Support Vector Machines

Dr. Jianlin Cheng

Department of Electrical Engineering and Computer Science

University of Missouri, Columbia

Fall, 2019

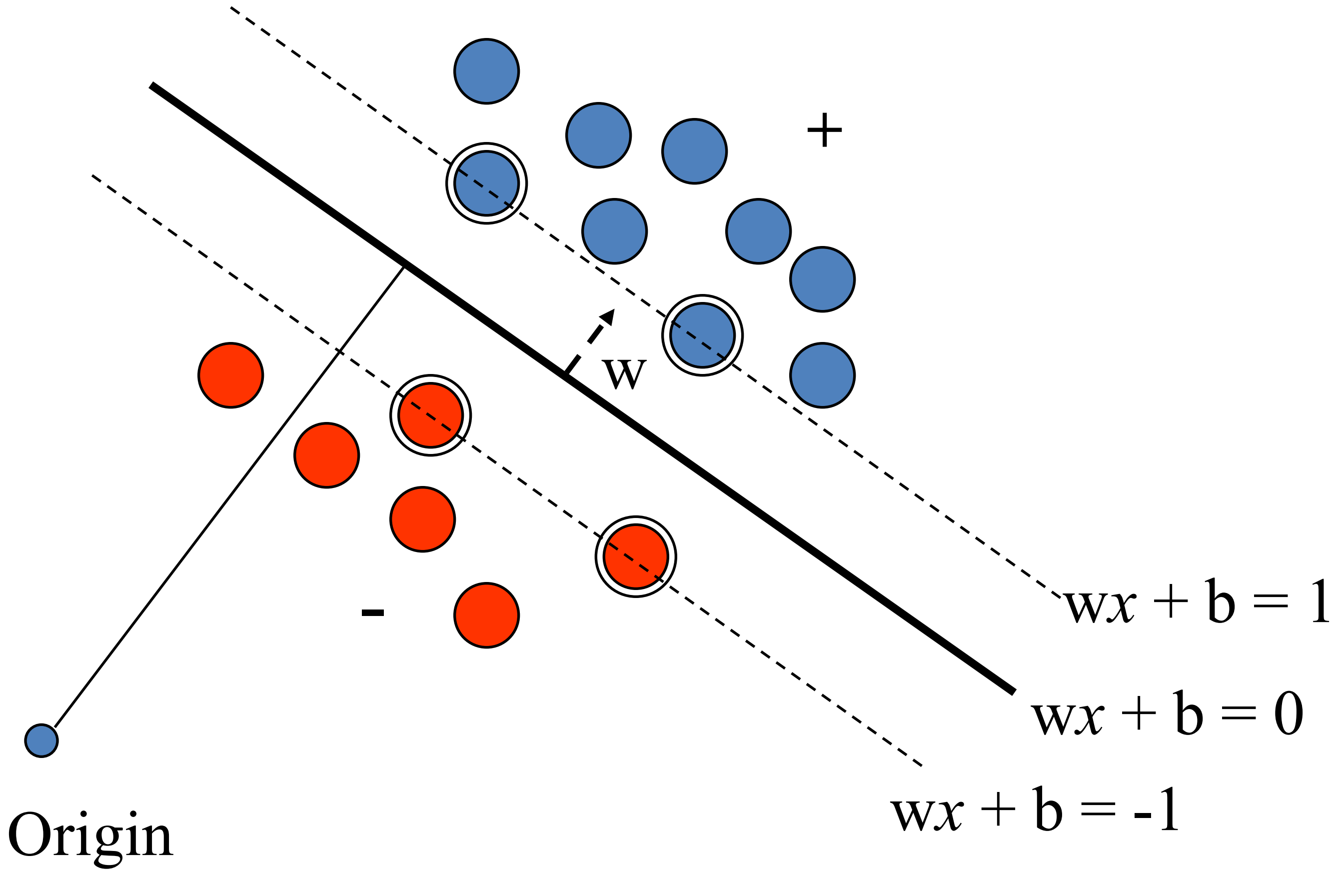
Slides Adapted from Book and CMU, Stanford Machine Learning Courses

