Support Vector Machines

CS 760@UW-Madison

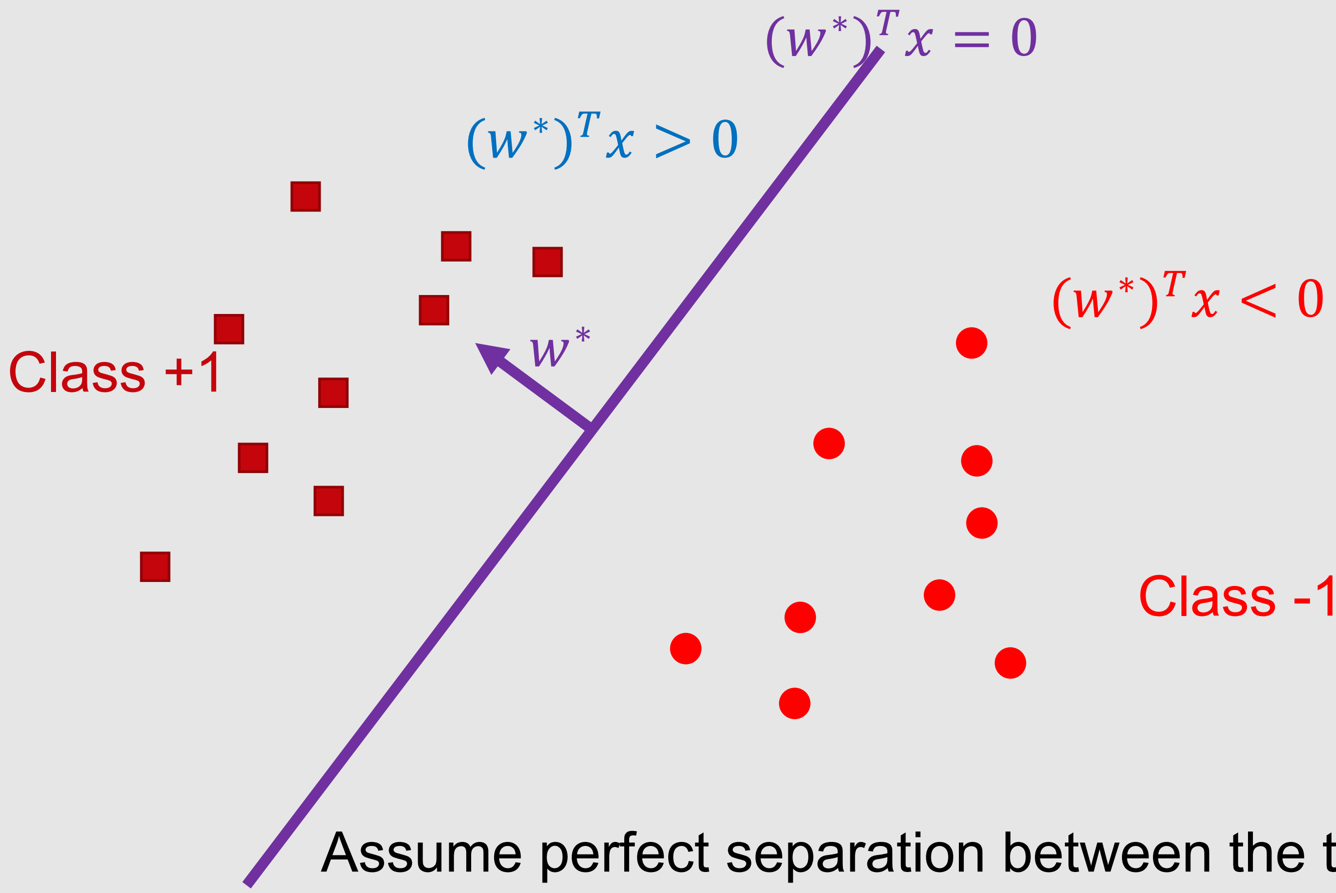
Goals for Part 1

you should understand the following concepts

- the margin
- the linear support vector machine
- the primal and dual formulations of SVM learning
- support vectors
- VC-dimension and maximizing the margin

Motivation

Linear classification



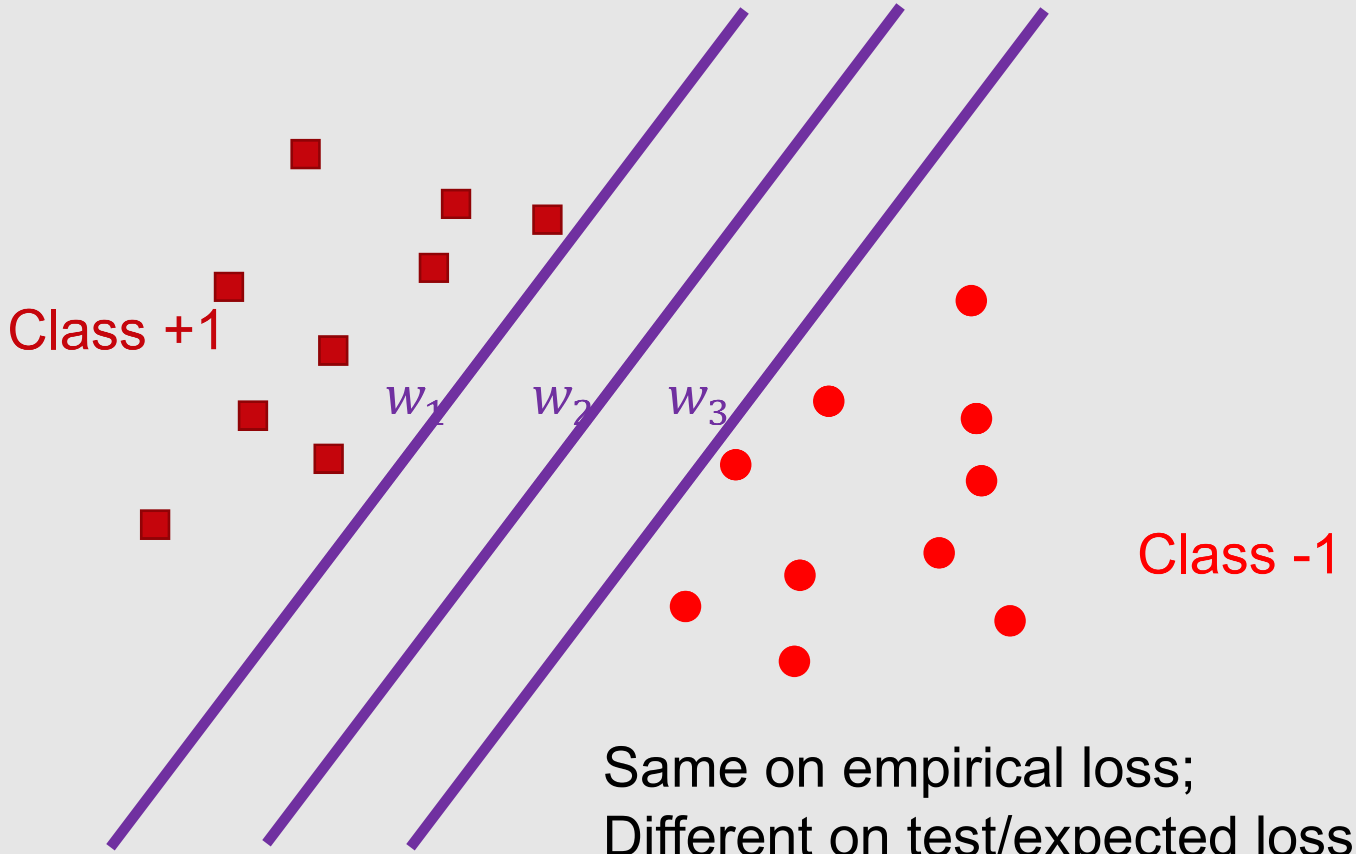
Assume perfect separation between the two classes

Attempt

- Given training data $\{(x_i, y_i): 1 \leq i \leq n\}$ i.i.d. from distribution D
- Hypothesis $y = \text{sign}(f_w(x)) = \text{sign}(w^T x)$
  - $y = +1$ if $w^T x > 0$
  - $y = -1$ if $w^T x < 0$

