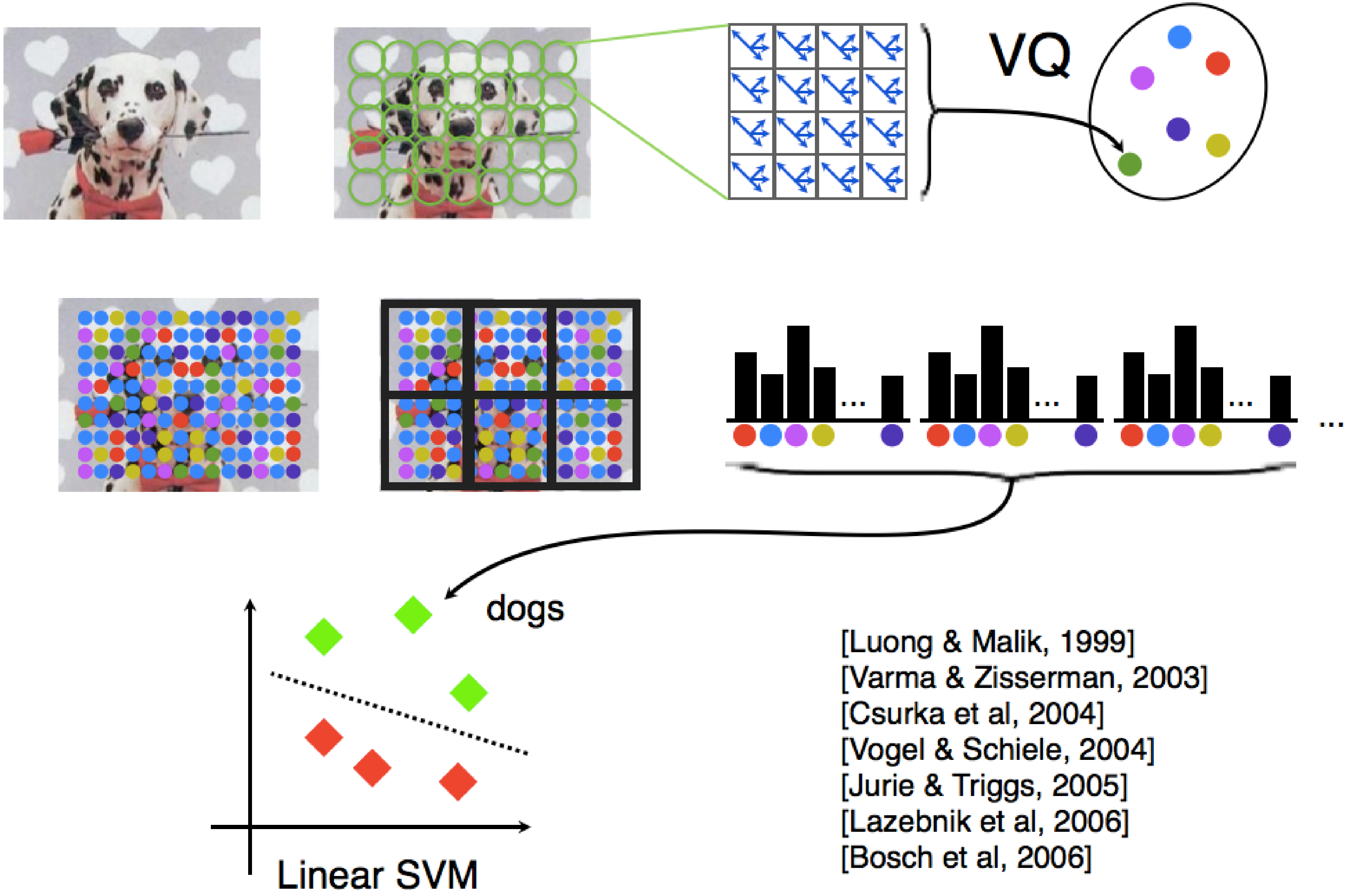






Slide credit: A. Vedaldi

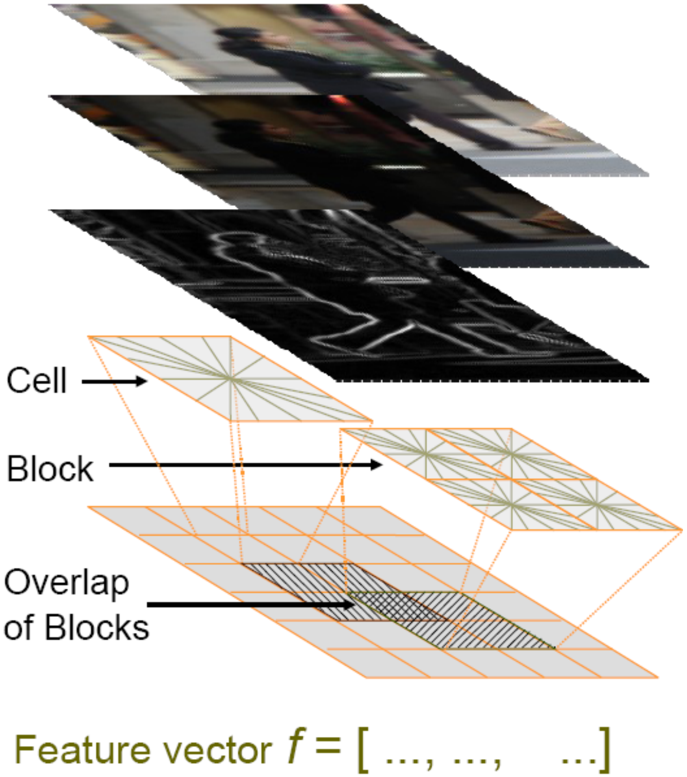
Image classification in a nutshell



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

Dalal and Triggs, ICCV 2005

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

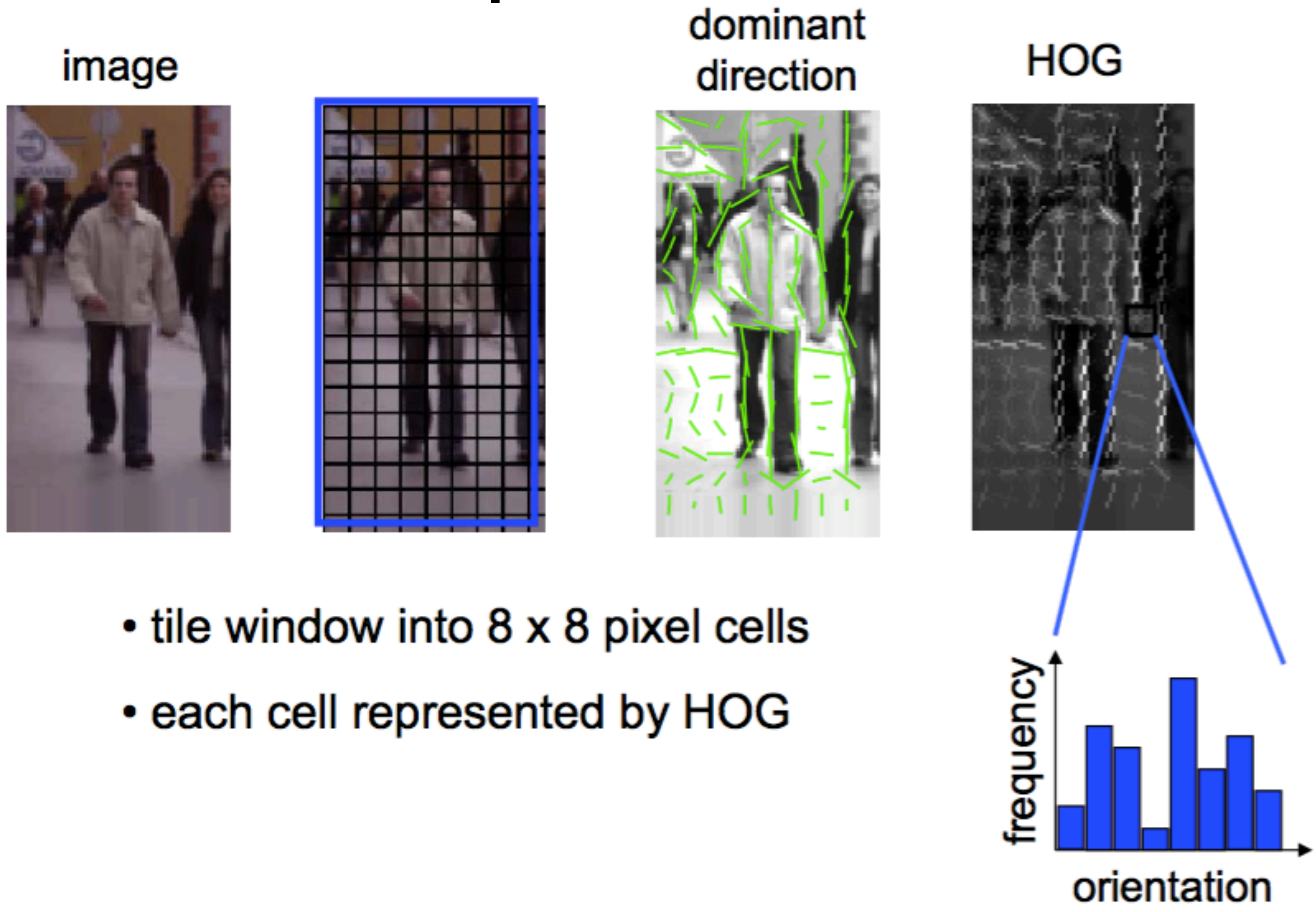

Cell

Block

Overlap of Blocks

Feature vector f = [..., ..., ...]

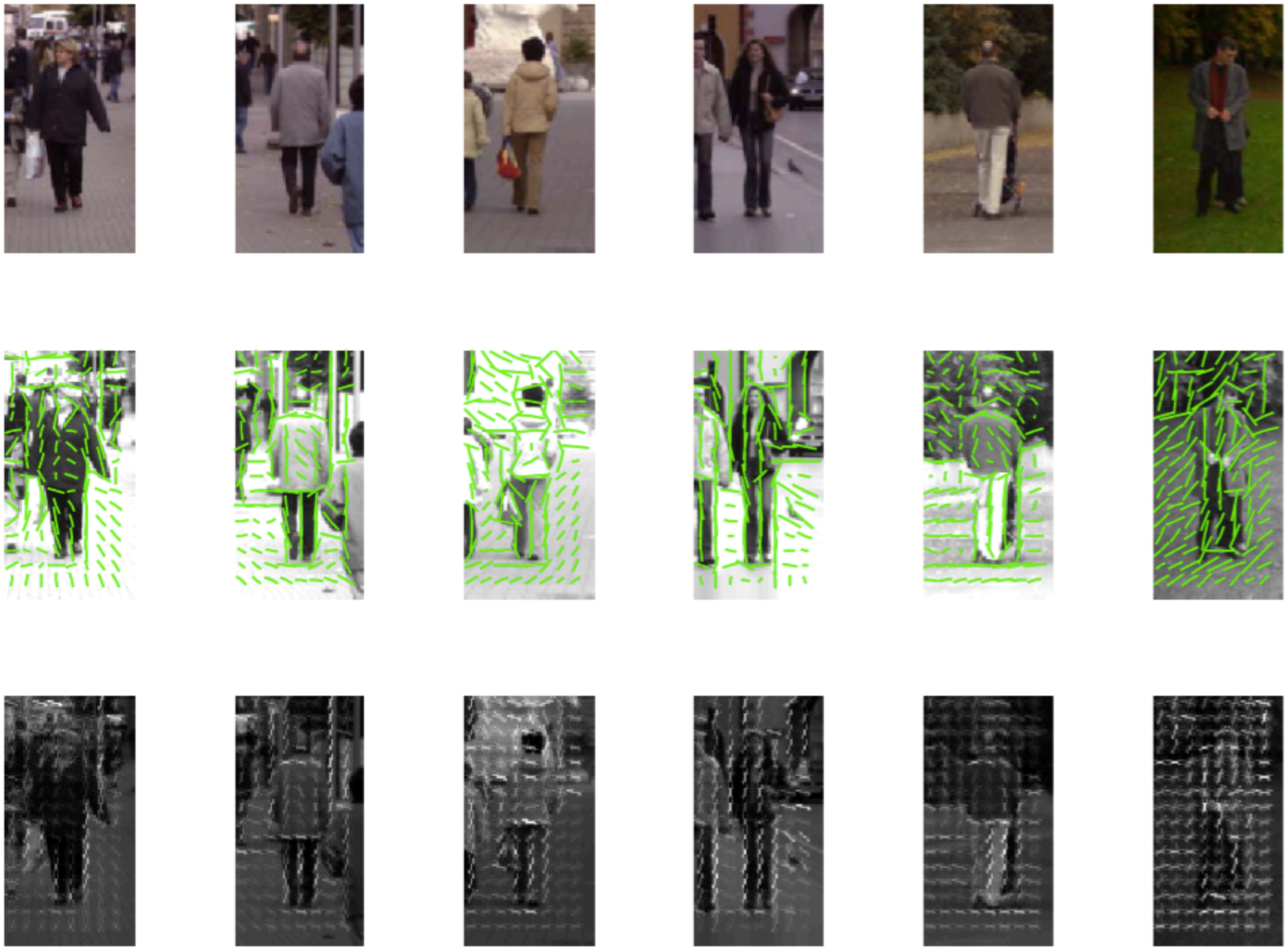
HOG features for pedestrians



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

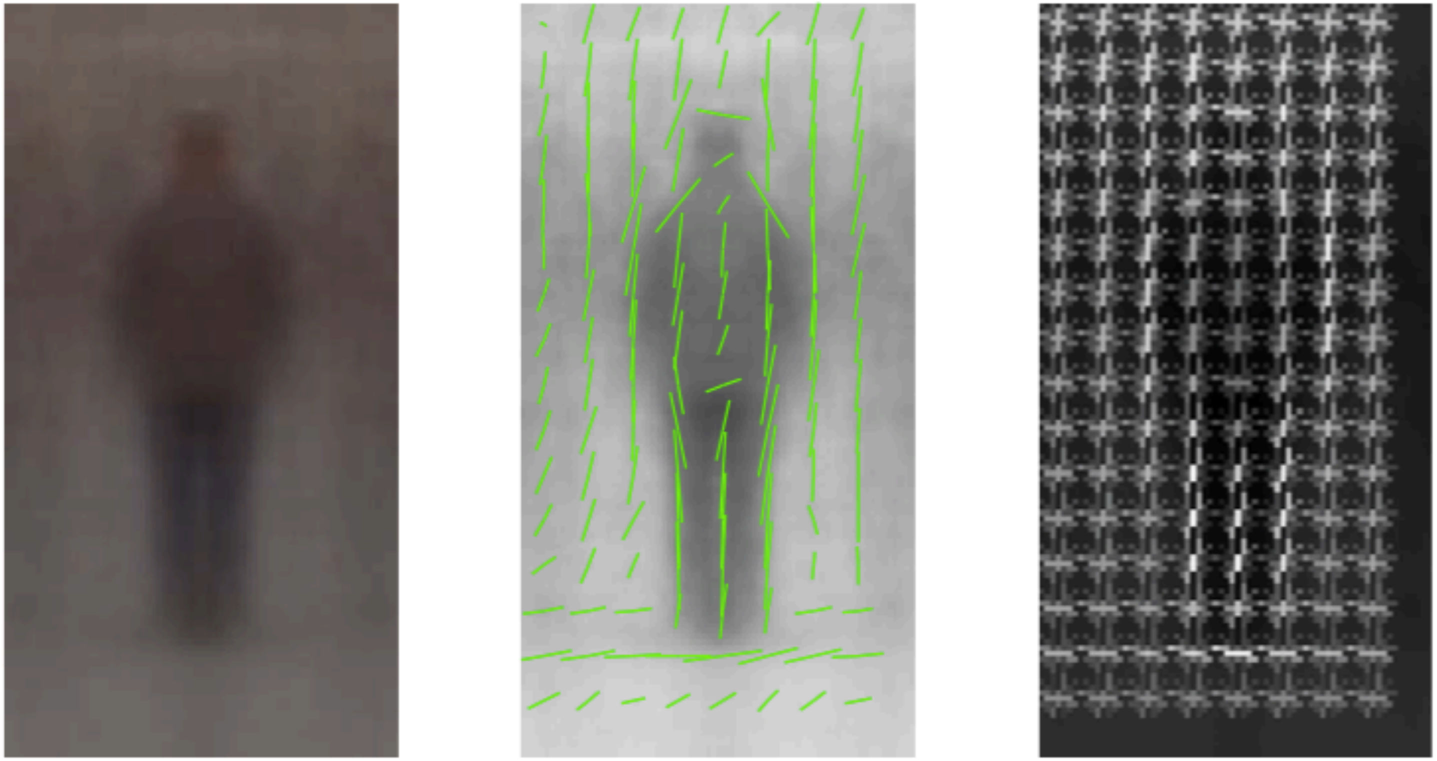
Feature vector dimension = 16 x 8 (for tiling) x 8 (orientations) = 1024