Deterministic Learning Machine

- Learning a mapping: $x_i \mapsto y_i$.
- The machine is defined by a set of mappings (functions): $f(x, a)$
- $f(x,a)$ are defined by the adjustable parameters a. The machine is assumed to be deterministic.
- A particular choice of a generates a “trained” machine (**examples?**)

Linear Classification Hyperplane

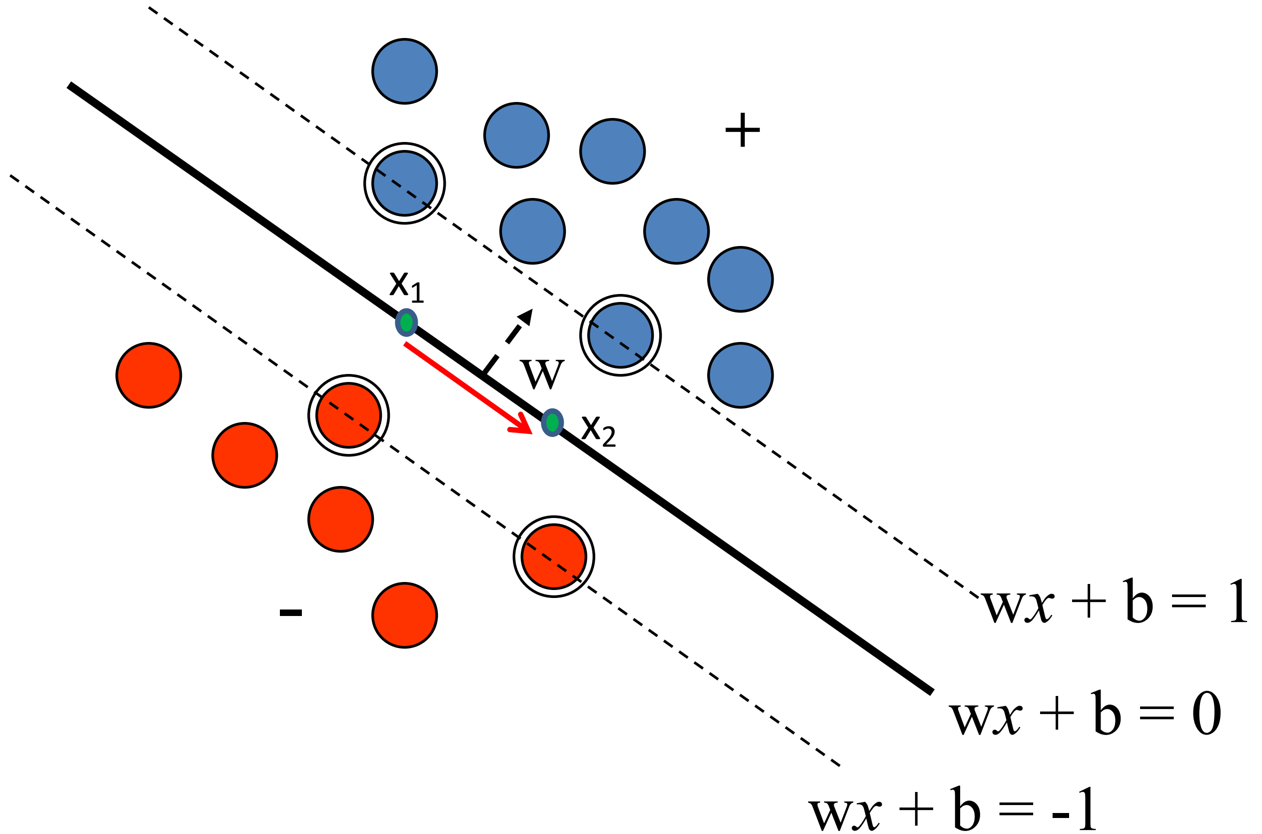
- A set of labeled training data $\{x_i, y_i\}$, i = 1, ..., l, x_i in R^d, y_i in {-1, 1}.
- A linear machine trained on the separable data.
- A linear hyperplane $f(x) = w.x+b$, separates the positive from negative examples, w is normal to the hyperplane.
- The points which lie on the hyperplane satisfy $w.x + b = 0$, positives $w.x+b > 0$, and negatives $w.x+b < 0$.