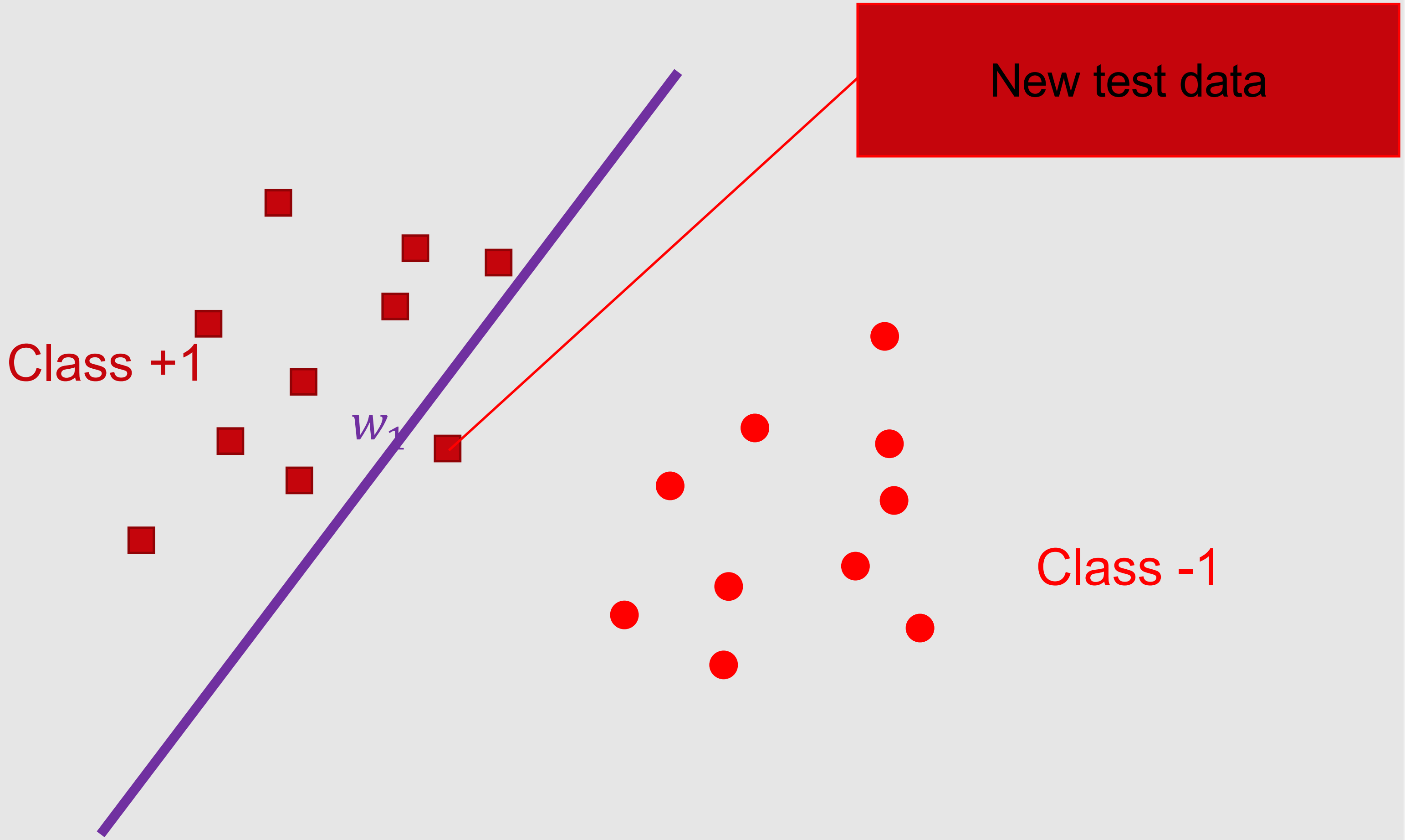
- Let's assume that we can optimize to find w

Multiple optimal solutions?