SVMs and Pedestrians



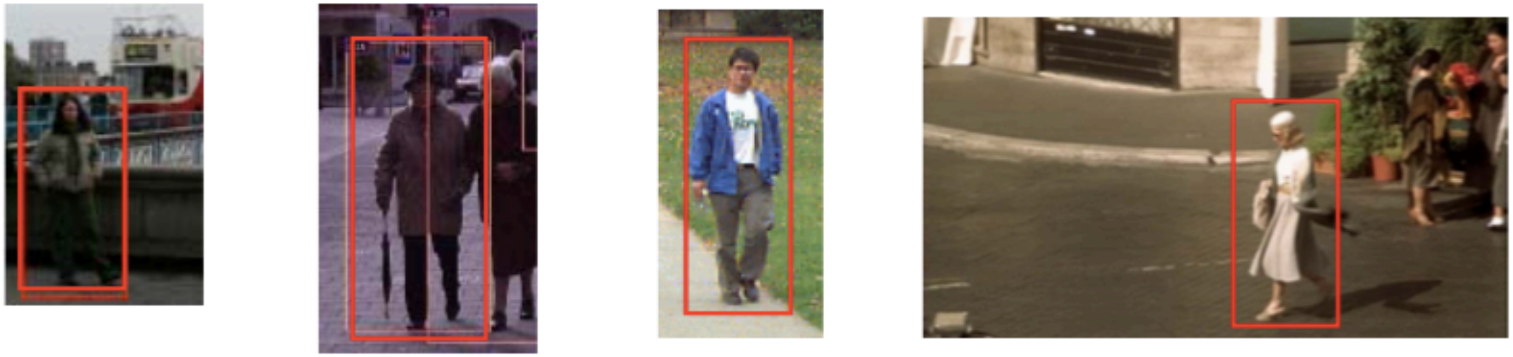
SVMs and Pedestrians

Averaged examples

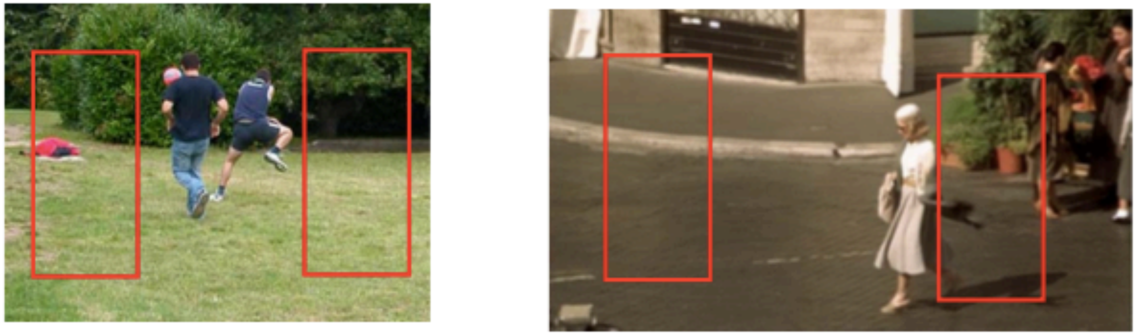


SVMs and Pedestrians

- Positive data – 1208 positive window examples

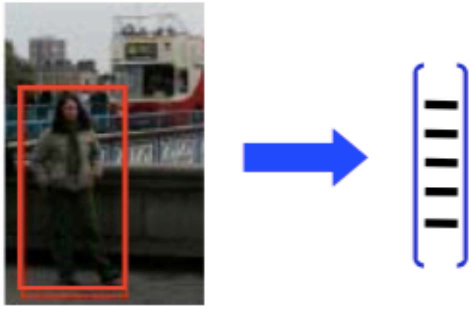


- Negative data – 1218 negative window examples (initially)



Training (Learning)

- Represent each example window by a HOG feature vector



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

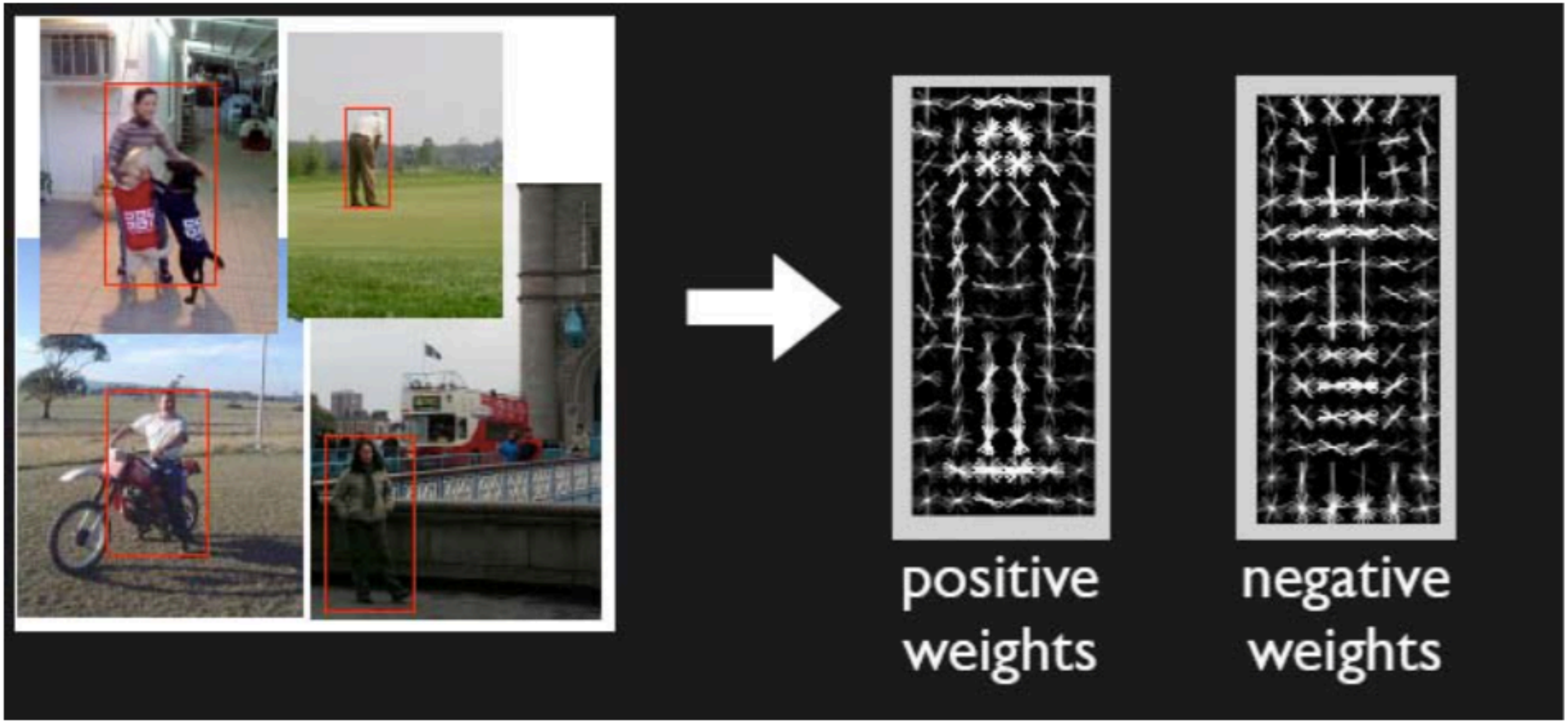
- Train a SVM classifier

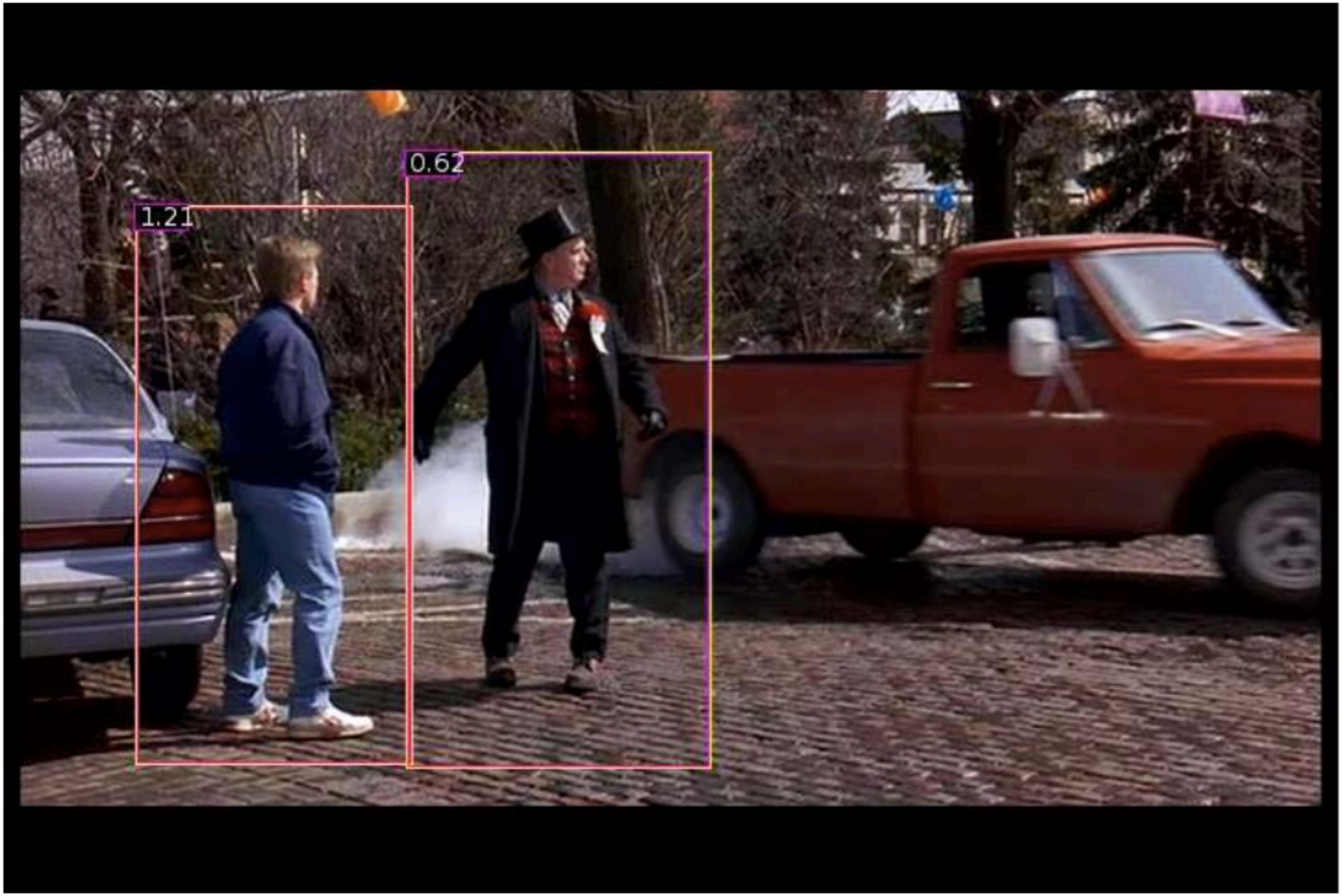
Testing (Detection)

- Sliding window classifier

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

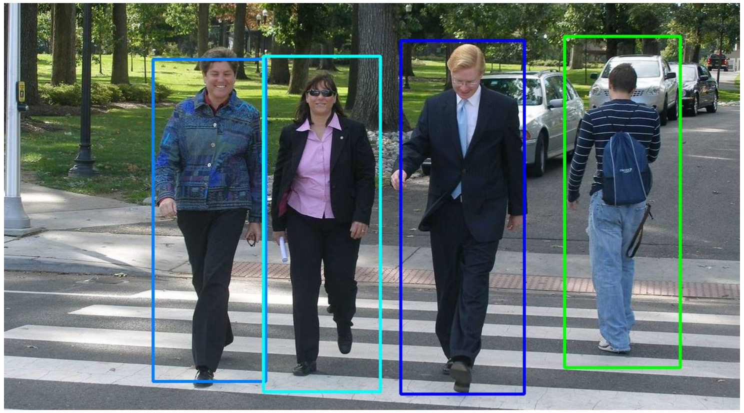
$$f(\mathbf{x}) = \mathbf{w}^\top \mathbf{x} + b$$

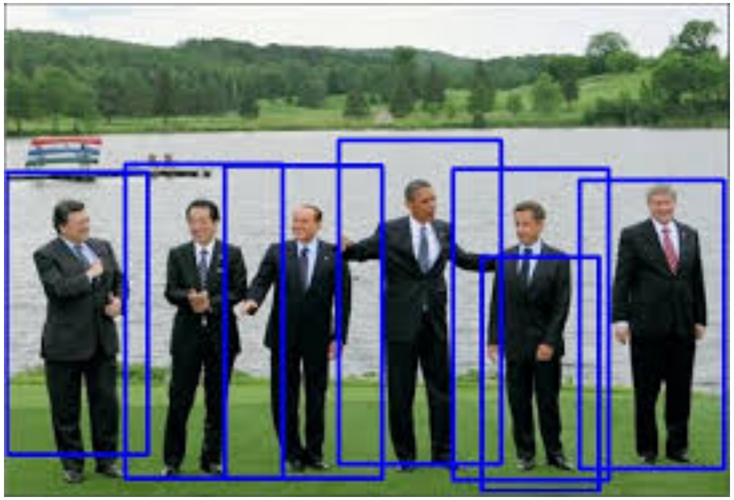


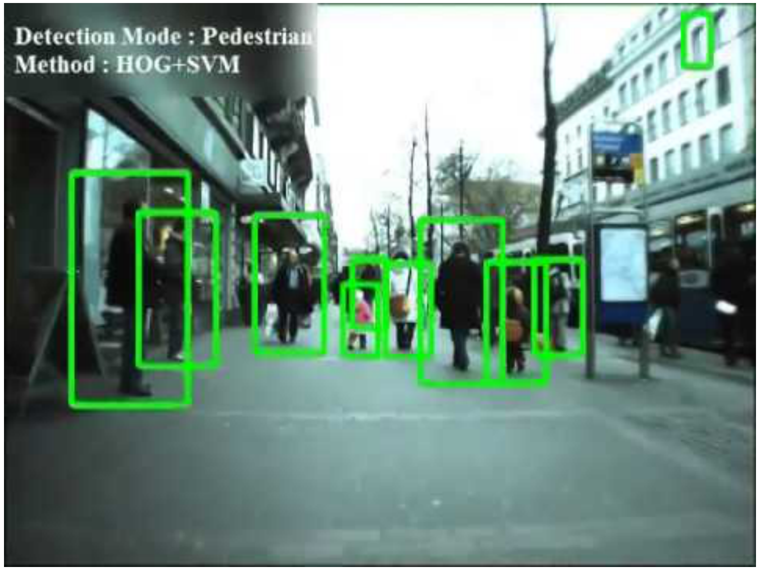


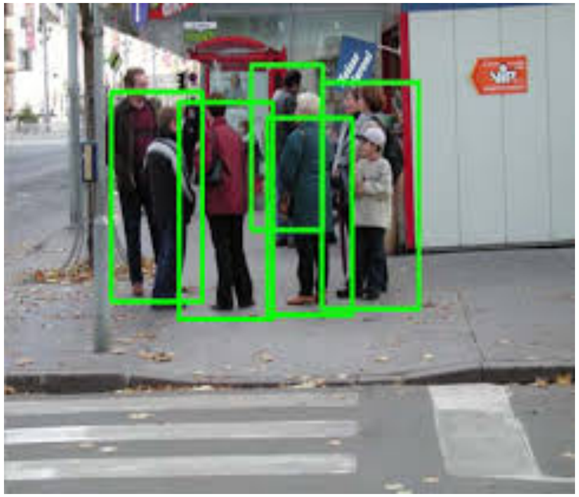
Dalal and Triggs, CVPR 2005

Pedestrian detection: almost done in 2005









Lecture outline

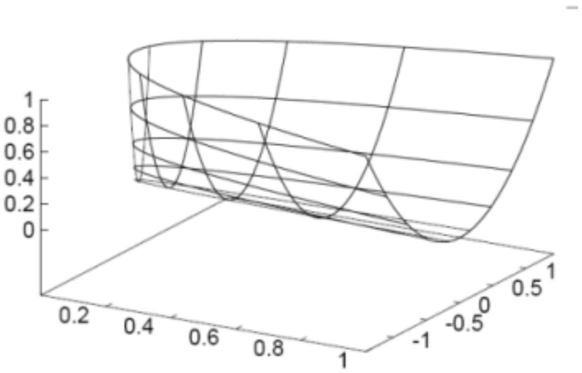
Recap

Large margins and generalization

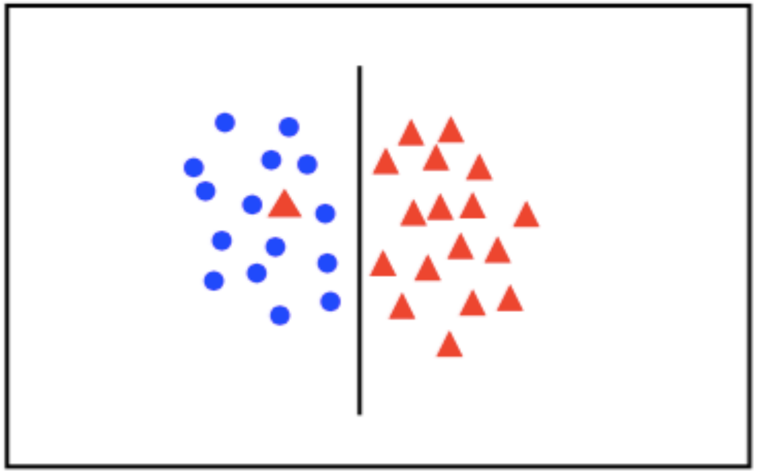
Optimization

Kernels

Applications to vision



Non-separable data

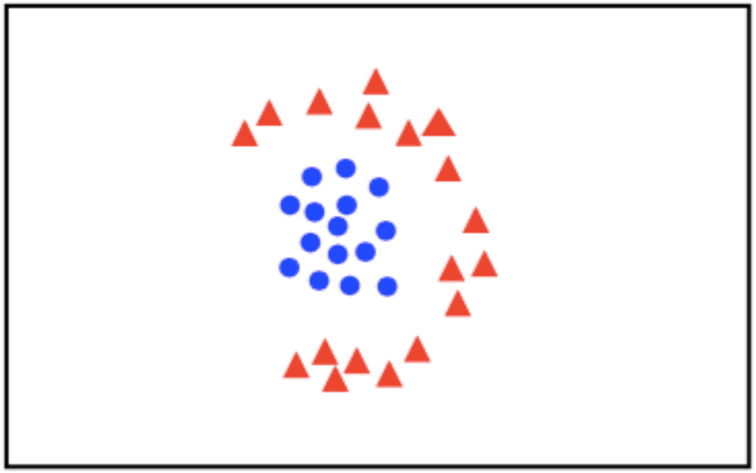


- introduce slack variables

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

subject to

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

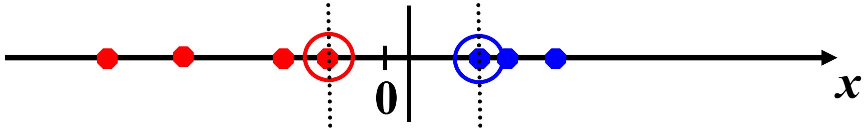


- linear classifier not appropriate

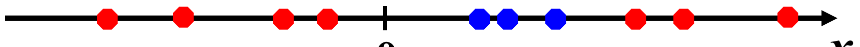
??