$x_i \,.\, w + b >= +1$ for $y_i = +1$
$x_i \,.\, w + b <= -1$, for $y_i = -1$

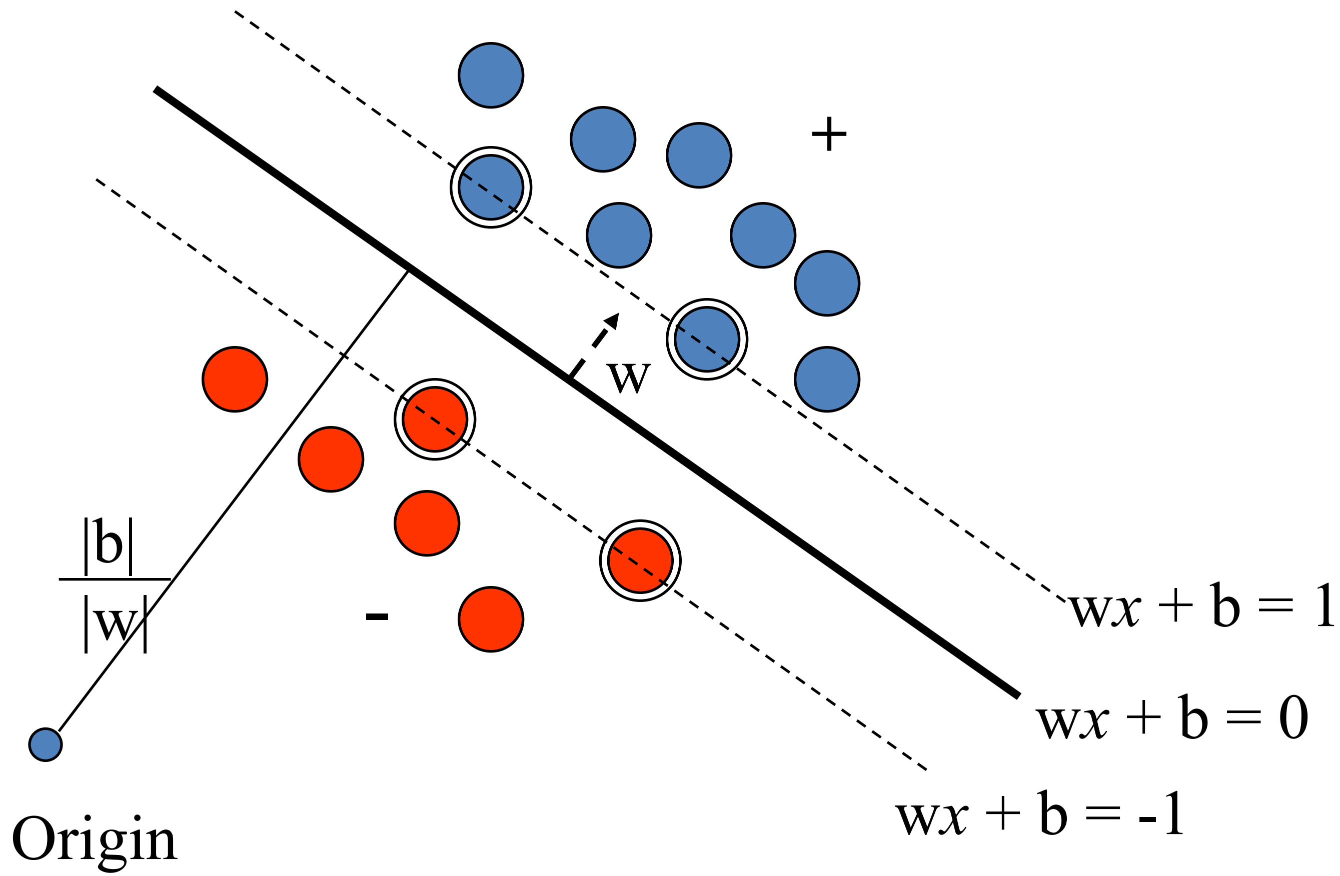
combined into: $y_i(x_i.w+b) >= 1$