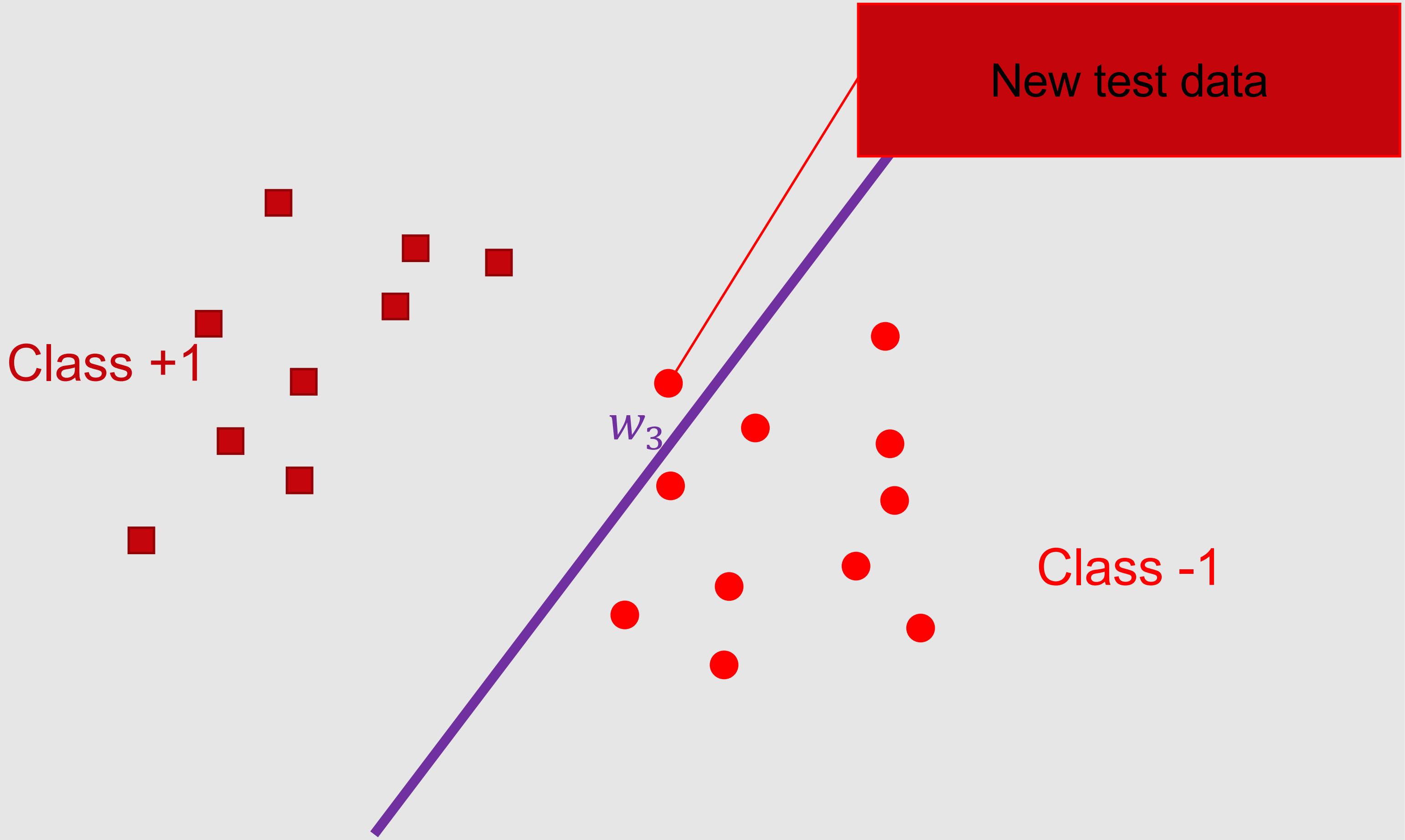


Same on empirical loss;
Different on test/expected loss

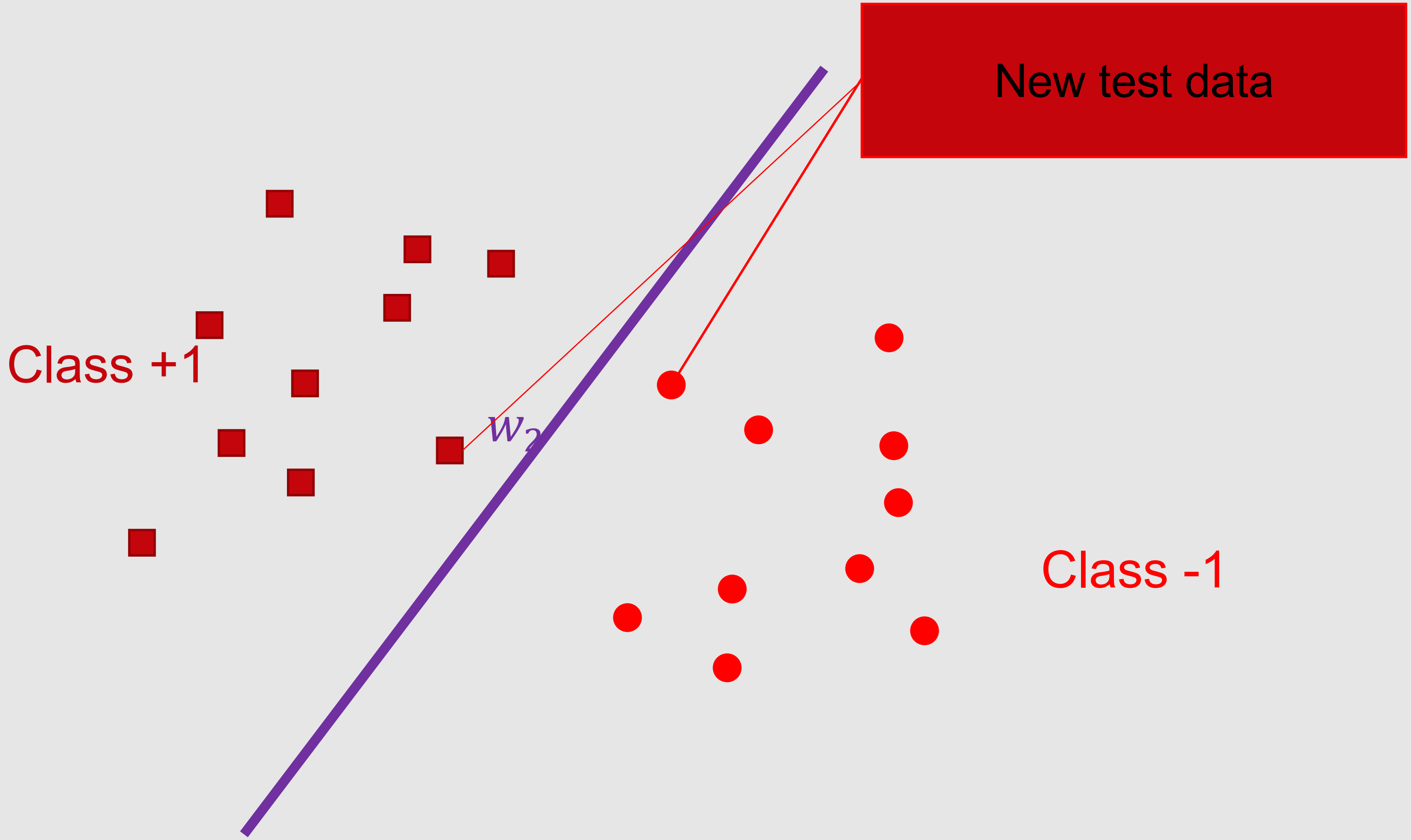
What about w_1?



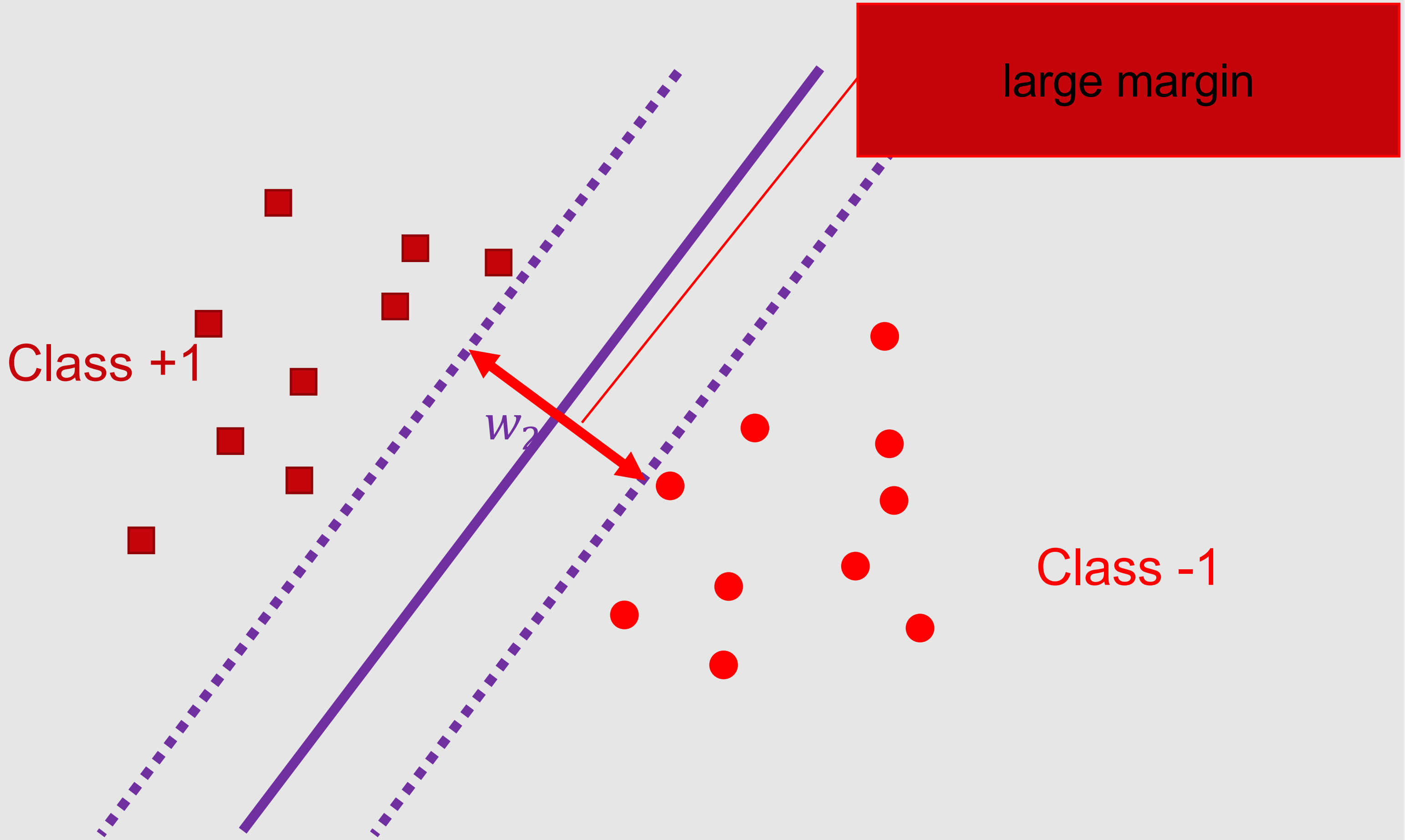
What about w_3?



Most confident: w_2



Intuition: margin



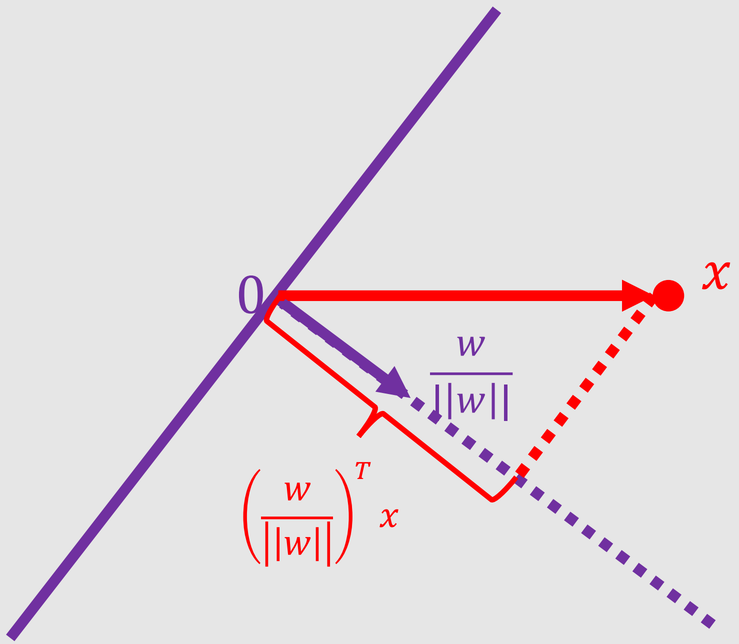
Margin

Margin

- Lemma 1: x has distance $\frac{|f_w(x)|}{||w||}$ to the hyperplane $f_w(x) = w^T x = 0$

Proof:

- w is orthogonal to the hyperplane
- The unit direction is $\frac{w}{||w||}$
- The projection of x is $\left(\frac{w}{||w||}\right)^T x = \frac{f_w(x)}{||w||}$



Margin: with bias

- Claim 1: w is orthogonal to the hyperplane $f_{w,b}(x) = w^T x + b = 0$

Proof:

- pick any x_1 and x_2 on the hyperplane
- $w^T x_1 + b = 0$
- $w^T x_2 + b = 0$

- So $w^T (x_1 - x_2) = 0$

Margin: with bias

- Claim 2: 0 has distance $\frac{|b|}{||w||}$ to the hyperplane $w^T x + b = 0$

Proof:

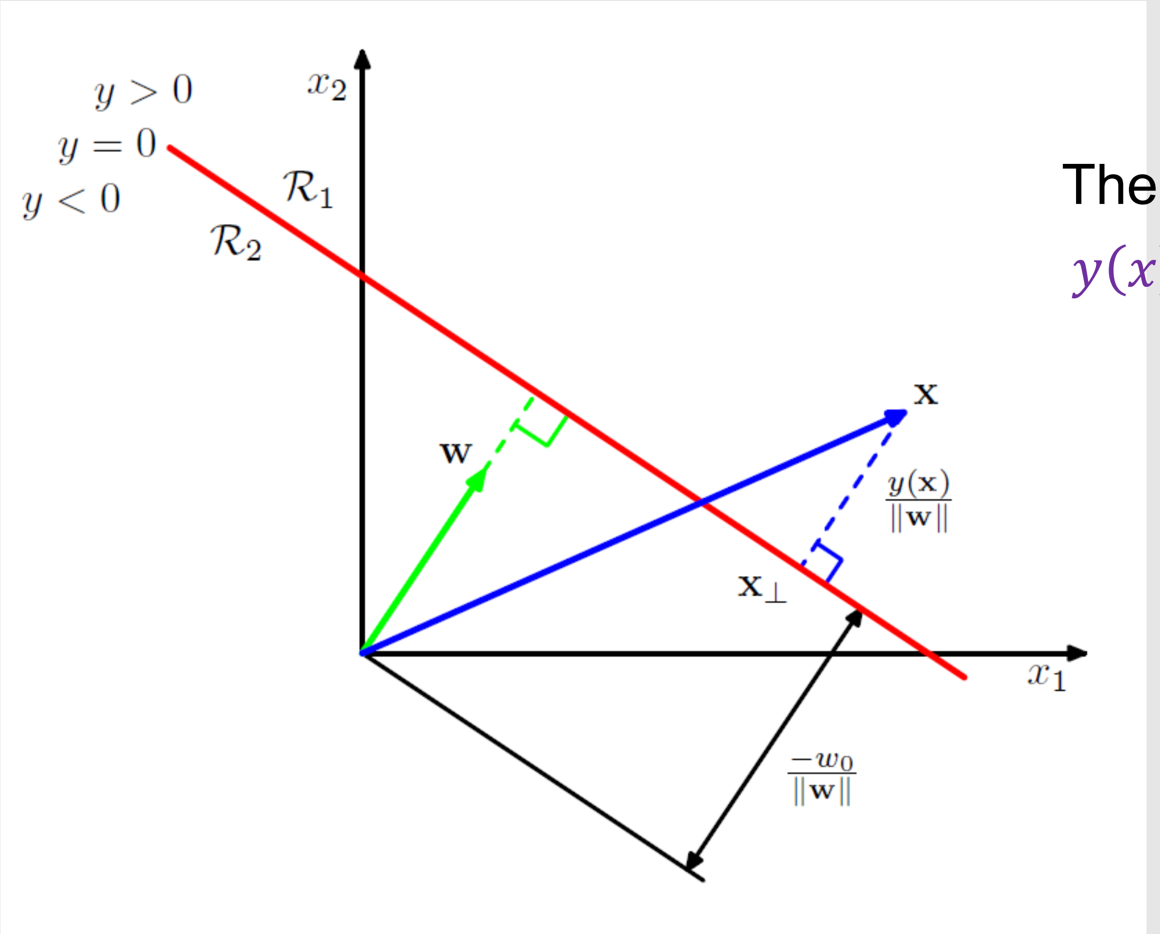
- pick any x_1 the hyperplane
- Project x_1 to the unit direction $\frac{w}{||w||}$ to get the distance
- $\left(\frac{w}{||w||}\right)^T x_1 = \frac{-b}{||w||}$ since $w^T x_1 + b = 0$

Margin: with bias

- Lemma 2: x has distance $\frac{|f_{w,b}(x)|}{||w||}$ to the hyperplane $f_{w,b}(x) = w^T x + b = 0$

Proof:

- Let $x = x_{\perp} + r\frac{w}{||w||}$, then $|r|$ is the distance
- Multiply both sides by w^T and add b
- Left hand side: $w^T x + b = f_{w,b}(x)$
- Right hand side: $w^T x_{\perp} + r\frac{w^T w}{||w||} + b = 0 + r||w||$



The notation here is:

$$y(x) = w^T x + w_0$$

Figure from *Pattern Recognition and Machine Learning*, Bishop

Support Vector Machine (SVM)

SVM: objective

- Margin over all training data points:

$$\gamma = \min_i \frac{|f_{w,b}(x_i)|}{||w||}$$

- Since only want correct $f_{w,b}$, and recall $y_i \in \{+1, -1\}$, we have

$$\gamma = \min_i \frac{y_i f_{w,b}(x_i)}{||w||}$$

- If $f_{w,b}$ incorrect on some x_i, the margin is negative

SVM: objective

- Maximize margin over all training data points:

$$\max_{w,b} \gamma = \max_{w,b} \min_{i} \frac{y_i f_{w,b}(x_i)}{||w||} = \max_{w,b} \min_{i} \frac{y_i(w^T x_i + b)}{||w||}$$

- A bit complicated …

SVM: simplified objective

- Observation: when (w, b) scaled by a factor c, the margin unchanged

$$\frac{y_i(cw^T x_i + cb)}{||cw||} = \frac{y_i(w^T x_i + b)}{||w||}$$

- Let's consider a fixed scale such that

$$y_{i^*}(w^T x_{i^*} + b) = 1$$

where x_{i^*} is the point closest to the hyperplane

SVM: simplified objective

- Let's consider a fixed scale such that

$$y_{i^*}(w^T x_{i^*} + b) = 1$$

where x_{i^*} is the point closet to the hyperplane

- Now we have for all data

$$y_i(w^T x_i + b) \geq 1$$

and at least for one i the equality holds

- Then the margin is $\frac{1}{||w||}$

SVM: simplified objective

- Optimization simplified to

$$\min_{w,b} \frac{1}{2} \left|\left|w\right|\right|^2$$

$$y_i(w^T x_i + b) \geq 1, \forall i$$

- How to find the optimum $\hat{w}^*$?
- Solved by Lagrange multiplier method

Lagrange multiplier

Lagrangian

- Consider optimization problem:

$$\min_{w} f(w)$$

$$h_i(w) = 0, \forall 1 \leq i \leq l$$

- Lagrangian:

$$\mathcal{L}(w, \boldsymbol{\beta}) = f(w) + \sum_i \beta_i h_i(w)$$

where β_i's are called Lagrange multipliers

Lagrangian

- Consider optimization problem:

$$\min_{w} f(w)$$

$$h_i(w) = 0, \forall 1 \leq i \leq l$$

- Solved by setting derivatives of Lagrangian to 0

$$\frac{\partial \mathcal{L}}{\partial w_i} = 0; \quad \frac{\partial \mathcal{L}}{\partial \beta_i} = 0$$

Generalized Lagrangian

- Consider optimization problem:

$$\min_w f(w)$$

$$g_i(w) \leq 0, \forall 1 \leq i \leq k$$

$$h_j(w) = 0, \forall 1 \leq j \leq l$$

- Generalized Lagrangian:

$$\mathcal{L}(w, \boldsymbol{\alpha}, \boldsymbol{\beta}) = f(w) + \sum_i \alpha_i g_i(w) + \sum_j \beta_j h_j(w)$$

where α_i, β_j's are called Lagrange multipliers

Generalized Lagrangian

- Consider the quantity:

$$\theta_P(w) \coloneqq \max_{\boldsymbol{\alpha},\boldsymbol{\beta}:\alpha_i \geq 0} \mathcal{L}(w, \boldsymbol{\alpha}, \boldsymbol{\beta})$$

- Why?

$$\theta_P(w) = \begin{cases} f(w), & \text{if } w \text{ satisfies all the constraints} \\ +\infty, & \text{if } w \text{ does not satisfy the constraints} \end{cases}$$

- So minimizing $f(w)$ is the same as minimizing $\theta_P(w)$

$$\min_w f(w) = \min_w \theta_P(w) = \min_w \max_{\boldsymbol{\alpha},\boldsymbol{\beta}:\alpha_i \geq 0} \mathcal{L}(w, \boldsymbol{\alpha}, \boldsymbol{\beta})$$

Lagrange duality

- The primal problem

$$p^* := \min_w f(w) = \min_w \max_{\boldsymbol{\alpha},\boldsymbol{\beta}:\alpha_i \geq 0} \mathcal{L}(w, \boldsymbol{\alpha}, \boldsymbol{\beta})$$

- The dual problem

$$d^* := \max_{\boldsymbol{\alpha},\boldsymbol{\beta}:\alpha_i \geq 0} \min_w \mathcal{L}(w, \boldsymbol{\alpha}, \boldsymbol{\beta})$$

- Always true:

$$d^* \leq p^*$$

Lagrange duality

- The primal problem

$$p^* := \min_w f(w) = \min_w \max_{\boldsymbol{\alpha},\boldsymbol{\beta}:\alpha_i \geq 0} \mathcal{L}(w, \boldsymbol{\alpha}, \boldsymbol{\beta})$$