Non-linear SVMs

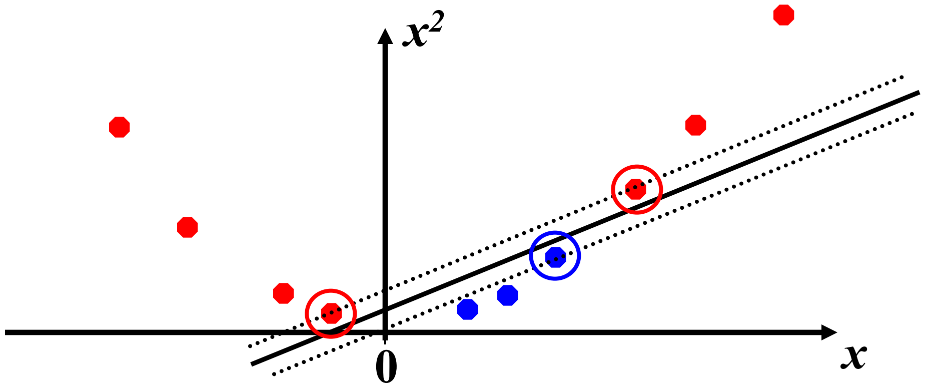
- Datasets that are linearly separable (with some noise) work out great:



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

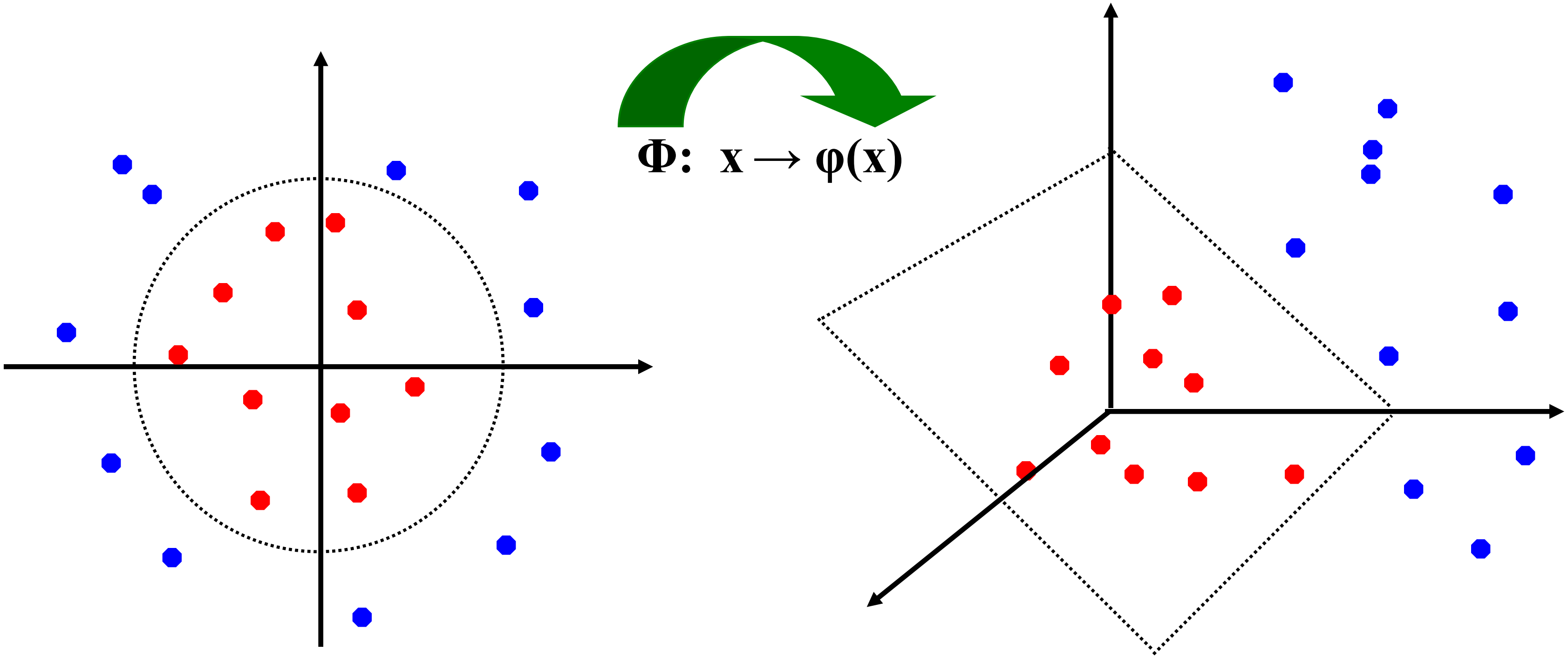


- How about … mapping data to a higher-dimensional space:

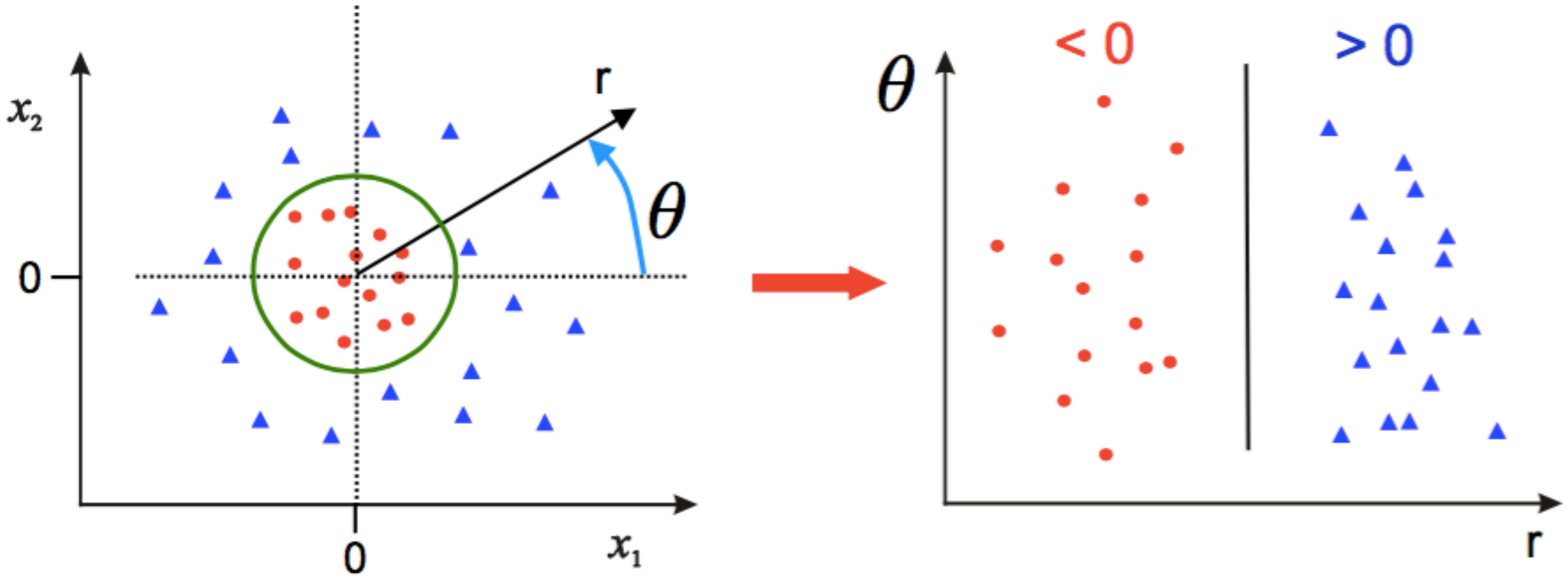


Non-linear SVMs: Feature spaces

- 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

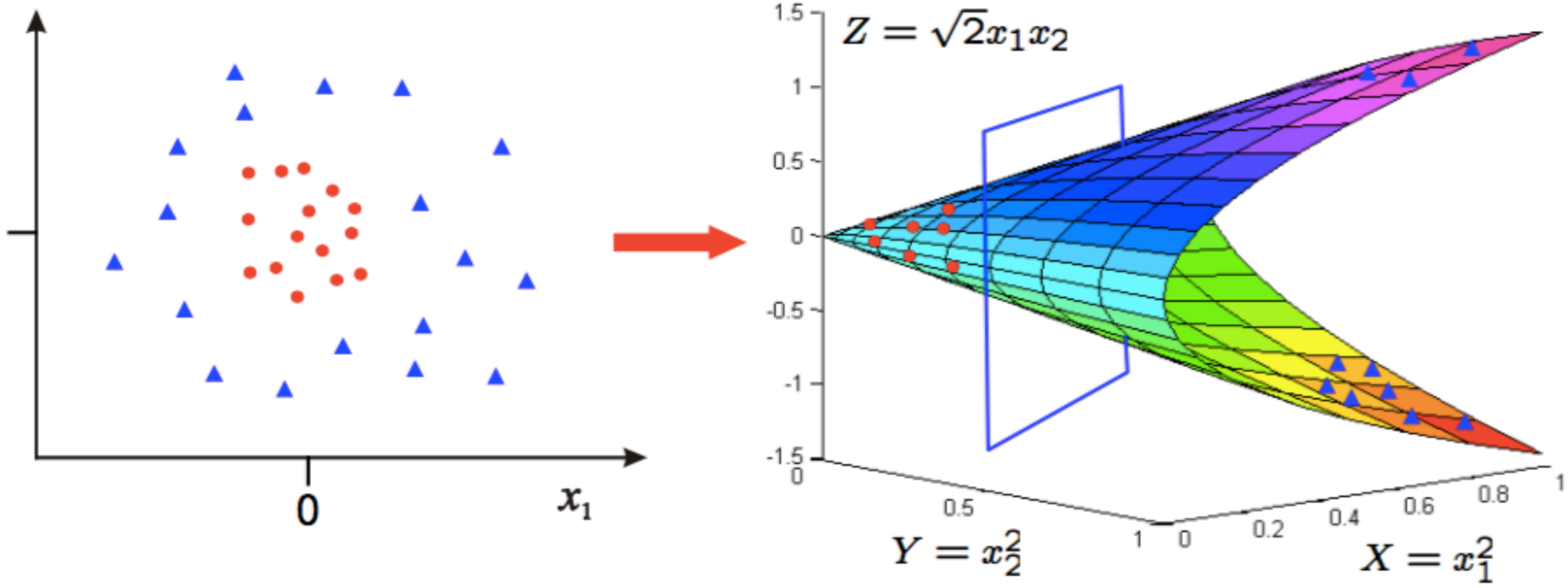


- Data is linearly separable in polar coordinates
- Acts non-linearly in original space