Comments: equivalent to general form $y_i(x_i.w+b) >= c$



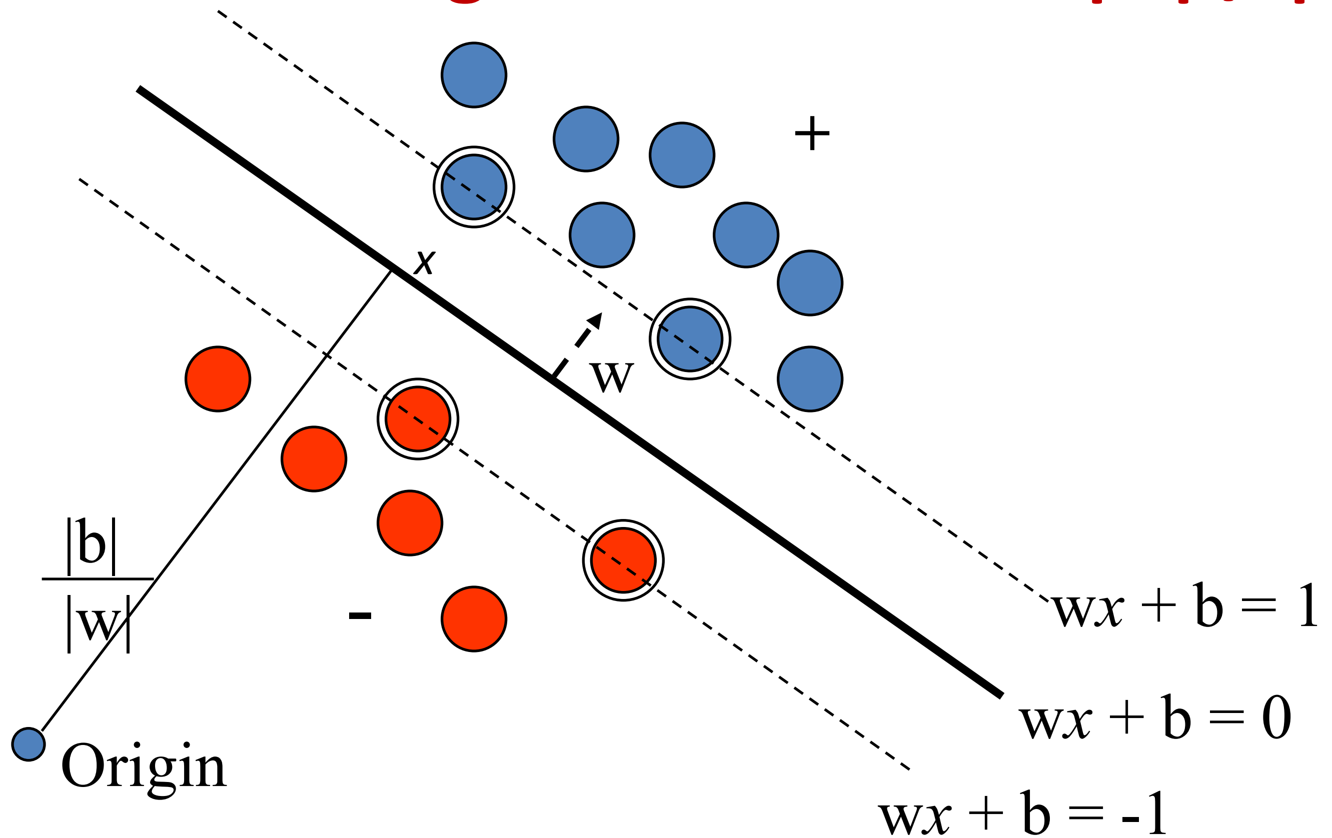
Prove w is normal (perpendicular) to the hyperplane