- The dual problem

$$d^* := \max_{\boldsymbol{\alpha},\boldsymbol{\beta}:\alpha_i \geq 0} \min_w \mathcal{L}(w, \boldsymbol{\alpha}, \boldsymbol{\beta})$$

- Interesting case: when do we have

$$d^* = p^*?$$

Lagrange duality

- Theorem: under proper conditions, there exists $(w^*, \boldsymbol{\alpha}^*, \boldsymbol{\beta}^*)$ such that

$$d^* = \mathcal{L}(w^*, \boldsymbol{\alpha}^*, \boldsymbol{\beta}^*) = p^*$$

Moreover, $(w^*, \boldsymbol{\alpha}^*, \boldsymbol{\beta}^*)$ satisfy Karush-Kuhn-Tucker (KKT) conditions:

$$\frac{\partial \mathcal{L}}{\partial w_i} = 0, \qquad \alpha_i g_i(w) = 0$$

$$g_i(w) \leq 0, \; h_j(w) = 0, \qquad \alpha_i \geq 0$$

Lagrange duality

- Theorem: under proper conditions, there exists $(w^*, \boldsymbol{\alpha}^*, \boldsymbol{\beta}^*)$ such that

$$d^* = \mathcal{L}(w^*, \boldsymbol{\alpha}^*, \boldsymbol{\beta}^*) = p^*$$

dual complementarity

Moreover, $(w^*, \boldsymbol{\alpha}^*, \boldsymbol{\beta}^*)$ satisfy Karush-Kuhn-Tucker (KKT) conditions:

$$\frac{\partial \mathcal{L}}{\partial w_i} = 0, \qquad \alpha_i g_i(w) = 0$$

$$g_i(w) \leq 0, \; h_j(w) = 0, \qquad \alpha_i \geq 0$$

Lagrange duality

- Theorem: under proper conditions, there exists $(w^*, \boldsymbol{\alpha}^*, \boldsymbol{\beta}^*)$ such that

$$d^* = \mathcal{L}(w^*, \boldsymbol{\alpha}^*, \boldsymbol{\beta}^*) = p^*$$

primal constraints

dual constraints

satisfy Karush-Kuhn-Tu

conditions:

$$\frac{\partial \mathcal{L}}{\partial w_i} = 0, \qquad \alpha_i g_i(w) = 0$$

$$g_i(w) \leq 0, \; h_j(w) = 0, \qquad \alpha_i \geq 0$$

Lagrange duality

- What are the proper conditions?
- A set of conditions (Slater conditions):
  - f, g_i convex, h_j affine, and exists w satisfying all $g_i(w) < 0$

- There exist other sets of conditions
  - Check textbooks, e.g., Convex Optimization by Boyd and Vandenberghe

SVM: optimization

SVM: optimization

- Optimization (Quadratic Programming):

$$\min_{w,b} \frac{1}{2}\left|\left|w\right|\right|^2$$

$$y_i(w^T x_i + b) \geq 1, \forall i$$

- Generalized Lagrangian:

$$\mathcal{L}(w, b, \boldsymbol{\alpha}) = \frac{1}{2}\left|\left|w\right|\right|^2 - \sum_i \alpha_i [y_i(w^T x_i + b) - 1]$$

where $\boldsymbol{\alpha}$ is the Lagrange multiplier

SVM: optimization

- KKT conditions:

$$\frac{\partial \mathcal{L}}{\partial w} = 0, \rightarrow w = \sum_i \alpha_i y_i x_i \quad (1)$$

$$\frac{\partial \mathcal{L}}{\partial b} = 0, \rightarrow 0 = \sum_i \alpha_i y_i \quad (2)$$

- Plug into $\mathcal{L}$:

$$\mathcal{L}(w, b, \boldsymbol{\alpha}) = \sum_i \alpha_i - \frac{1}{2} \sum_{ij} \alpha_i \alpha_j y_i y_j x_i^T x_j \quad (3)$$

combined with $0 = \sum_i \alpha_i y_i \, , \alpha_i \geq 0$

SVM: optimization

Only depend on inner products

- Reduces to dual problem:

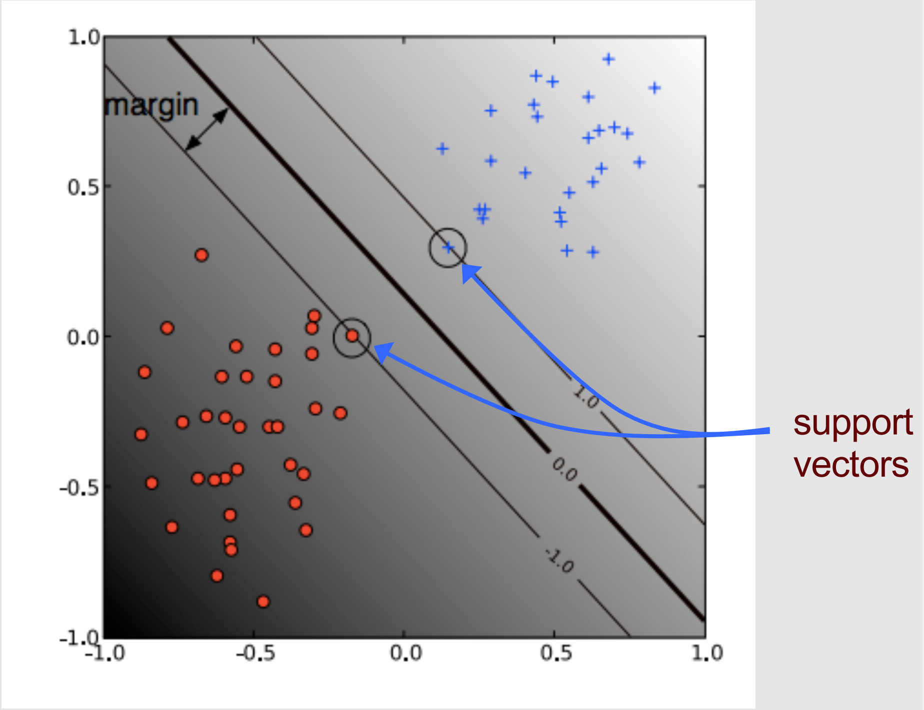
$$\mathcal{L}(w, b, \boldsymbol{\alpha}) = \sum_i \alpha_i - \frac{1}{2} \sum_{ij} \alpha_i \alpha_j y_i y_j x_i^T x_j$$

$$\sum_i \alpha_i y_i = 0, \alpha_i \geq 0$$

- Since $w = \sum_i \alpha_i y_i x_i$, we have $w^T x + b = \sum_i \alpha_i y_i x_i^T x + b$

Support Vectors

- final solution is a sparse linear combination of the training instances
- those instances with $\alpha_i > 0$ are called *support vectors*
  - they lie on the margin boundary
- solution NOT changed if delete the instances with $\alpha_i = 0$



Learning theory justification

$$error(h) \leq error_D(h) + \sqrt{\frac{VC\left(\log\frac{2m}{VC}+1\right)+\log\frac{4}{\delta}}{m}}$$

error on true distribution

training set error

VC: VC-dimension of hypothesis class

- Vapnik showed a connection between the margin and VC dimension

$$VC \leq \frac{4R^2}{margin_D(h)}$$

constant dependent on training data

- thus to minimize the VC dimension (and to improve the error bound) ➔ maximize the margin

Goals for Part 2

you should understand the following concepts

- soft margin SVM
- support vector regression
- the kernel trick
- polynomial kernel
- Gaussian/RBF kernel
- valid kernels and Mercer's theorem
- kernels and neural networks

Variants: soft-margin and SVR

Hard-margin SVM

- Optimization (Quadratic Programming):

$$\min_{w,b} \frac{1}{2}\left|\left|w\right|\right|^2$$

$$y_i(w^T x_i + b) \geq 1, \forall i$$

Soft-margin SVM [Cortes & Vapnik, *Machine Learning* 1995]