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

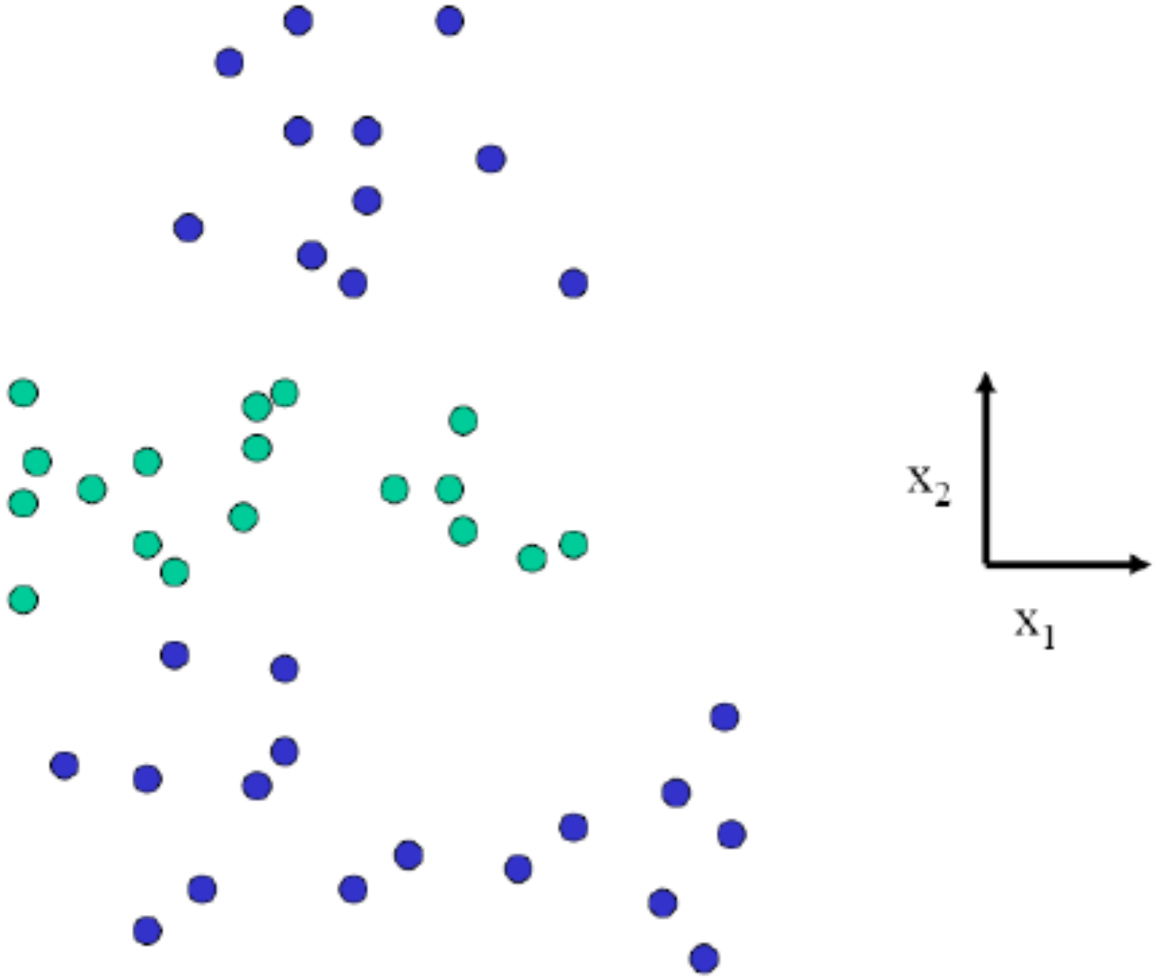
More general method

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

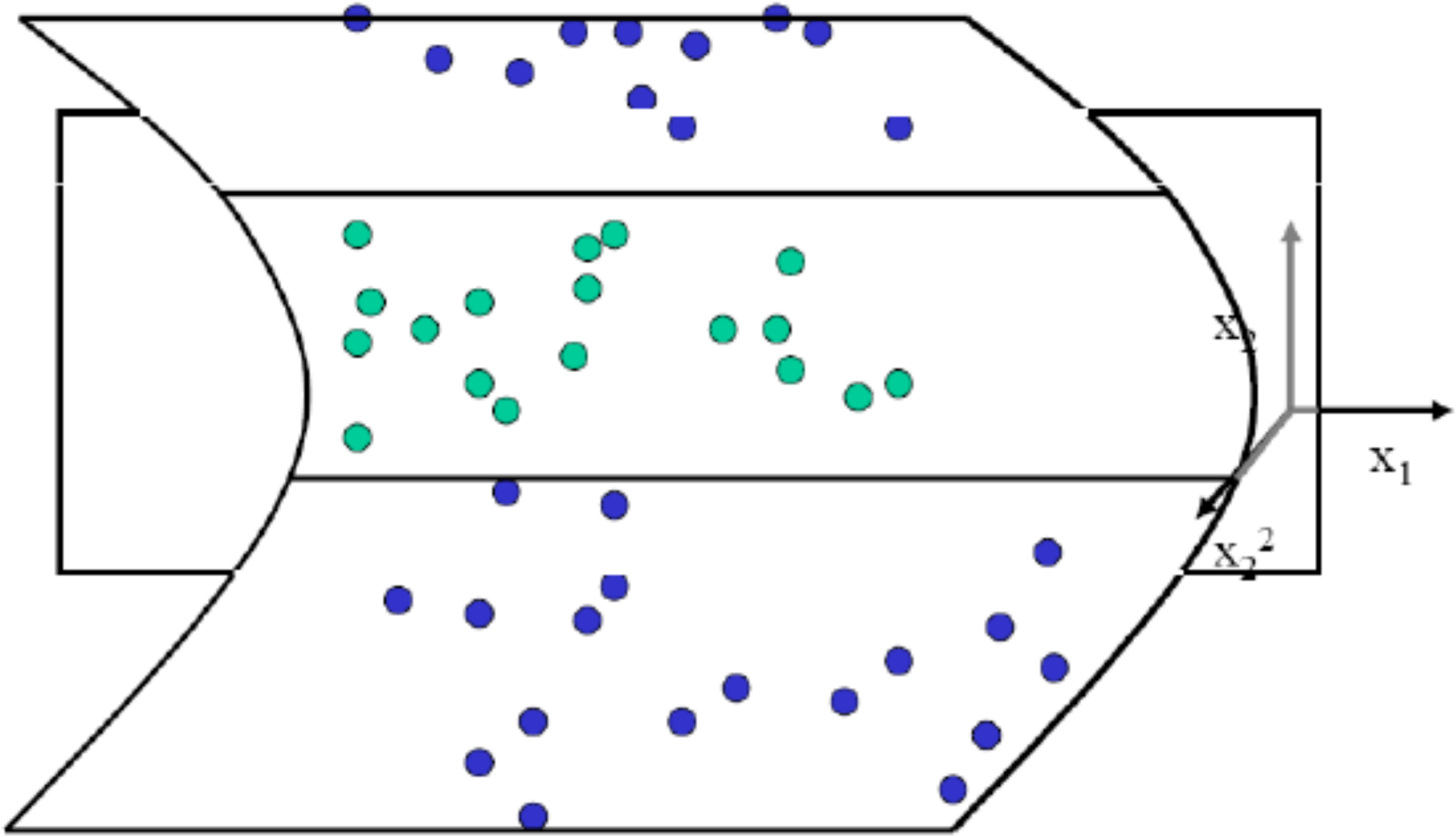


- Data is linearly separable in 3D
- This means that the problem can still be solved by a linear classifier

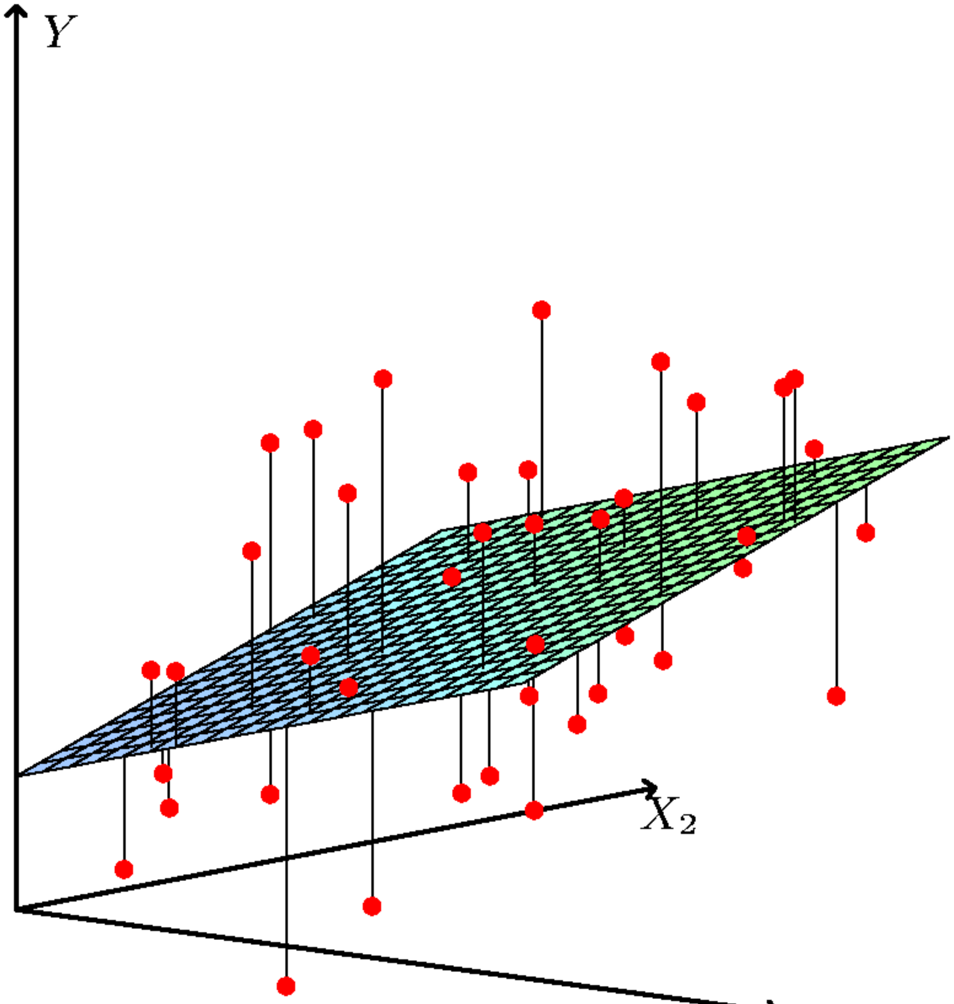
Nonseparable in 2D



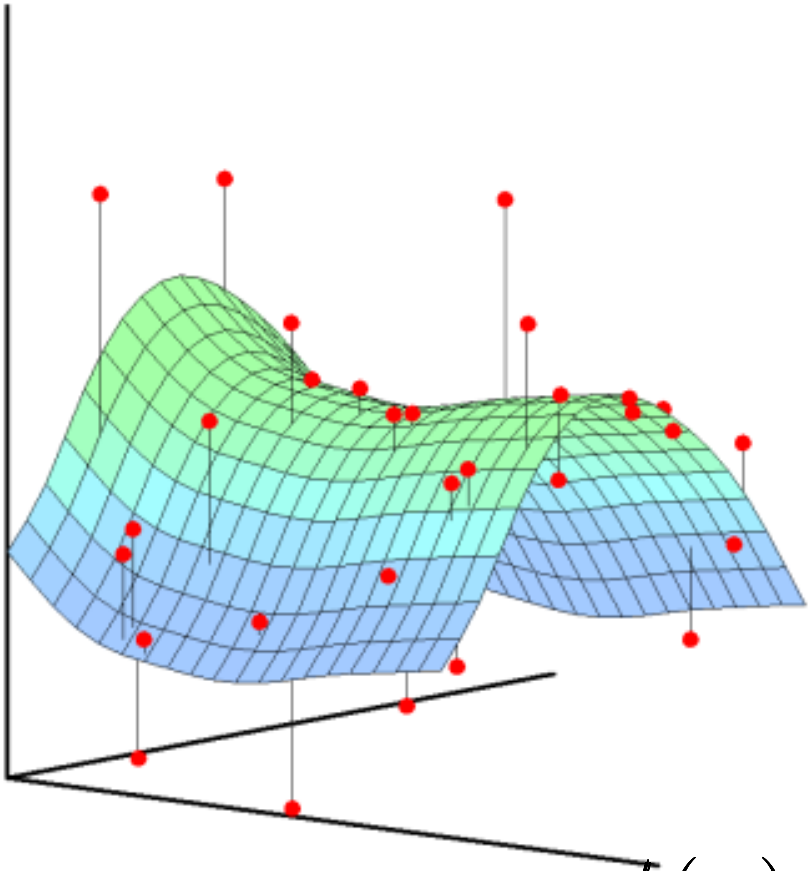
Separable in 3D



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

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

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

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

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

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

$$f(\mathbf{x}) = \sum_{i}^{N} \underbrace{\alpha_i}_{\text{weight (may be zero)}} y_i k(\underbrace{\mathbf{x}_i}_{\text{support vector}}, \mathbf{x}) + b$$

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

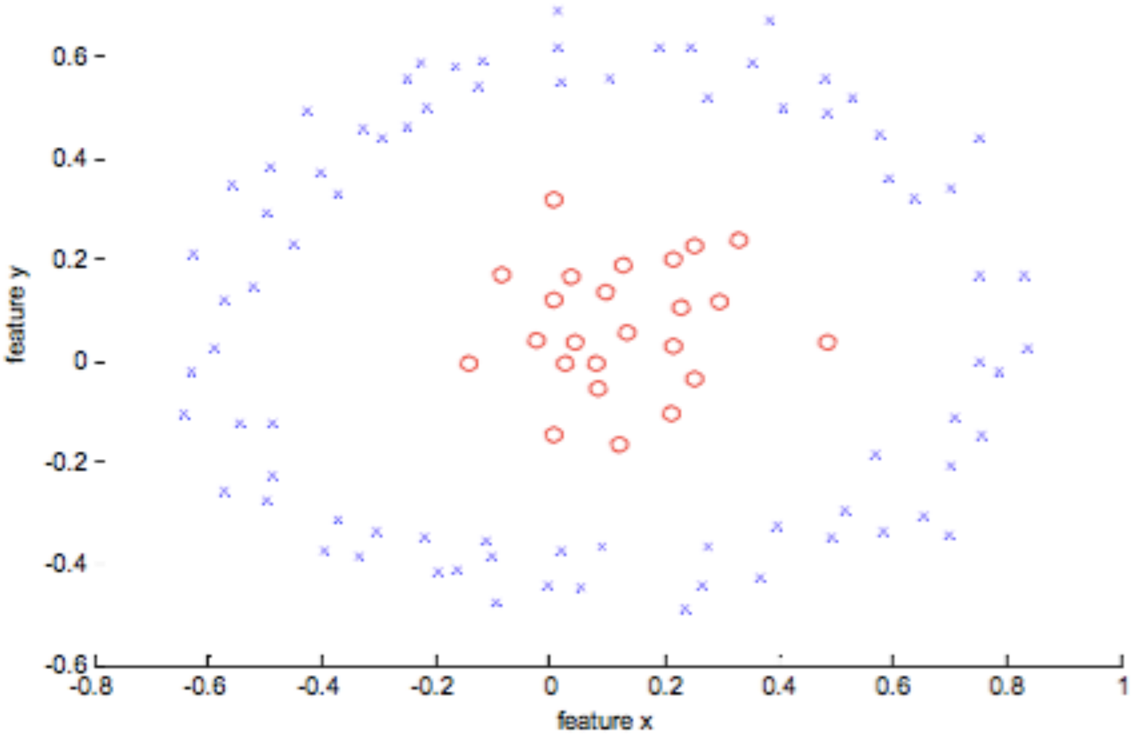
$$f(\mathbf{x}) = \sum_{i=1}^{N} \alpha^i y^i K(\mathbf{x}^i, \mathbf{x}) + b$$

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

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

Discriminant form: sum of bumps centered on training points

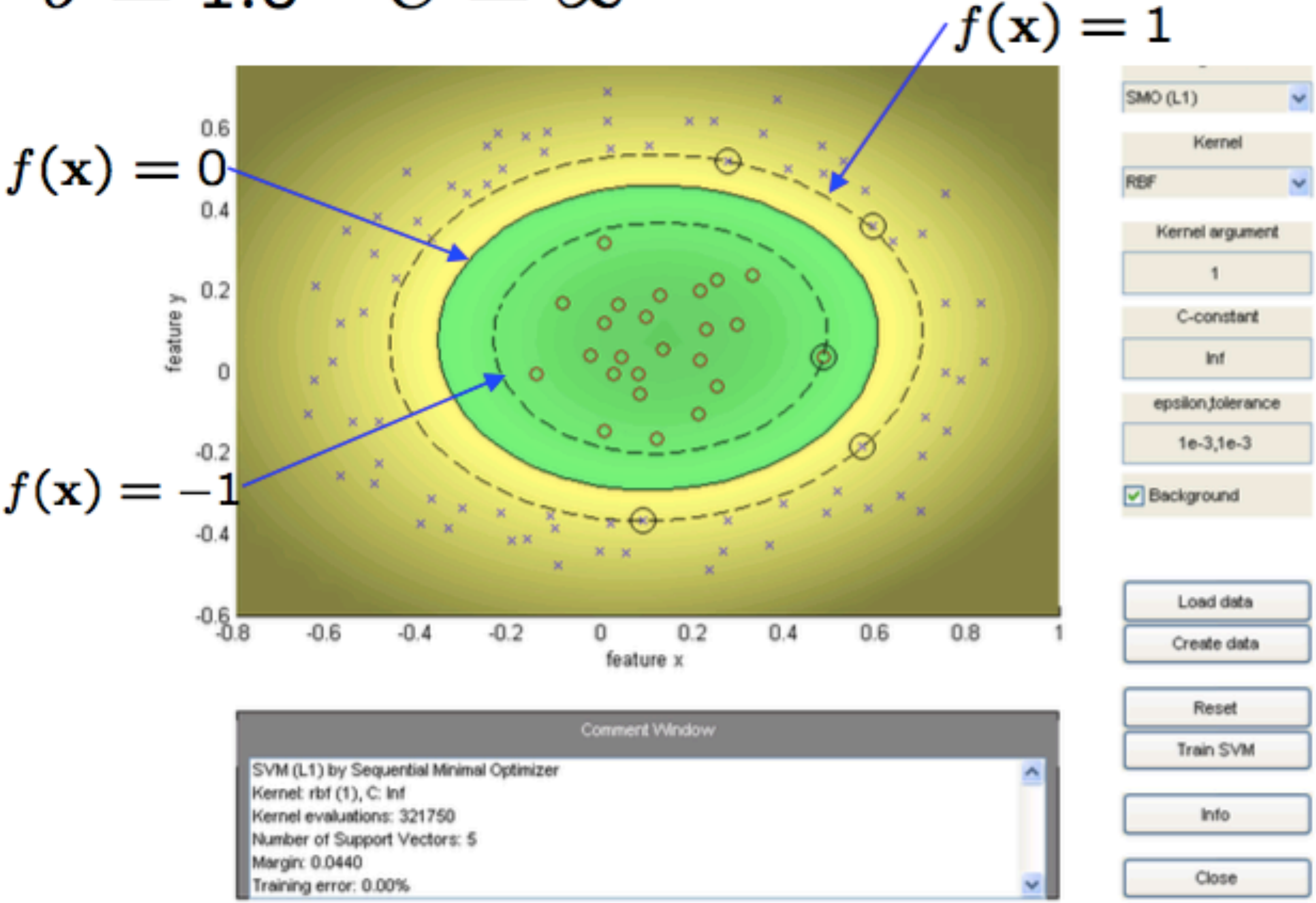
RBF-SVM example



- data is not linearly separable in original feature space

RBF-SVM example

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



$f(\mathbf{x}) = 1$

$f(\mathbf{x}) = 0$

$f(\mathbf{x}) = -1$

SMO (L1)