$$(x_2 - x_1) \cdot w = x_2 . w - x_1 . w = -b - (-b) = 0$$



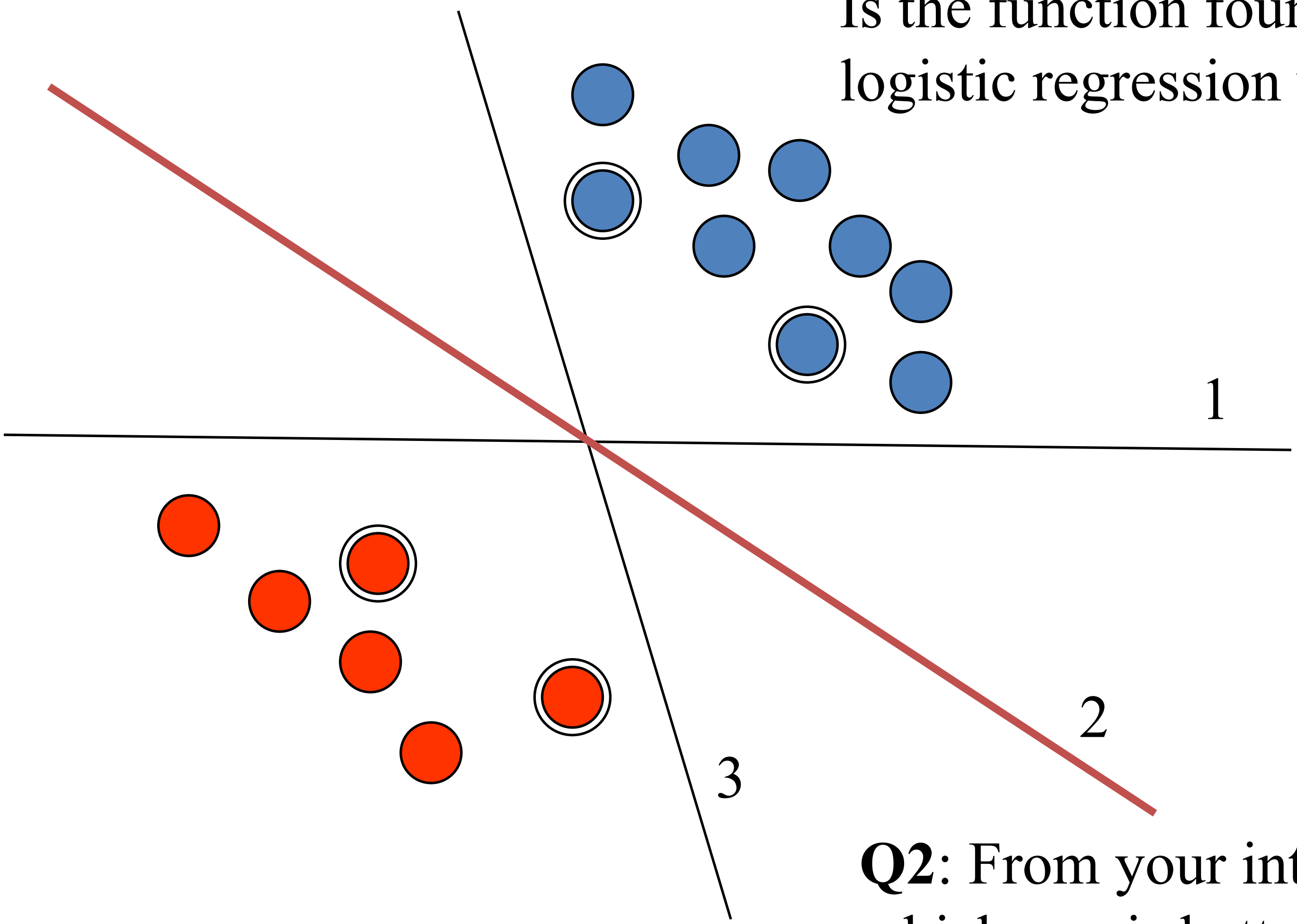
+
w
-
|b|
|w|
$wx + b = 1$
$wx + b = 0$
$wx + b = -1$
Origin