- if the training instances are not linearly separable, the previous formulation will fail
- we can adjust our approach by using *slack variables* (denoted by ζ_i) to tolerate errors

$$\min_{w,b,\zeta_i} \frac{1}{2}||w||^2 + C\sum_i \zeta_i$$

$$y_i(w^T x_i + b) \geq 1 - \zeta_i, \zeta_i \geq 0, \forall i$$

- C determines the relative importance of maximizing margin vs. minimizing slack

The effect of C in soft-margin SVM

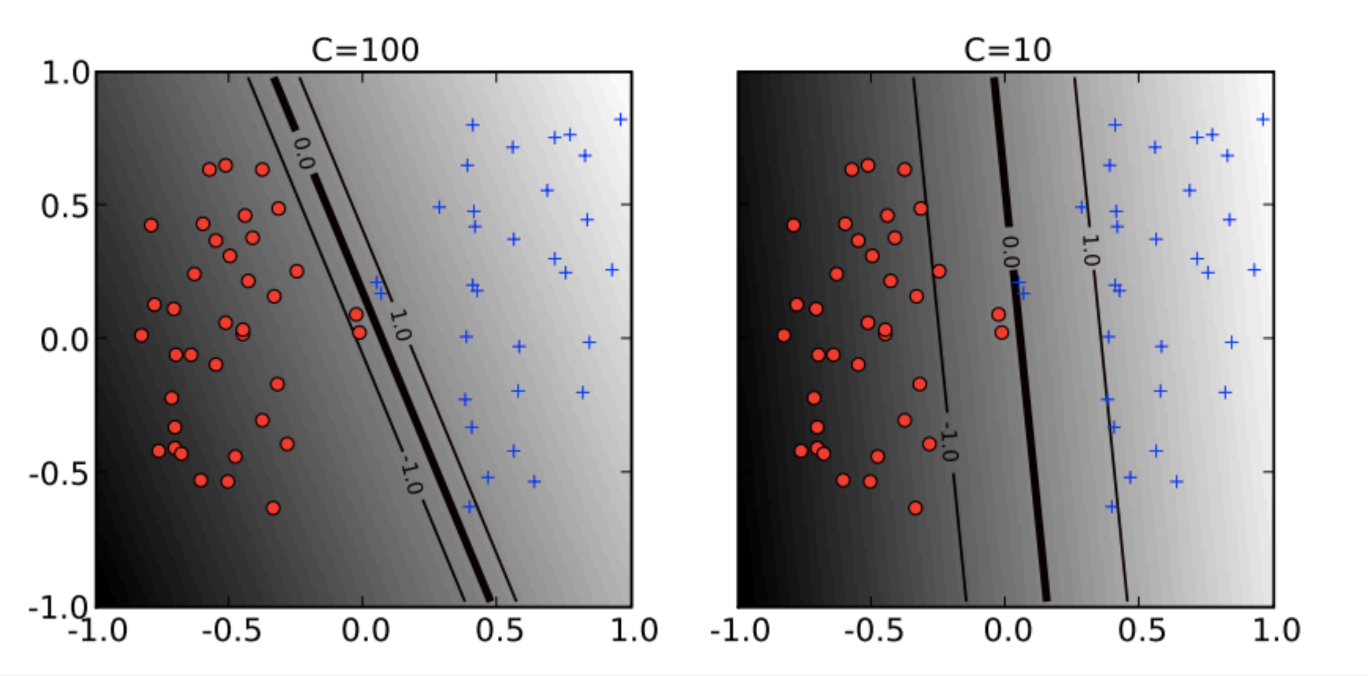
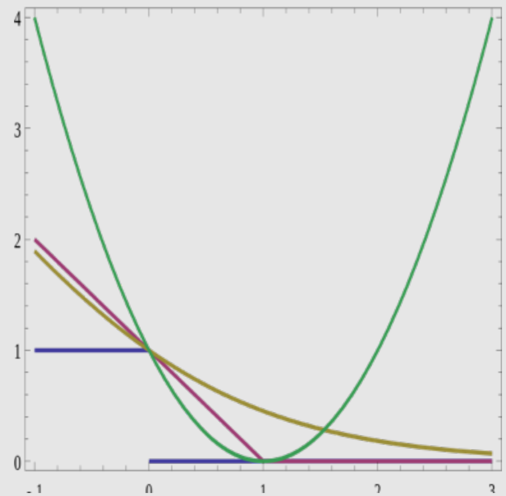


Figure from Ben-Hur & Weston,
Methods in Molecular Biology 2010

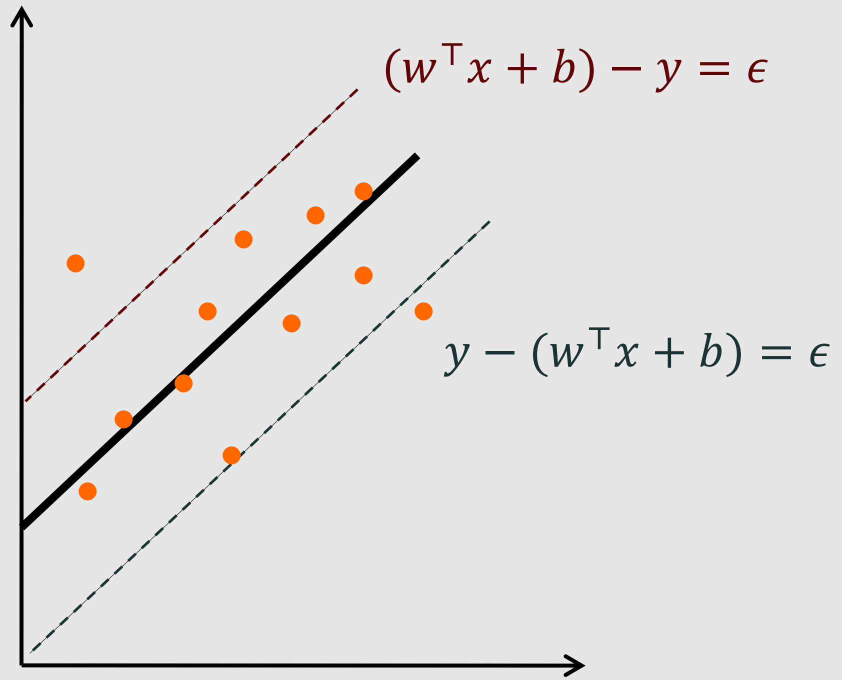
Hinge loss

- when we covered neural nets, we talked about minimizing squared loss and cross-entropy loss
- SVMs minimize *hinge loss*



Support Vector Regression

- the SVM idea can also be applied in regression tasks
- an ϵ-insensitive error function specifies that a training instance is well explained if the model's prediction is within ϵ of y_i



Support Vector Regression

- Regression using *slack variables* (denoted by ζ_i, ξ_i) to tolerate errors

$$\min_{w,b,\zeta_i,\xi_i} \frac{1}{2}||w||^2 + C\sum_i \zeta_i + \xi_i$$

$$(w^T x_i + b) - y_i \leq \epsilon + \zeta_i,$$
$$y_i - (w^T x_i + b) \leq \epsilon + \xi_i,$$
$$\zeta_i, \xi_i \geq 0.$$

slack variables allow predictions for some training instances to be off by more than ϵ

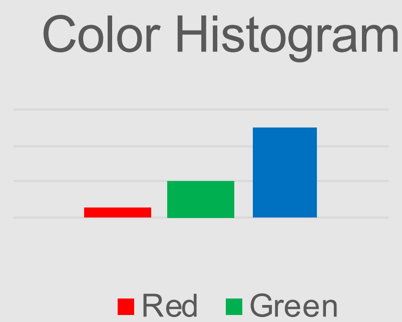
Kernel methods

Features

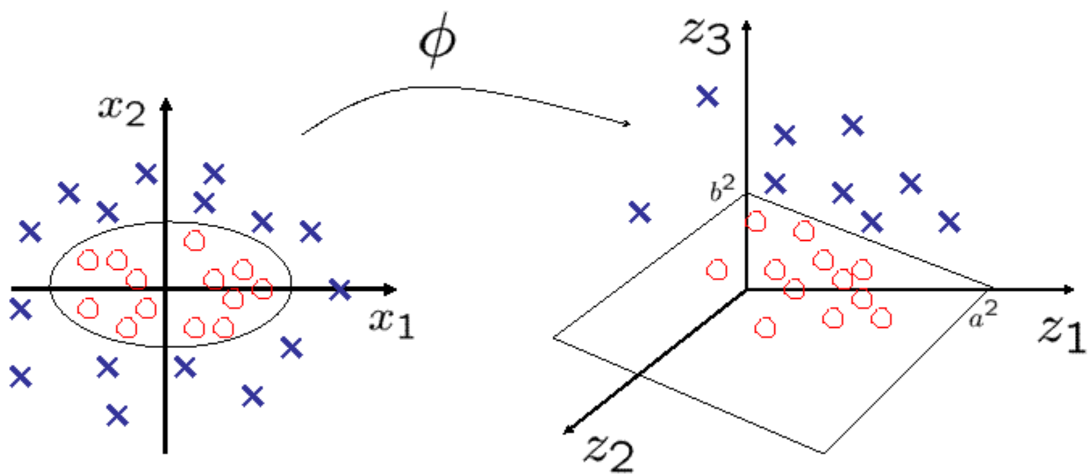
x

Extract features

$\phi(x)$



Features



$$\phi : (x_1, x_2) \longrightarrow (x_1^2, \sqrt{2}x_1x_2, x_2^2)$$

$$\left(\frac{x_1}{a}\right)^2 + \left(\frac{x_2}{b}\right)^2 = 1 \longrightarrow \frac{z_1}{a^2} + \frac{z_3}{b^2} = 1$$

Proper feature mapping can make non-linear to linear!