Kernel: RBF

Kernel argument: 1

C-constant: Inf

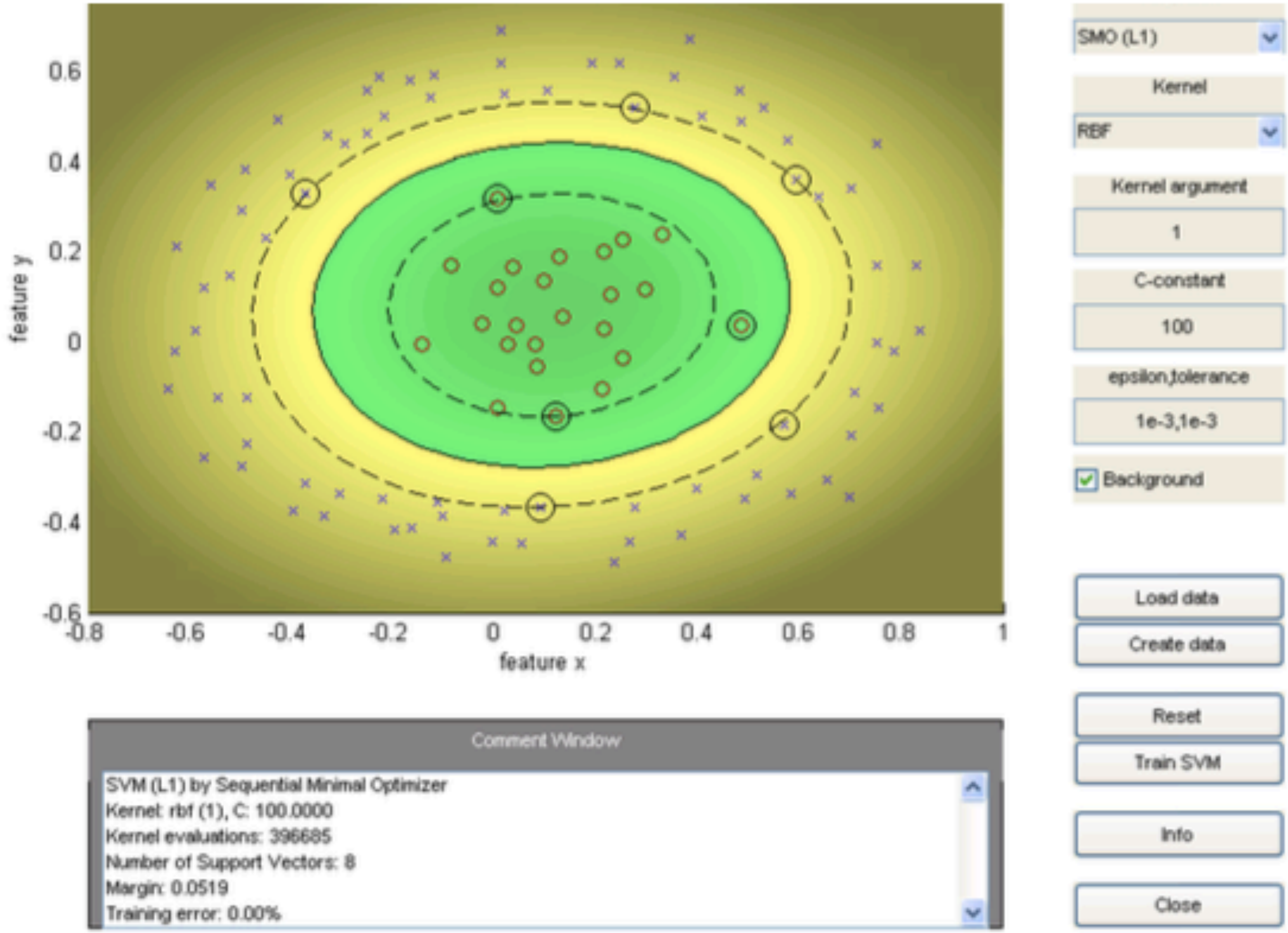
epsilon,tolerance: 1e-3,1e-3

☑ Background

```
SVM (L1) by Sequential Minimal Optimizer
Kernel: rbf (1), C: Inf
Kernel evaluations: 321750
Number of Support Vectors: 5
Margin: 0.0440
Training error: 0.00%
```

RBF-SVM example

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



SMO (L1)

Kernel: RBF

Kernel argument: 1

C-constant: 100

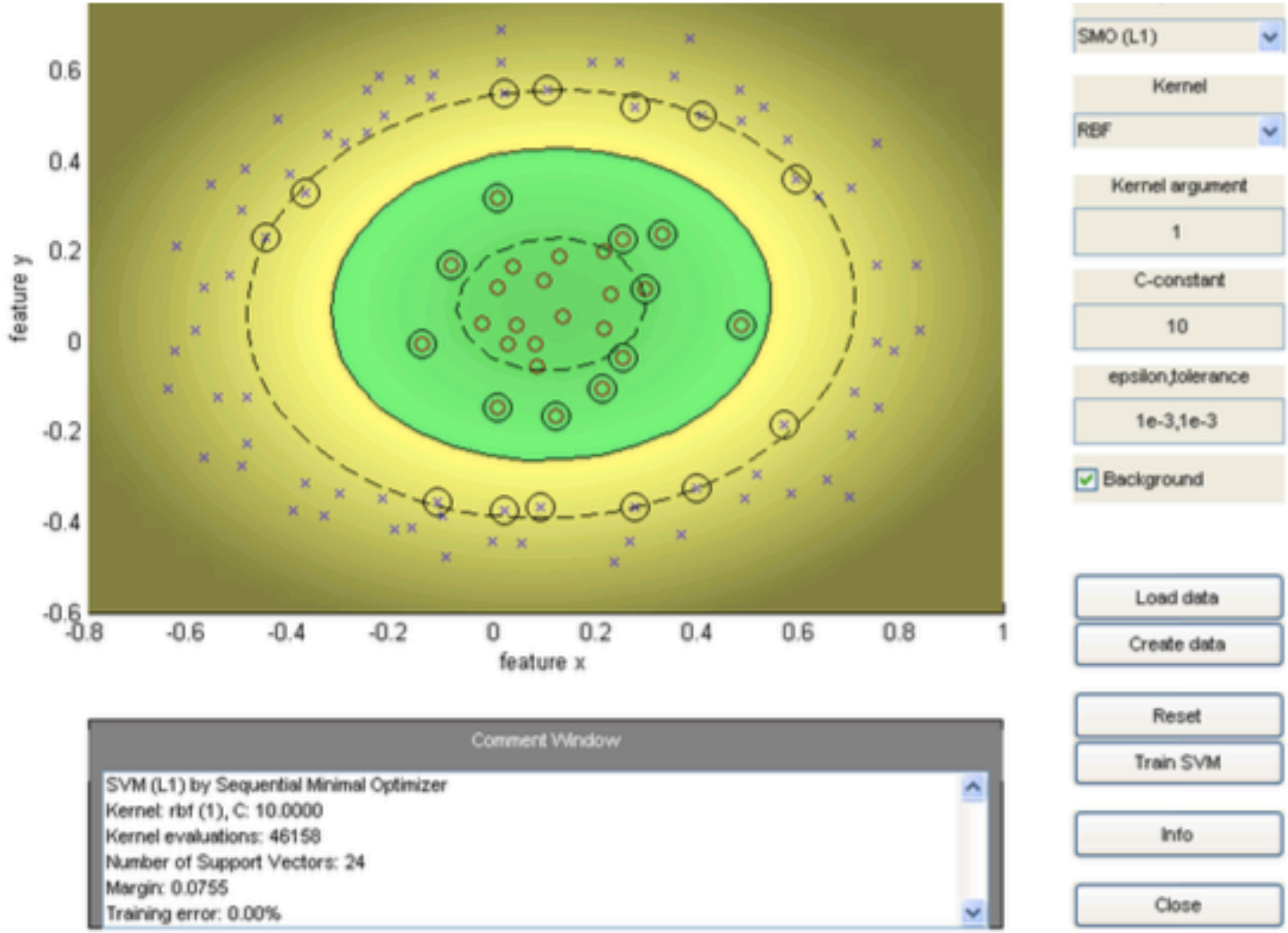
epsilon,tolerance: 1e-3,1e-3

☑ Background

```
SVM (L1) by Sequential Minimal Optimizer
Kernel: rbf (1), C: 100.0000
Kernel evaluations: 396685
Number of Support Vectors: 8
Margin: 0.0519
Training error: 0.00%
```

RBF-SVM example

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



SMO (L1)

Kernel: RBF

Kernel argument: 1

C-constant: 10

epsilon,tolerance: 1e-3,1e-3

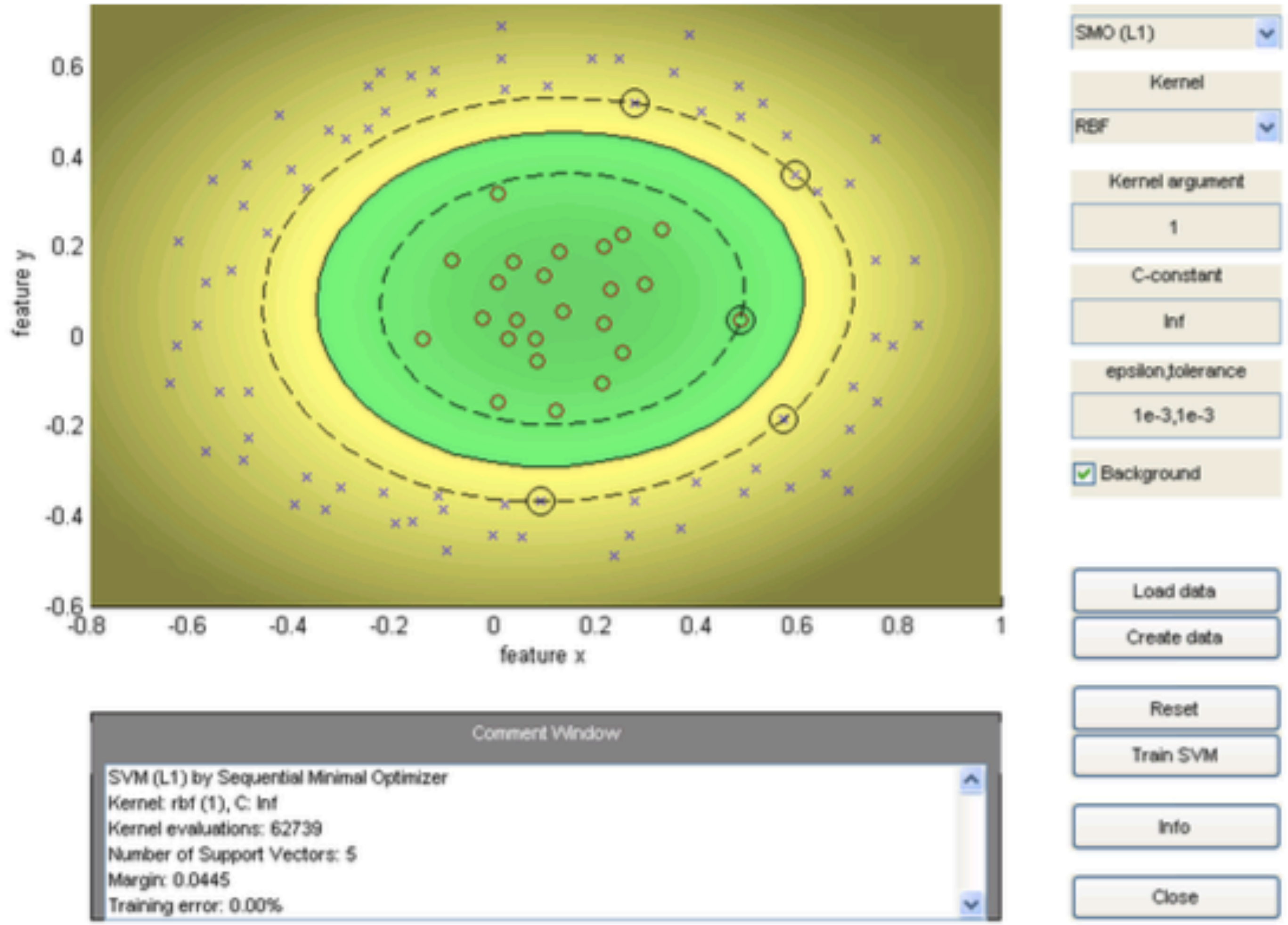
☑ Background

Comment Window

```
SVM (L1) by Sequential Minimal Optimizer
Kernel: rbf (1), C: 10.0000
Kernel evaluations: 46158
Number of Support Vectors: 24
Margin: 0.0755
Training error: 0.00%
```

RBF-SVM example

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



SMO (L1)

Kernel: RBF

Kernel argument: 1

C-constant: Inf

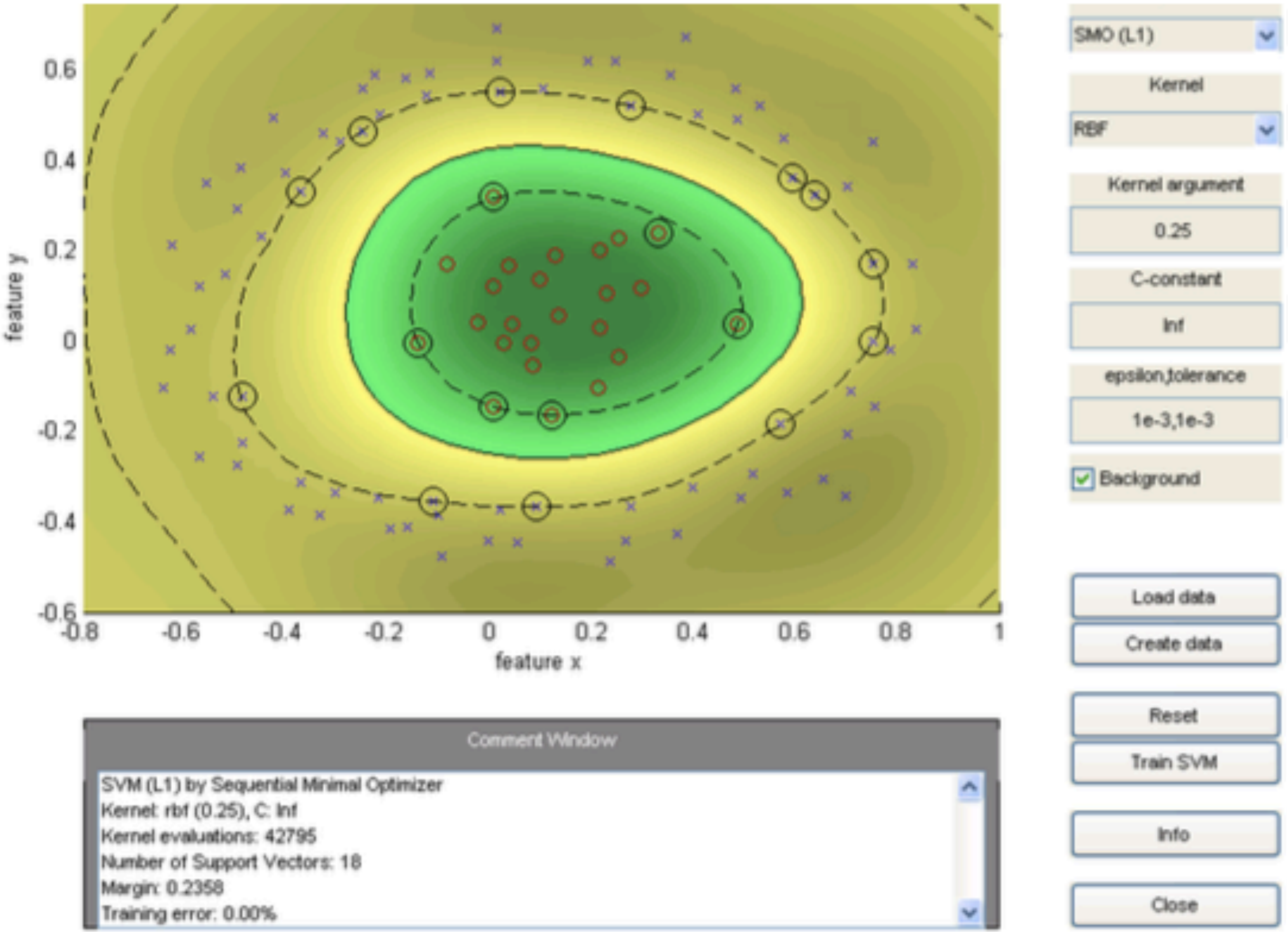
epsilon,tolerance: 1e-3,1e-3

☑ Background

```
SVM (L1) by Sequential Minimal Optimizer
Kernel: rbf (1), C: Inf
Kernel evaluations: 62739
Number of Support Vectors: 5
Margin: 0.0445
Training error: 0.00%
```

RBF-SVM example

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



SMO (L1)

Kernel
RBF

Kernel argument
0.25

C-constant
Inf

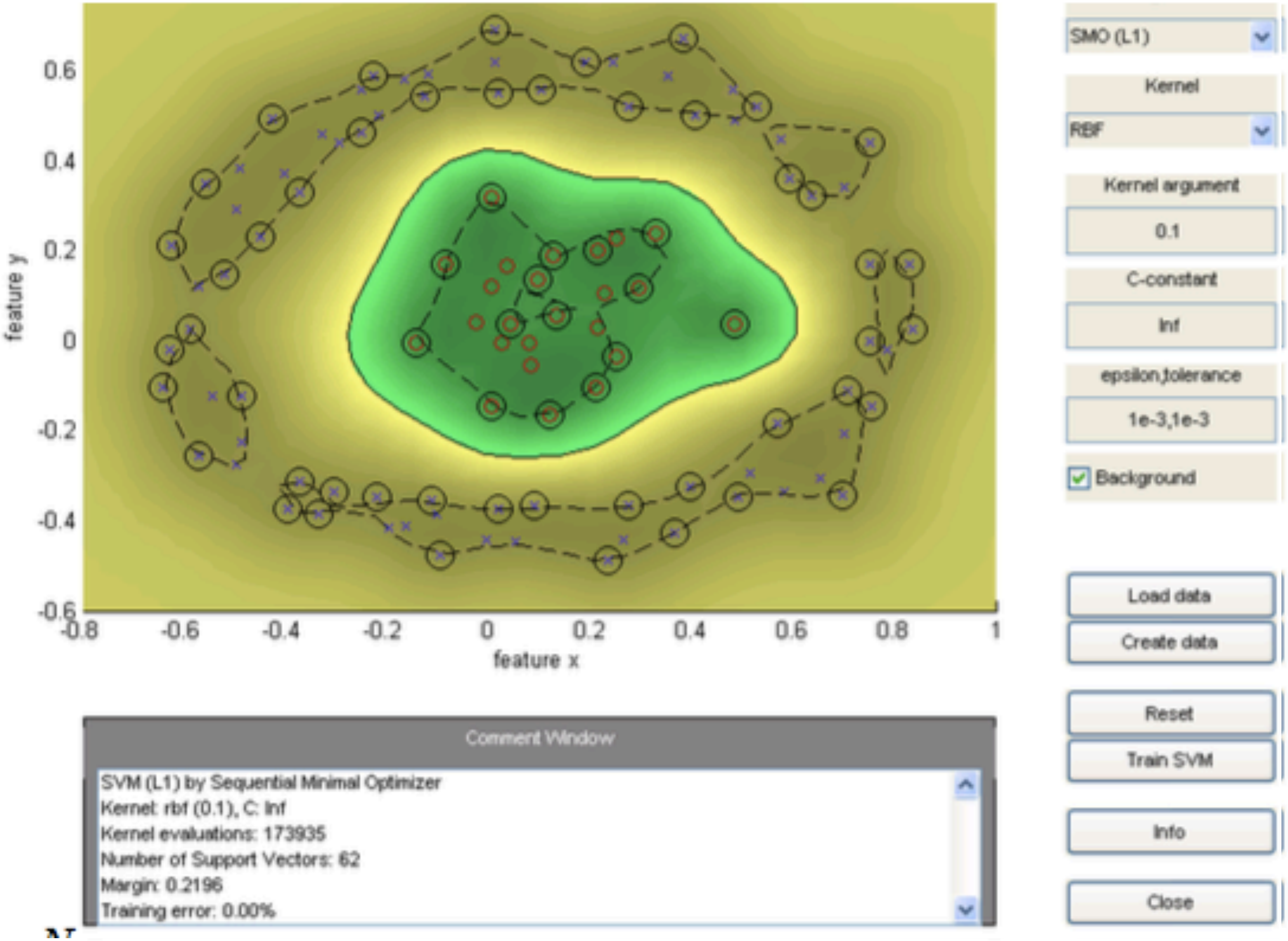
epsilon,tolerance
1e-3,1e-3

☑ Background

```
SVM (L1) by Sequential Minimal Optimizer
Kernel: rbf (0.25), C: Inf
Kernel evaluations: 42795
Number of Support Vectors: 18
Margin: 0.2358
Training error: 0.00%
```