Distance from origin to wx + b=0 is |b| / |w|



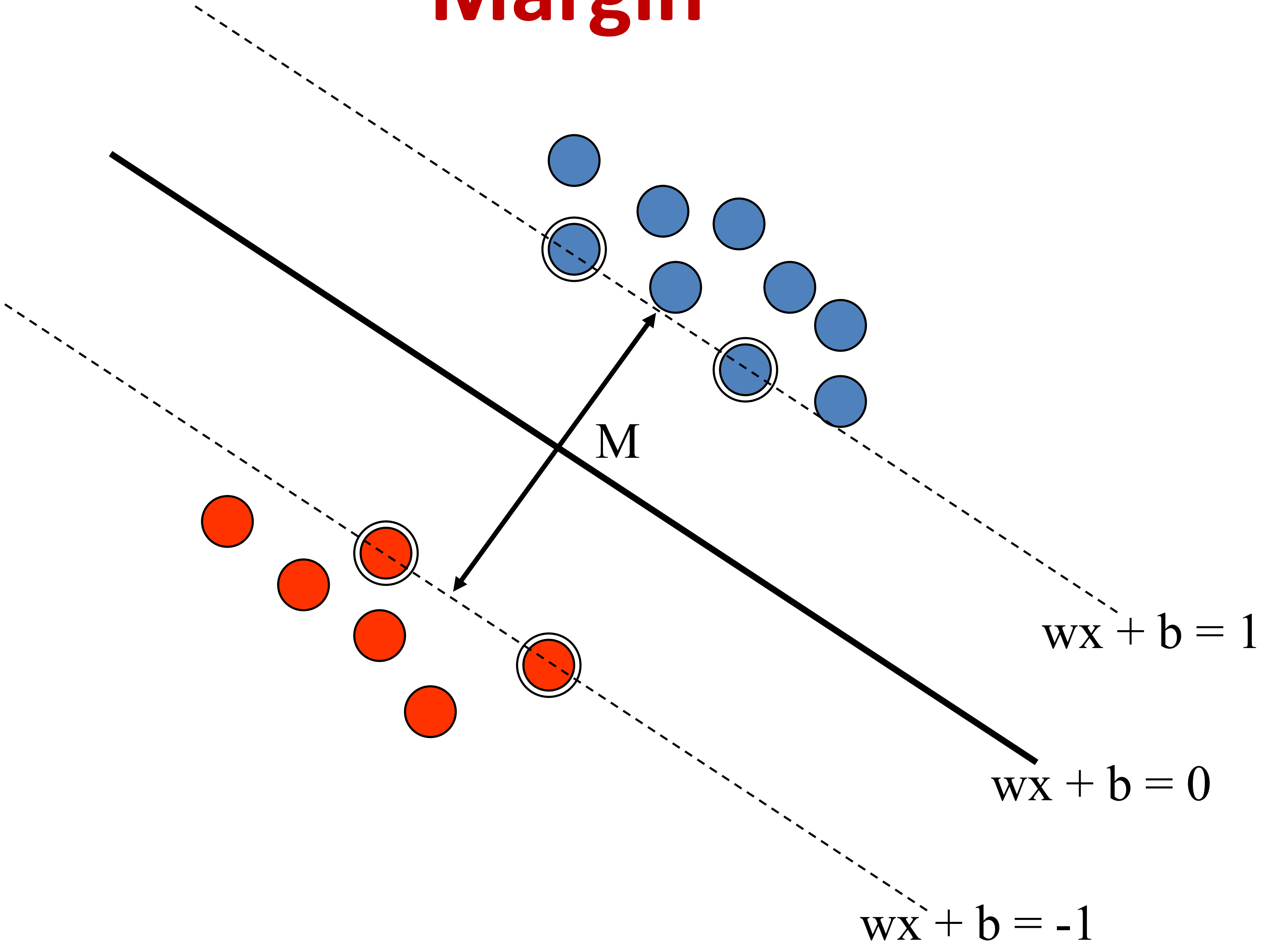
- Choose a point x on wx+b=0 such that vector (0, x) is perpendicular to wx+b = 0. So x is λw because w is norm of wx+b=0.
- So λw.w+b = 0 ➔ λ = -b/ w.w = -b / $|w|^2$
- So x = -b / $|w|^2$ *w ➔ |x| = |b| / |w|.