Recall: SVM dual form

Only depend on inner products

- Reduces to dual problem:

$$\mathcal{L}(w, b, \boldsymbol{\alpha}) = \sum_i \alpha_i - \frac{1}{2}\sum_{ij} \alpha_i \alpha_j y_i y_j x_i^T x_j$$

$$\sum_i \alpha_i y_i = 0, \alpha_i \geq 0$$

- Since $w = \sum_i \alpha_i y_i x_i$, we have $w^T x + b = \sum_i \alpha_i y_i x_i^T x + b$

Features

- Using SVM on the feature space $\{\phi(x_i)\}$: only need $\phi(x_i)^T\phi(x_j)$
- Conclusion: no need to design $\phi(\cdot)$, only need to design

$$k(x_i, x_j) = \phi(x_i)^T\phi(x_j)$$

Polynomial kernels

- Fix degree d and constant c:
$$k(x, x') = (x^T x' + c)^d$$
- What are $\phi(x)$?
- Expand the expression to get $\phi(x)$

Polynomial kernels

$$\forall \mathbf{x}, \mathbf{x}' \in \mathbb{R}^2, \quad K(\mathbf{x}, \mathbf{x}') = (x_1 x_1' + x_2 x_2' + c)^2 = \begin{bmatrix} x_1^2 \\ x_2^2 \\ \sqrt{2}\, x_1 x_2 \\ \sqrt{2c}\, x_1 \\ \sqrt{2c}\, x_2 \\ c \end{bmatrix} \cdot \begin{bmatrix} {x'}_1^2 \\ {x'}_2^2 \\ \sqrt{2}\, x_1' x_2' \\ \sqrt{2c}\, x_1' \\ \sqrt{2c}\, x_2' \\ c \end{bmatrix}$$

Figure from Foundations of Machine Learning, by M. Mohri, A. Rostamizadeh, and A. Talwalkar

SVMs with polynomial kernels

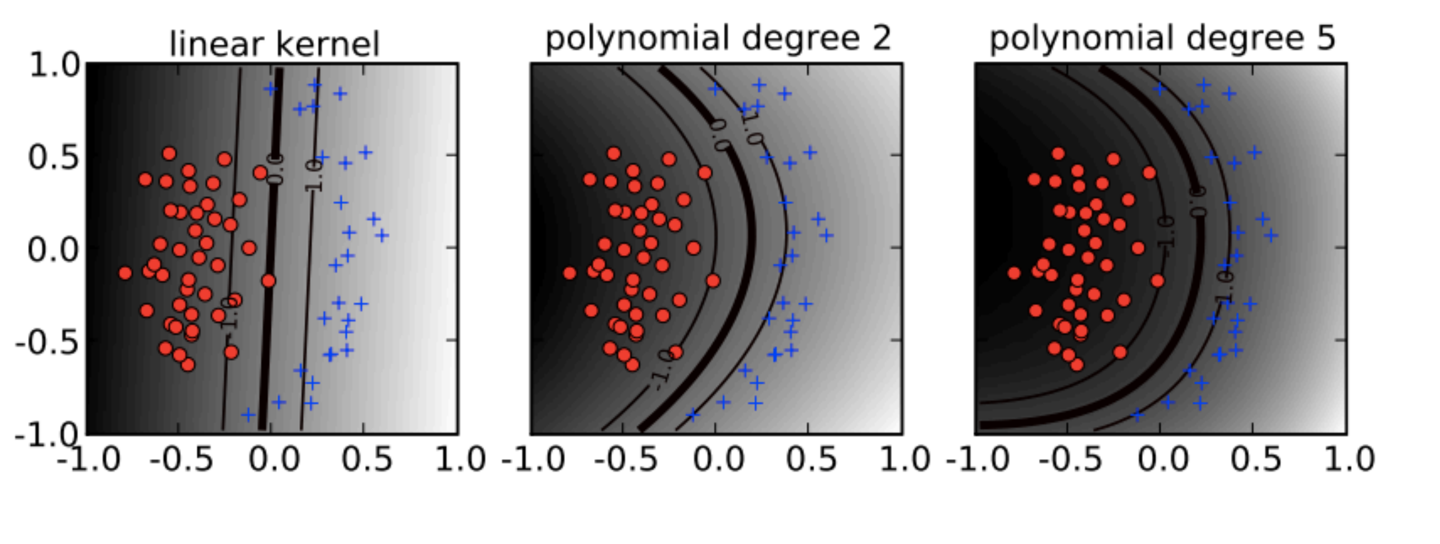


Figure from Ben-Hur & Weston,
Methods in Molecular Biology 2010

Gaussian/RBF kernels

- Fix bandwidth σ:

$$k(x, x') = \exp(-||x - x'||^2 / 2\sigma^2)$$

- Also called radial basis function (RBF) kernels

- What are $\phi(x)$? Consider the un-normalized version

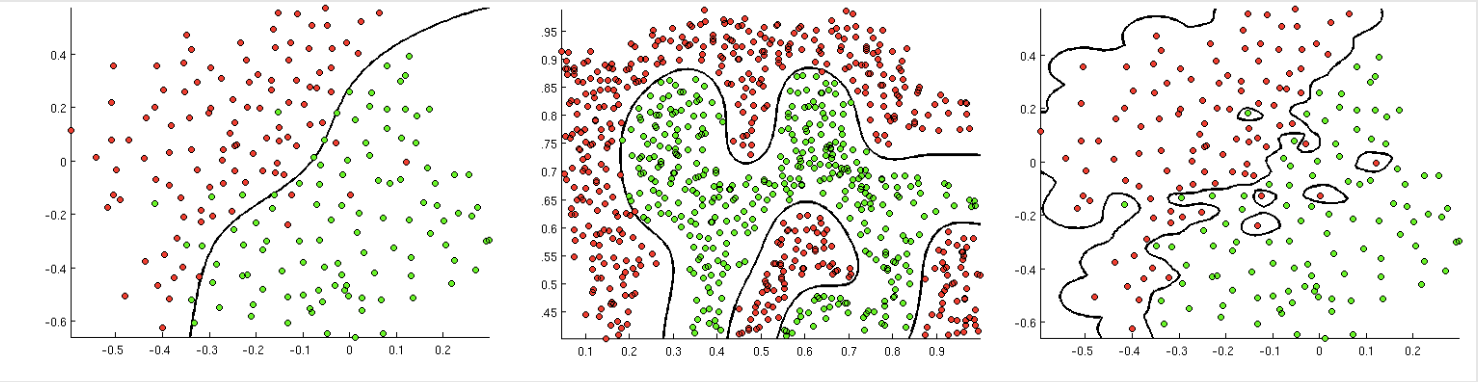
$$k'(x, x') = \exp(x^T x' / \sigma^2)$$

- Power series expansion:

$$k'(x, x') = \sum_{i}^{+\infty} \frac{(x^T x')^i}{\sigma^i i!}$$

The RBF kernel illustrated

$\gamma = -10$ $\gamma = -100$ $\gamma = -1000$



Figures from openclassroom.stanford.edu (Andrew Ng)

Mercer's condition for kenerls

- Theorem: $k(x, x')$ has expansion

$$k(x, x') = \sum_{i}^{+\infty} a_i \phi_i(x) \phi_i(x')$$

if and only if for any function $c(x)$,

$$\int \int c(x) c(x') k(x, x') dx dx' \geq 0$$

(Omit some math conditions for k and c)

Constructing new kernels

- Kernels are closed under positive scaling, sum, product, pointwise limit, and composition with a power series $\sum_i^{+\infty} a_i k^i(x, x')$

- Example: $k_1(x, x'), k_2(x, x')$ are kernels, then also is

$$k(x, x') = 2k_1(x, x') + 3k_2(x, x')$$

- Example: $k_1(x, x')$ is kernel, then also is

$$k(x, x') = \exp(k_1(x, x'))$$

Kernel algebra

- given a valid kernel, we can make new valid kernels using a variety of operators