RBF-SVM example

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



SMO (L1)

Kernel
RBF

Kernel argument
0.1

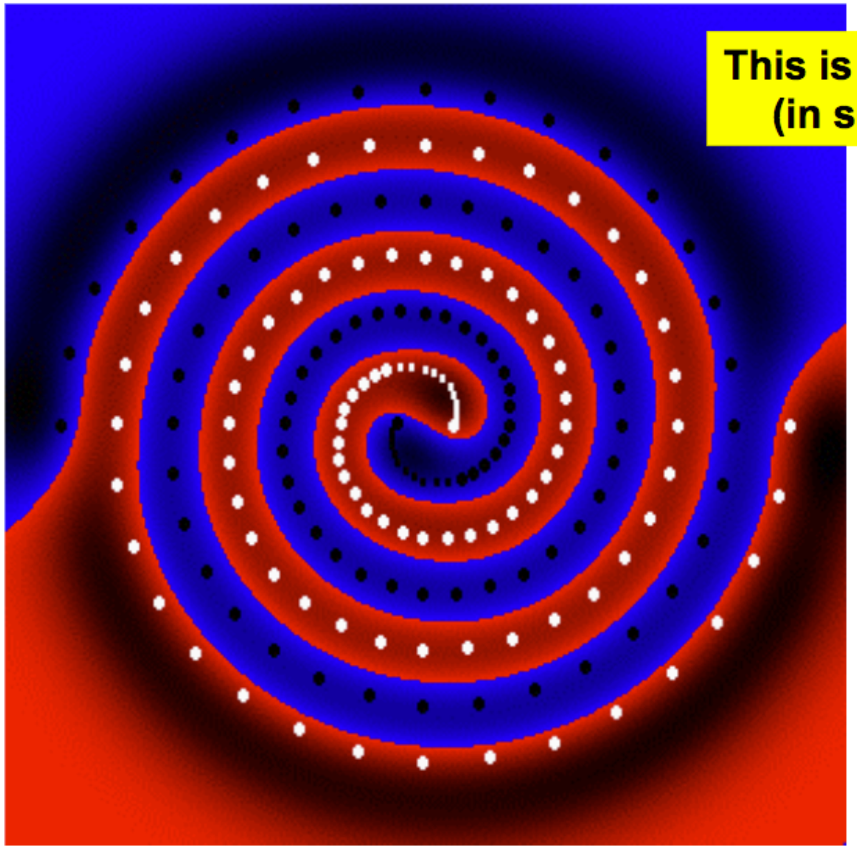
C-constant
Inf

epsilon,tolerance
1e-3,1e-3

☑ Background

```
SVM (L1) by Sequential Minimal Optimizer
Kernel: rbf (0.1), C: Inf
Kernel evaluations: 173935
Number of Support Vectors: 62
Margin: 0.2196
Training error: 0.00%
```

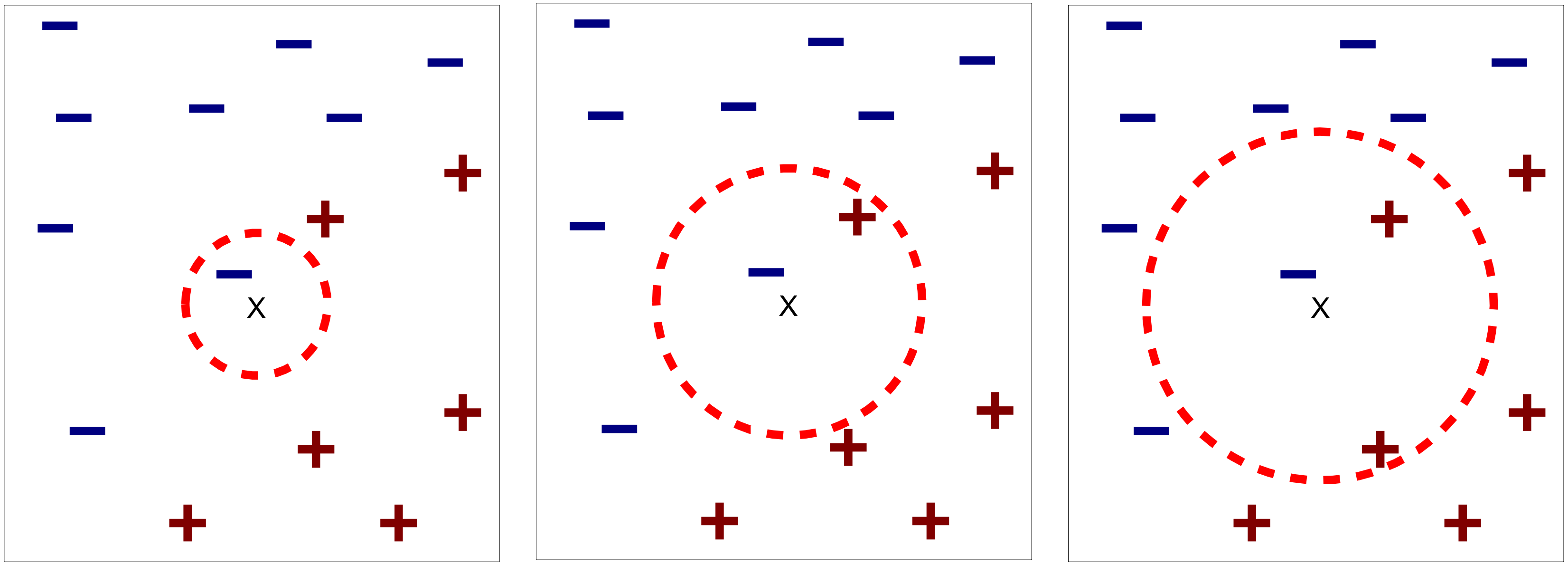
All of the flexibility you may need is there



This is a hyperplane!
(in some space)

www.kernel-methods.net

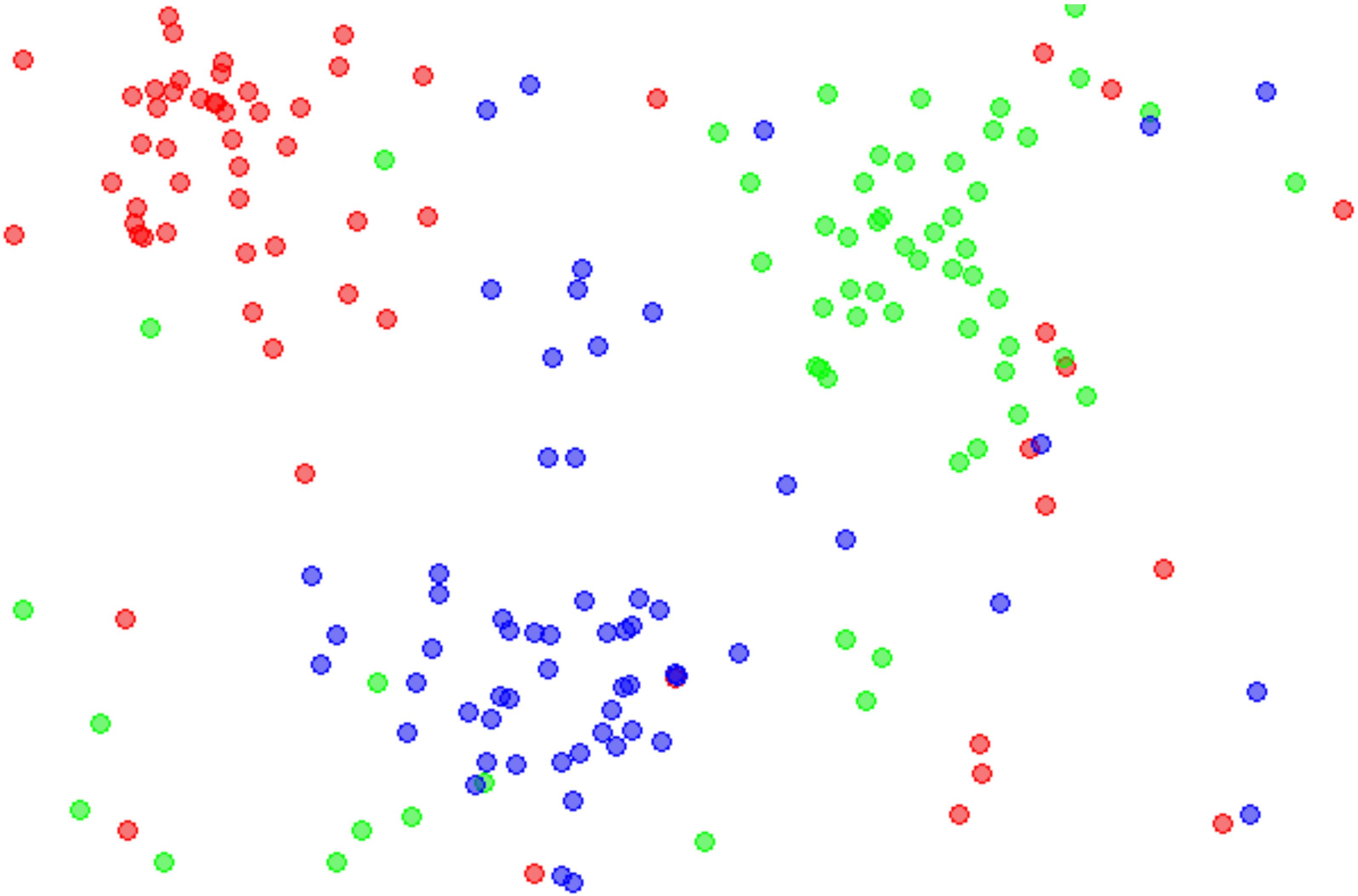
Reminder: K-nearest neighbor classifier



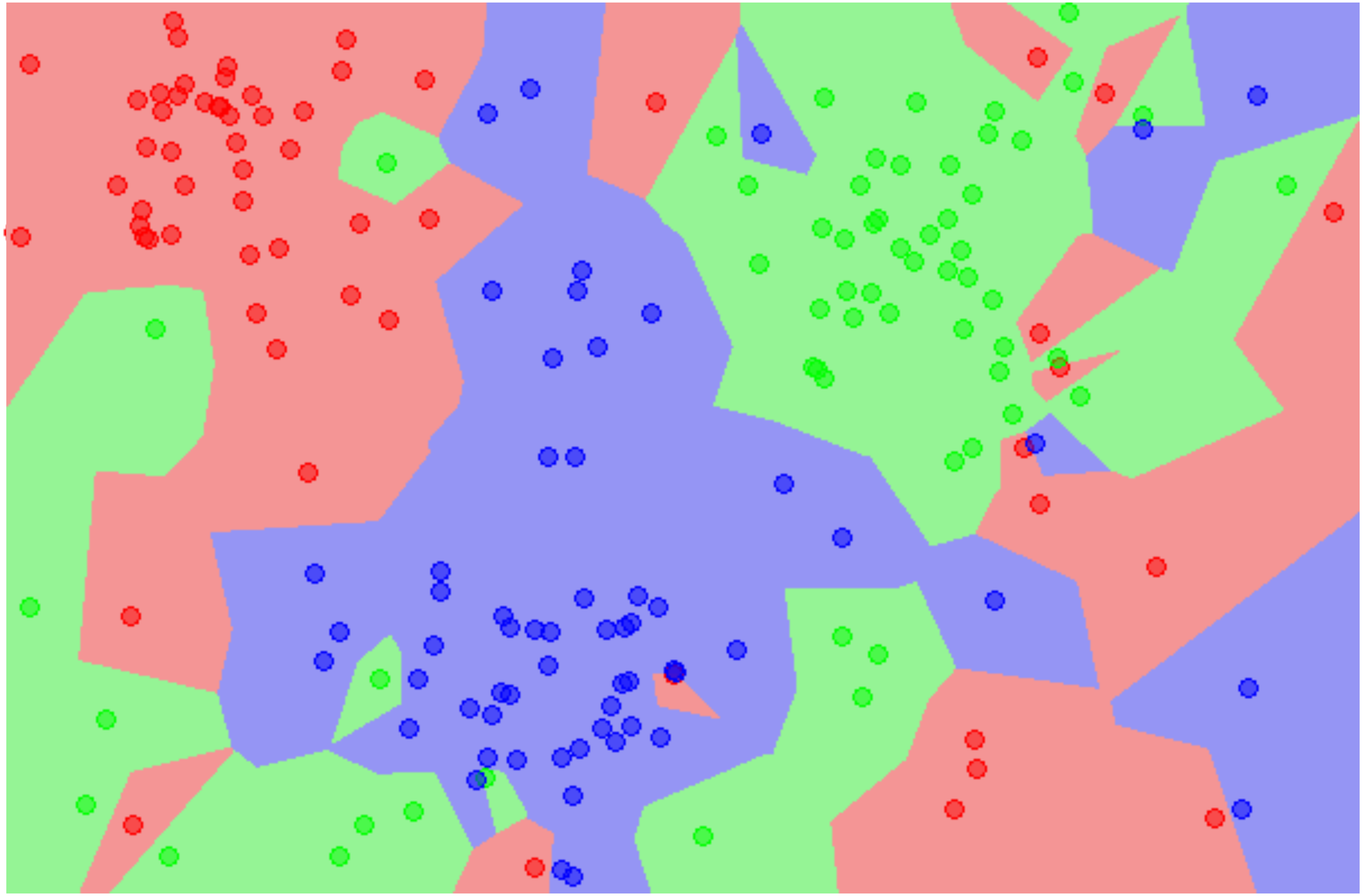
(a) 1-nearest neighbor (b) 2-nearest neighbor (c) 3-nearest neighbor

- **Compute distance to other training records**
- **Identify *K* nearest neighbors**
- **Take majority vote**

Training data for NN classifier (in R^2)



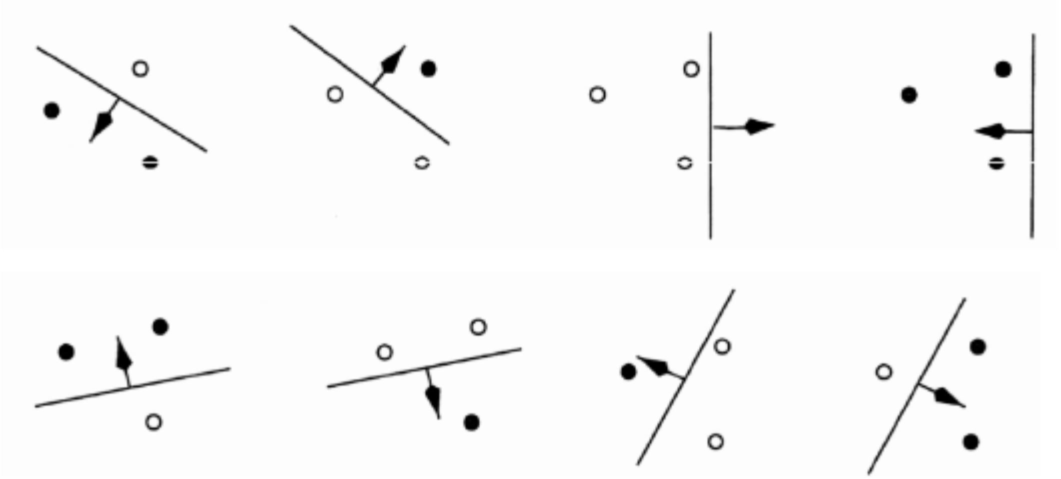
1-nn classifier prediction (in R^2)



VC dimension of 1-nearest neighbor classifier?

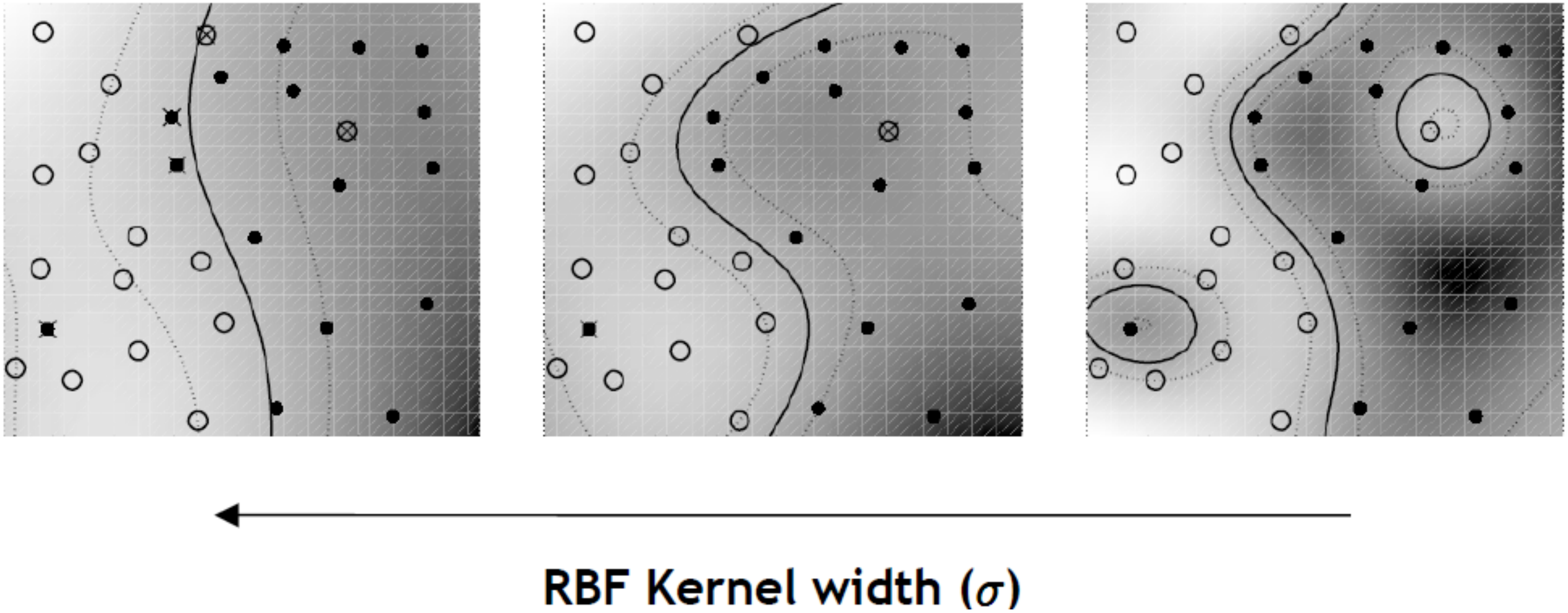
The VC dimension for the set of functions $\{f(\alpha)\}$ is defined as the maximum number of training points that can be shattered by $\{f(\alpha)\}$.

- VC dimension of N-dimensional linear classifier: N+1



- VC dimension of 1-NN: infinite

Large margins for nonlinear classifiers

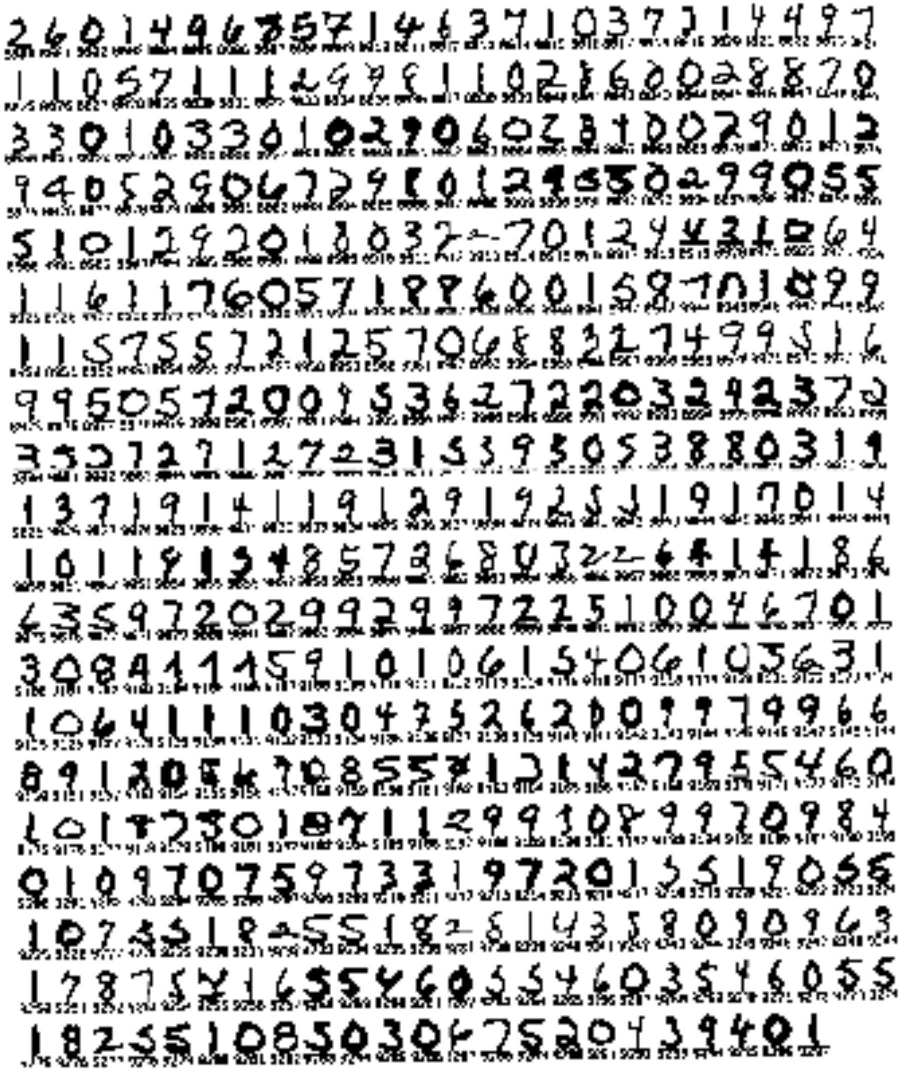


Margin size: determined by both σ and regularizer

We can slide between a linear and a Nearest-Neighbor classifier!

Guyon & Vapnik, 1995

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



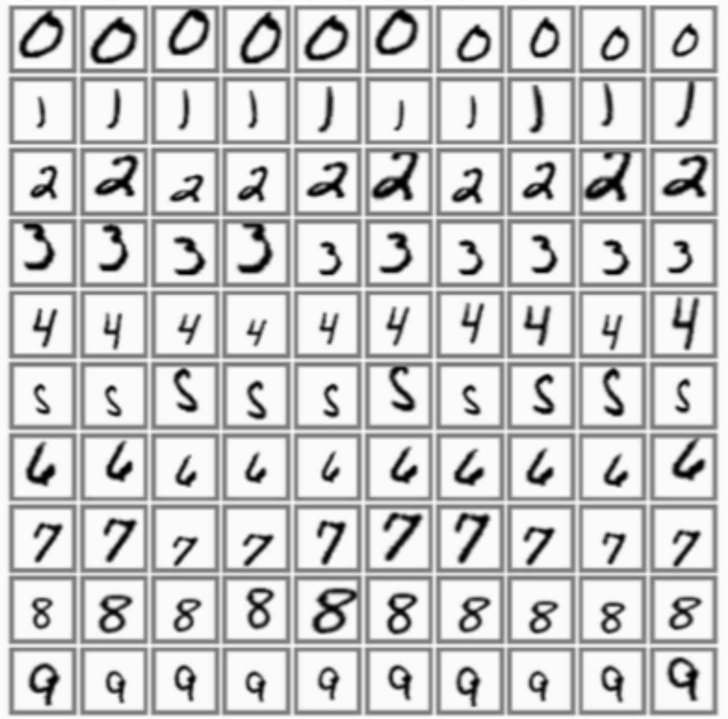
Application: Handwritten digit recognition

- 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

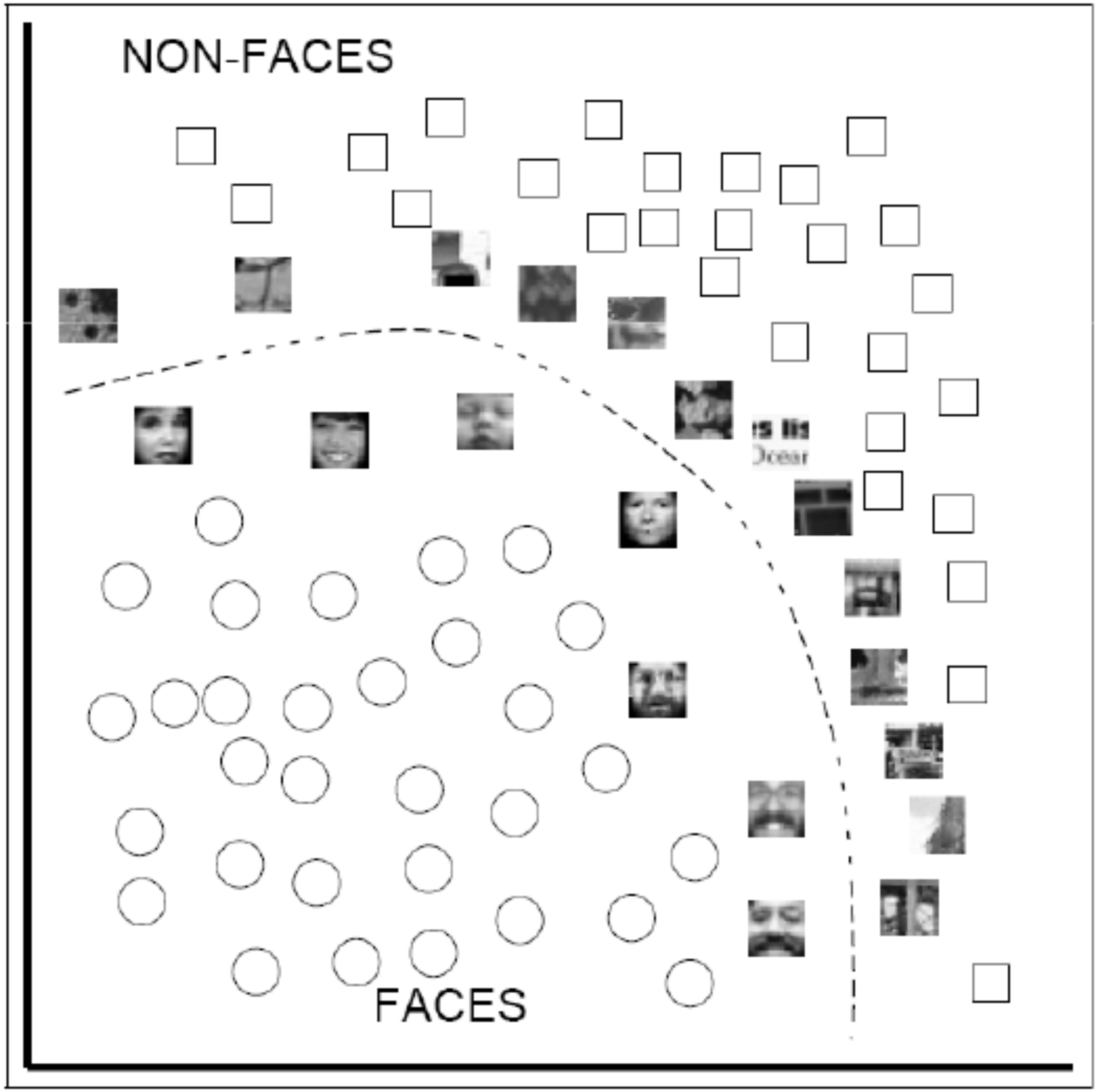
$$f(\mathbf{x}) = \max_k f_k(\mathbf{x})$$



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)

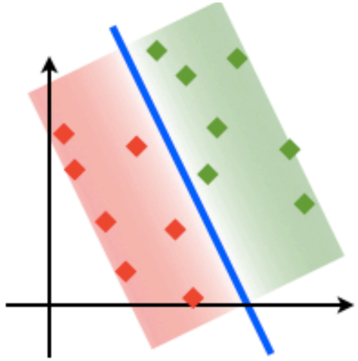


Linear vs. Nonlinear

Linear SVM

✓ fast

✗ restrictive

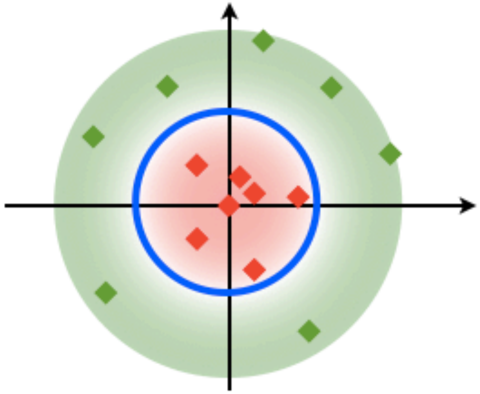


$$F(\mathbf{x}) = \langle \mathbf{w}, \mathbf{x} \rangle$$

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

Other kernels

- From http://www.kernel-methods.net/kernels.html

Kernel Functions Described in the Book:
Definition 9.1 Polynomial kernel 286
Computation 9.6 All-subsets kernel 289
Computation 9.8 Gaussian kernel 290
Computation 9.12 ANOVA kernel 293
Computation 9.18 Alternative recursion for ANOVA kernel 296
Computation 9.24 General graph kernels 301
Definition 9.33 Exponential difiusion kernel 307
Definition 9.34 von Neumann difiusion kernel 307
Computation 9.35 Evaluating difiusion kernels 308
Computation 9.46 Evaluating randomised kernels 315
Definition 9.37 Intersection kernel 309
Definition 9.38 Union-complement kernel 310
Remark 9.40 Agreement kernel 310
Section 9.6 Kernels on real numbers 311
Remark 9.42 Spline kernels 313
Definition 9.43 Derived subsets kernel 313
Definition 10.5 Vector space kernel 325
Computation 10.8 Latent semantic kernels 332
Definition 11.7 The p-spectrum kernel 342
Computation 11.10 The p-spectrum recursion 343
Remark 11.13 Blended spectrum kernel 344
Computation 11.17 All-subsequences kernel 347
Computation 11.24 Fixed length subsequences kernel 352
Computation 11.33 Naive recursion for gap-weighted subsequences kernel 358
Computation 11.36 Gap-weighted subsequences kernel 360
Computation 11.45 Trie-based string kernels 367
Algorithm 9.14 ANOVA kernel 294
Algorithm 9.25 Simple graph kernels 302
Algorithm 11.20 All�non-contiguous subsequences kernel 350
Algorithm 11.25 Fixed length subsequences kernel 352
Algorithm 11.38 Gap-weighted subsequences kernel 361
Algorithm 11.40 Character weighting string kernel 364
Algorithm 11.41 Soft matching string kernel 365
Algorithm 11.42 Gap number weighting string kernel 366
Algorithm 11.46 Trie-based p-spectrum kernel 368
Algorithm 11.51 Trie-based mismatch kernel 371
Algorithm 11.54 Trie-based restricted gap-weighted kernel 374
Algorithm 11.62 Co-rooted subtree kernel 380
Algorithm 11.65 All-subtree kernel 383
Algorithm 12.8 Fixed length HMM kernel 401
Algorithm 12.14 Pair HMM kernel 407
Algorithm 12.17 Hidden tree model kernel 411
Algorithm 12.34 Fixed length Markov model Fisher kernel 427

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

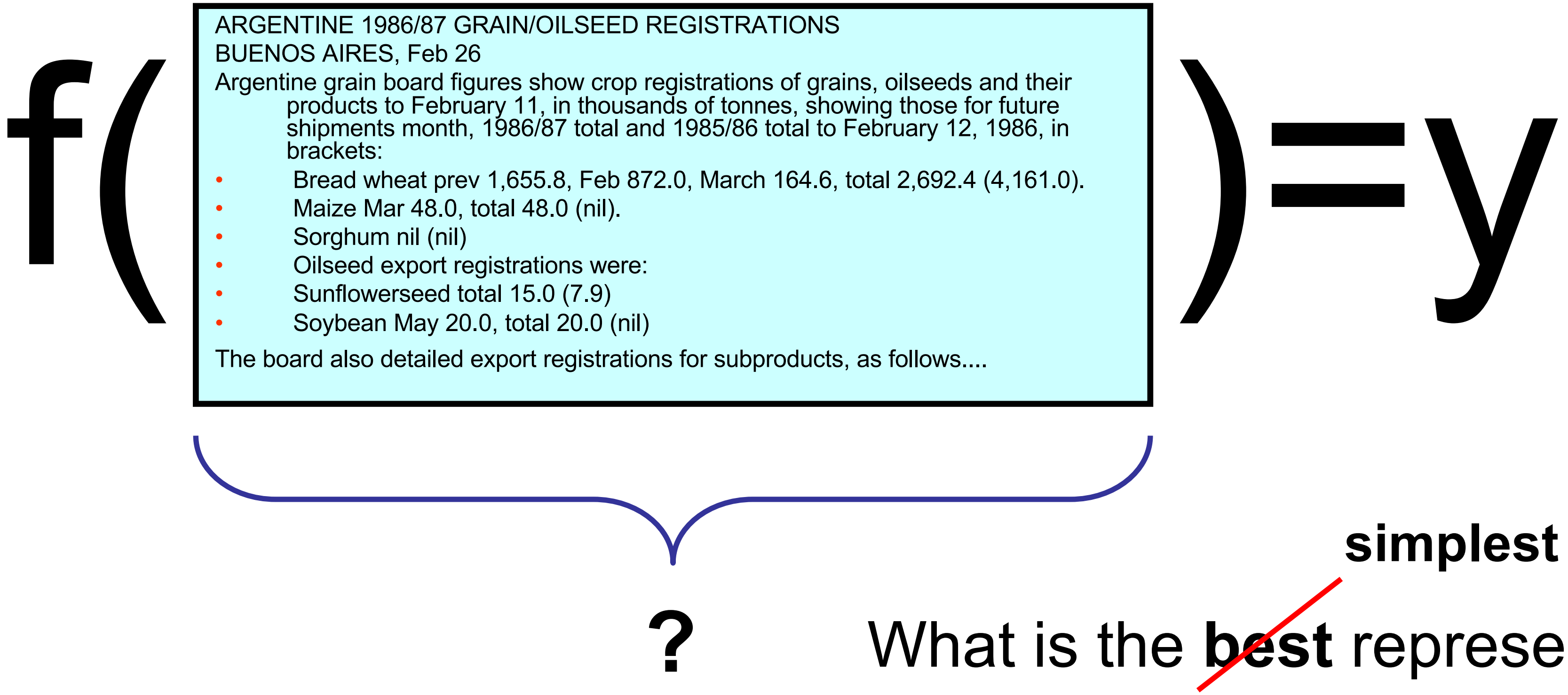
Argentine grain board figures show crop registrations of grains, oilseeds and their products to February 11, in thousands of tonnes, showing those for future shipments month, 1986/87 total and 1985/86 total to February 12, 1986, in brackets:

- 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



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....