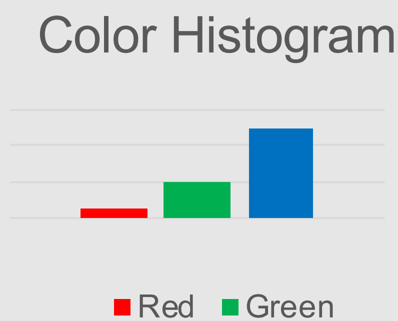
kernel composition	mapping composition
$k(\boldsymbol{x},\boldsymbol{v}) = k_a(\boldsymbol{x},\boldsymbol{v}) + k_b(\boldsymbol{x},\boldsymbol{v})$	$\phi(\boldsymbol{x}) = \left(\phi_a(\boldsymbol{x}),\ \phi_b(\boldsymbol{x})\right)$
$k(\boldsymbol{x},\boldsymbol{v}) = \gamma\ k_a(\boldsymbol{x},\boldsymbol{v}),\ \gamma > 0$	$\phi(\boldsymbol{x}) = \sqrt{\gamma}\ \phi_a(\boldsymbol{x})$
$k(\boldsymbol{x},\boldsymbol{v}) = k_a(\boldsymbol{x},\boldsymbol{v})k_b(\boldsymbol{x},\boldsymbol{v})$	$\phi_l(\boldsymbol{x}) = \phi_{ai}(\boldsymbol{x})\phi_{bj}(\boldsymbol{x})$
$k(\boldsymbol{x},\boldsymbol{v}) = \boldsymbol{x}^{\mathsf{T}}A\boldsymbol{v},\quad A \text{ is p.s.d.}$	$\phi(\boldsymbol{x}) = L^{\mathsf{T}}\boldsymbol{x},\ \text{where } A = LL^{\mathsf{T}}$
$k(\boldsymbol{x},\boldsymbol{v}) = f(\boldsymbol{x})f(\boldsymbol{v})k_a(\boldsymbol{x},\boldsymbol{v})$	$\phi(\boldsymbol{x}) = f(\boldsymbol{x})\phi_a(\boldsymbol{x})$

Kernels v.s. Neural networks

Features

x

Extract features

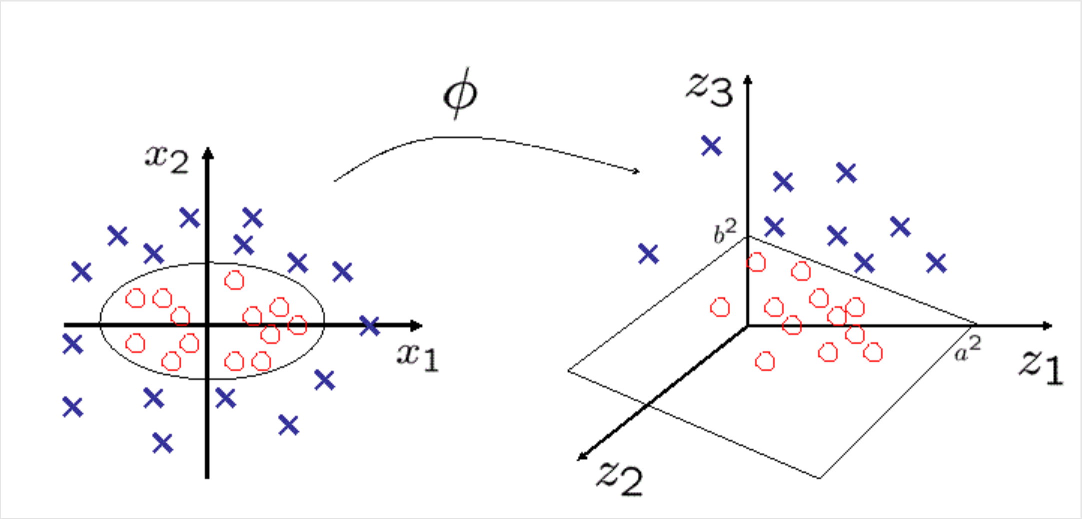


build hypothesis

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

Features: part of the model

Nonlinear model



build hypothesis

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

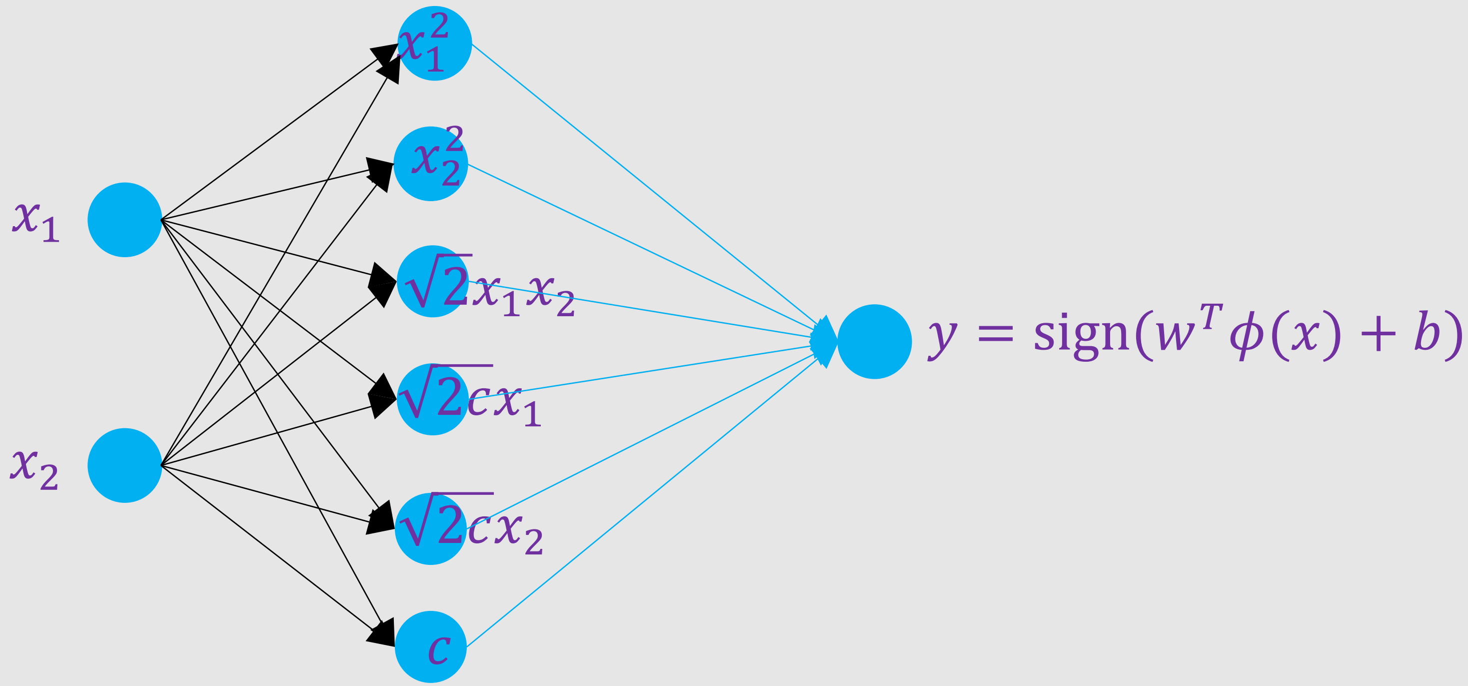
Linear model

Polynomial kernels

$$\forall \mathbf{x}, \mathbf{x}' \in \mathbb{R}^2, \quad K(\mathbf{x}, \mathbf{x}') = (x_1 x_1' + x_2 x_2' + c)^2 = \begin{bmatrix} x_1^2 \\ x_2^2 \\ \sqrt{2}\, x_1 x_2 \\ \sqrt{2c}\, x_1 \\ \sqrt{2c}\, x_2 \\ c \end{bmatrix} \cdot \begin{bmatrix} {x'}_1^2 \\ {x'}_2^2 \\ \sqrt{2}\, x_1' x_2' \\ \sqrt{2c}\, x_1' \\ \sqrt{2c}\, x_2' \\ c \end{bmatrix}$$

Figure from Foundations of Machine Learning, by M. Mohri, A. Rostamizadeh, and A. Talwalkar

Polynomial kernel SVM as two layer neural network



First layer is fixed. If also learn first layer, it becomes two layer neural network

Comments on SVMs

- we can find solutions that are globally optimal (maximize the margin)
  - because the learning task is framed as a convex optimization problem
  - no local minima, in contrast to multi-layer neural nets

- there are two formulations of the optimization: *primal* and *dual*
  - dual represents classifier decision in terms of support vectors
  - dual enables the use of kernel functions

- we can use a wide range of optimization methods to learn SVM
  - standard quadratic programming solvers
  - SMO [Platt, 1999]
  - linear programming solvers for some formulations
  - etc.

Comments on SVMs

- kernels provide a powerful way to
  - allow nonlinear decision boundaries
  - represent/compare complex objects such as strings and trees
  - incorporate domain knowledge into the learning task
- using the kernel trick, we can implicitly use high-dimensional mappings without explicitly computing them
- one SVM can represent only a binary classification task; multi-class problems handled using multiple SVMs and some encoding
- empirically, SVMs have shown (close to) state-of-the art accuracy for many tasks
- the kernel idea can be extended to other tasks (anomaly detection, regression, etc.)

THANK YOU

Some of the slides in these lectures have been adapted/borrowed from materials developed by Yingyu Liang, Mark Craven, David Page, Jude Shavlik, Tom Mitchell, Nina Balcan, Elad Hazan, Tom Dietterich, and Pedro Domingos.