What is the ~~**best**~~ **simplest useful** 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

xxxxxxxxxxxxxxxxxxxxx **GRAIN/OILSEED** xxxxxxxxxxxxx

xxxxxxxxxxxxxxxxxxxxxxx

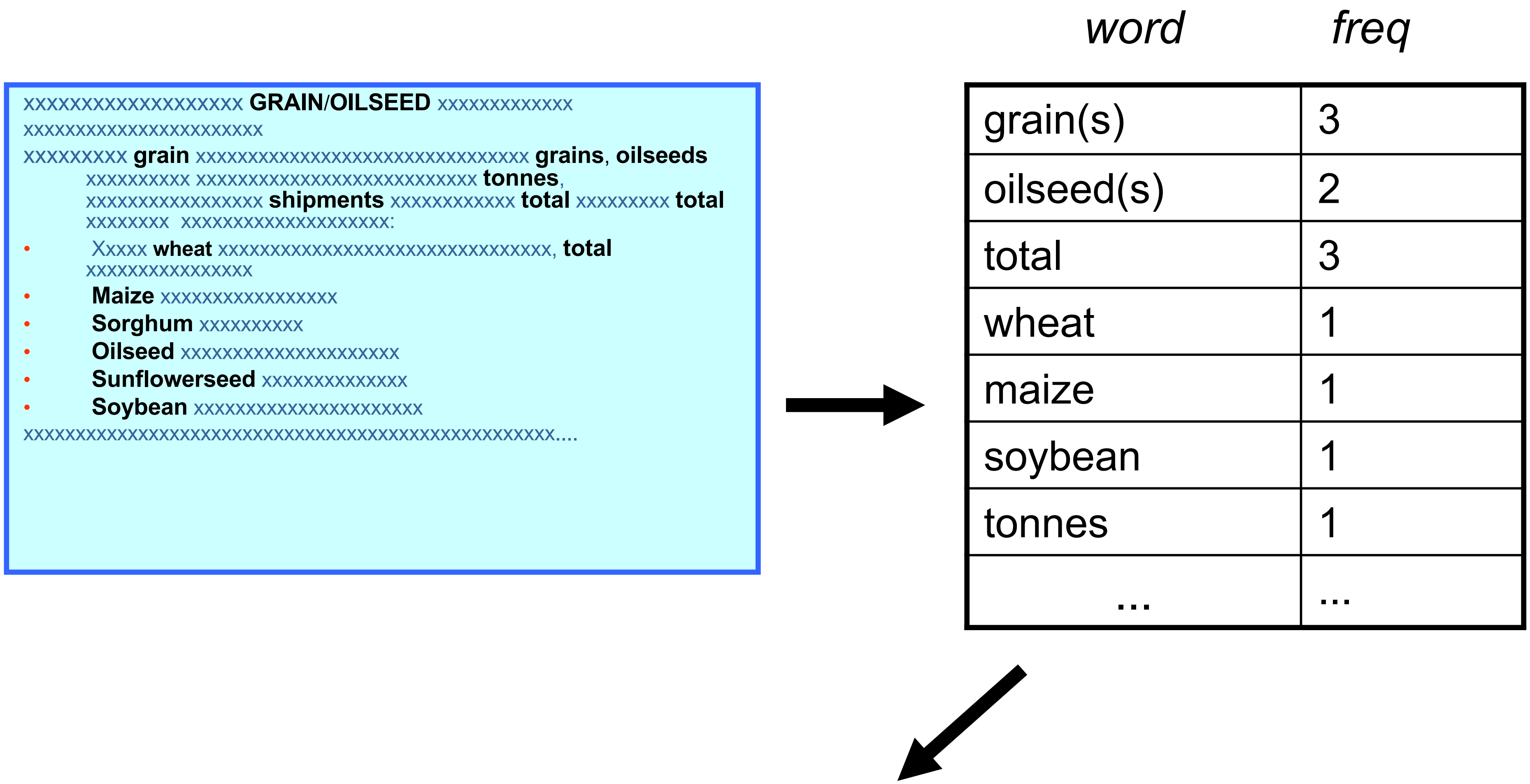
xxxxxxxxxx **grain** xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx **grains**, **oilseeds** xxxxxxxxxx xxxxxxxxxxxxxxxxxxxxxxxxxxx **tonnes**, xxxxxxxxxxxxxxxxx **shipments** xxxxxxxxxxxx **total** xxxxxxxxx **total** xxxxxxxx xxxxxxxxxxxxxxxxxxxx:

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

xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx....

→ Categories: **grain**, **wheat**

Bag of words representation



xxxxxxxxxxxxxxxxxxx **GRAIN/OILSEED** xxxxxxxxxxxx
xxxxxxxxxxxxxxxxxxxxx
xxxxxxxxx **grain** xxxxxxxxxxxxxxxxxxxxxxxxxxxxx **grains**, **oilseeds**
xxxxxxxxx xxxxxxxxxxxxxxxxxxxxxxxxx **tonnes**,
xxxxxxxxxxxxxxx **shipments** xxxxxxxxxxx **total** xxxxxxxx **total**
xxxxxxx xxxxxxxxxxxxxxxxxx:

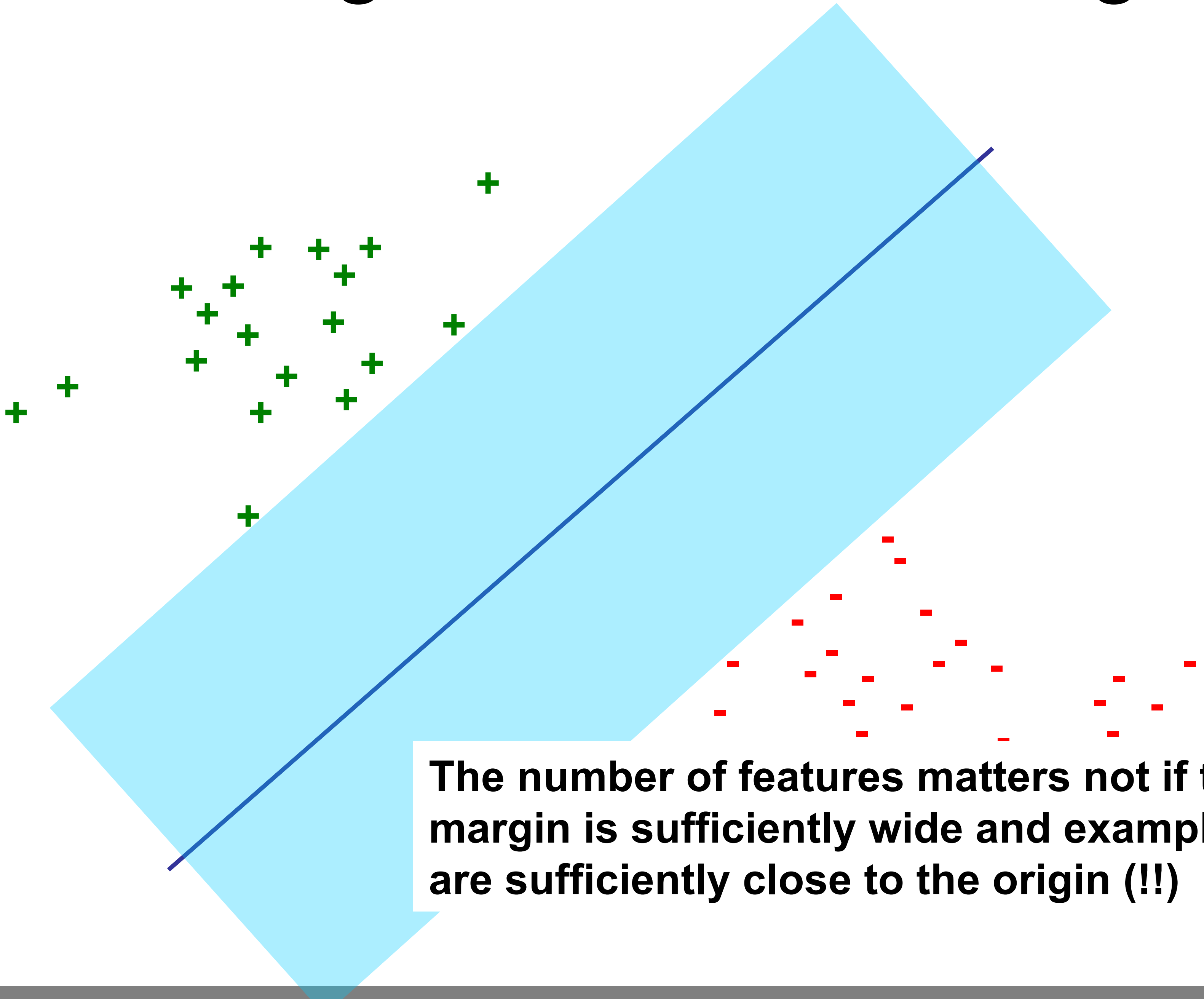
- Xxxxx **wheat** xxxxxxxxxxxxxxxxxxxxxxxxxxxxx, **total** xxxxxxxxxxxxxx
- **Maize** xxxxxxxxxxxxxxx
- **Sorghum** xxxxxxxxx
- **Oilseed** xxxxxxxxxxxxxxxxxxx
- **Sunflowerseed** xxxxxxxxxxxxx
- **Soybean** xxxxxxxxxxxxxxxxxxxx

xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx....

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

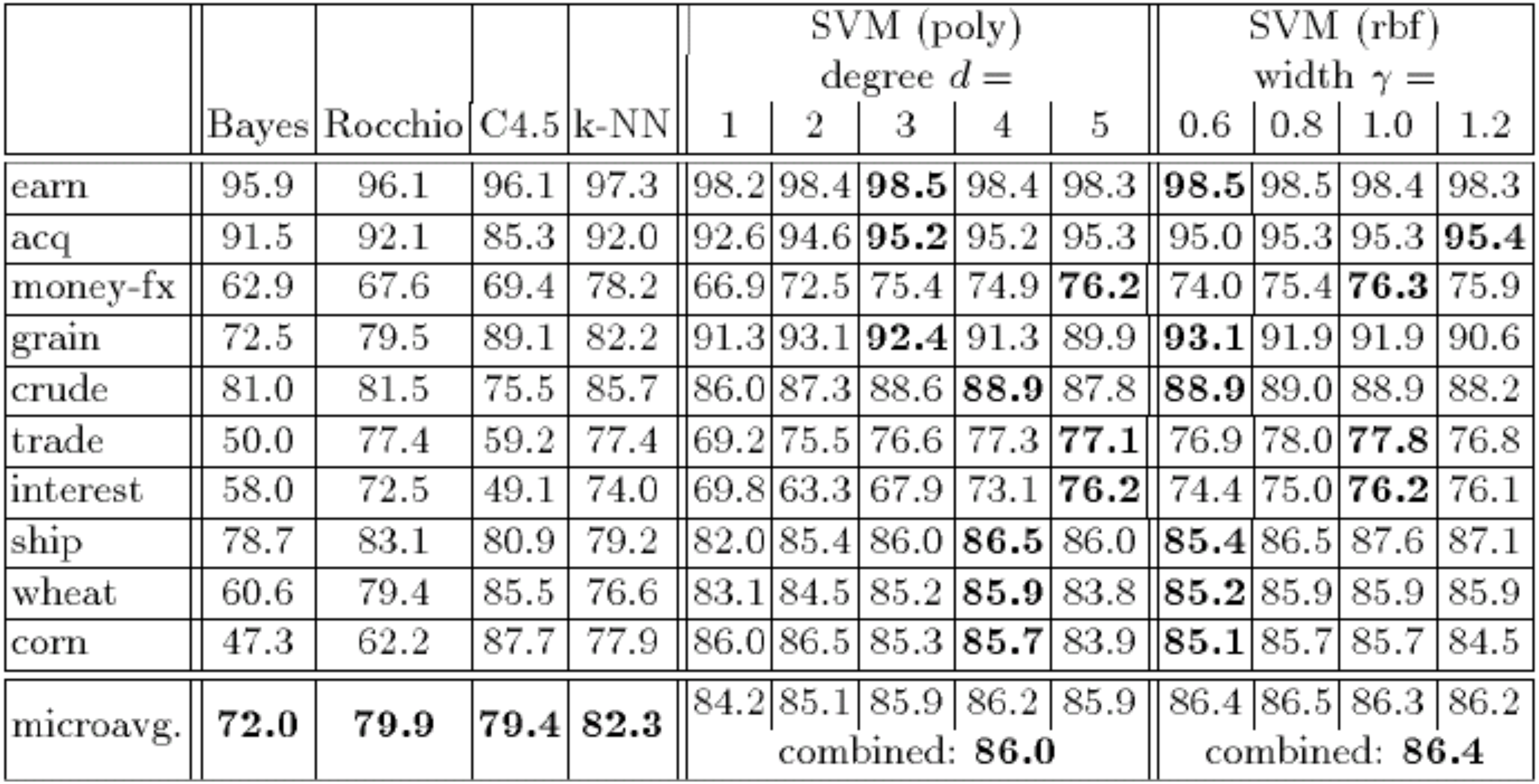
Categories: **grain**, **wheat**

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**			

Sequence Data versus Structure and Function

Sequences for four chains of human hemoglobin

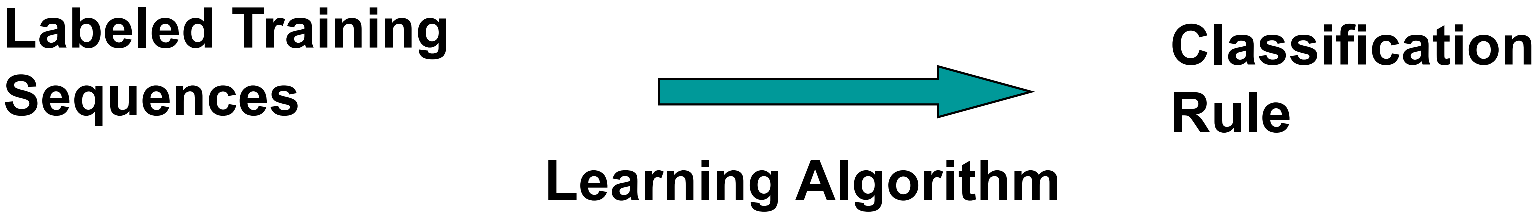
```
>1A3N:A HEMOGLOBIN
VLSPADKTNVKAAWGKVGAHAGEYGAEALERMFLSFPTTKTYFPHFDLSHGSAQVKGHGK
KVADALTNAVAHVDDMPNALSALSDLHAHKLRVDPVNFKLLSHCLLVTLAAHLPAEFTPA
VHASLDKFLASVSTVLTSKYR
>1A3N:B HEMOGLOBIN
VHLTPEEKSAVTALWGKVNVDEVGGEALGRLLVVYPWTQRFFESFGDLSTPDAVMGNPKV
KAHGKKVLGAFSDGLAHLDNLKGTFATLSELHCDKLHVDPENFRLLGNVLVCVLAHHFGK
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
- Use *supervised learning* approach to *train* a classifier



Labeled Training Sequences

Learning Algorithm

Classification Rule

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

Appendix

- Primal and Dual form of SVMs: the full story

References:

S. Boyd and L. Vandeberghe: Convex Optimization (textbook)

C. Burges: A tutorial on SVMs for pattern recognition