Q1: How logistic regression finds a linear hyperplane? Is the function found by logistic regression unique?

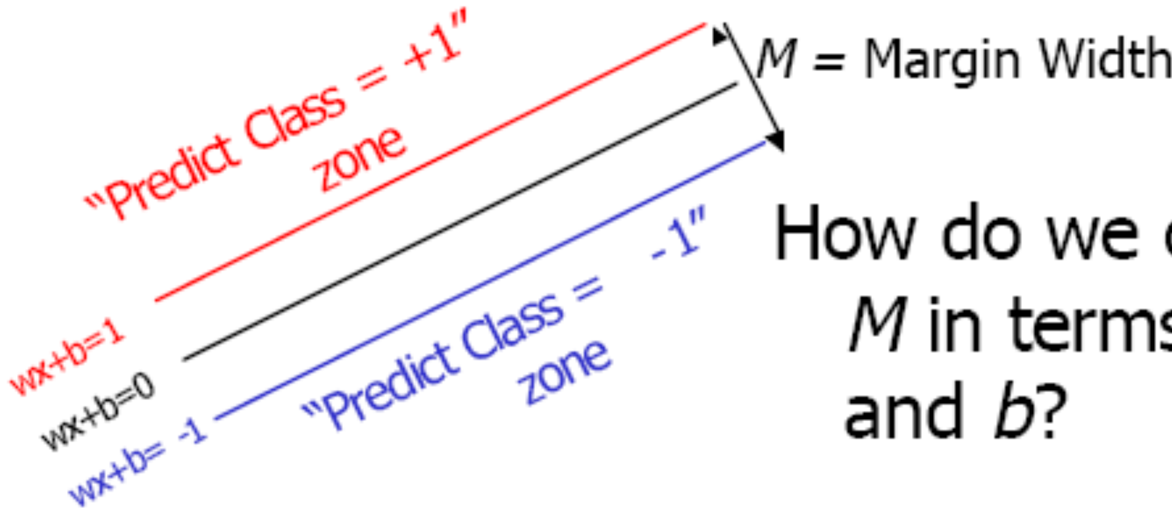


Q2: From your intuition, which one is better?

Margin



How to Compute Margin?

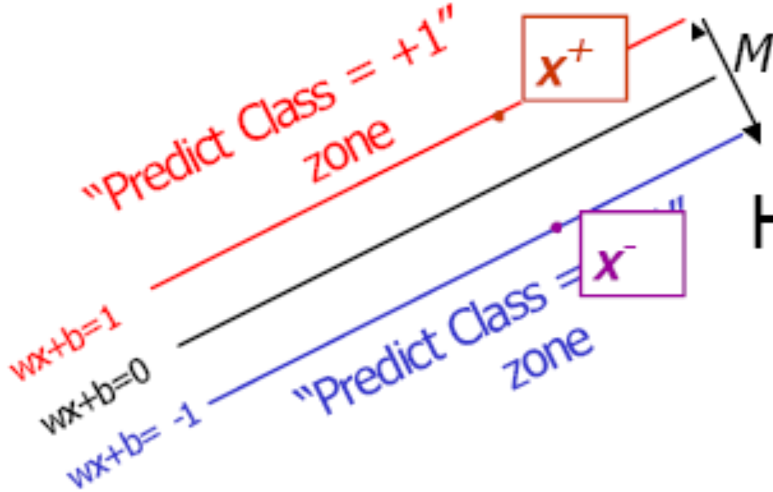


How do we compute M in terms of $\boldsymbol{w}$ and b?

- Plus-plane = $\{ \mathbf{x} : \mathbf{w} \cdot \mathbf{x} + b = +1 \}$
- Minus-plane = $\{ \mathbf{x} : \mathbf{w} \cdot \mathbf{x} + b = -1 \}$

A. Moore, 2003

Computing the margin width



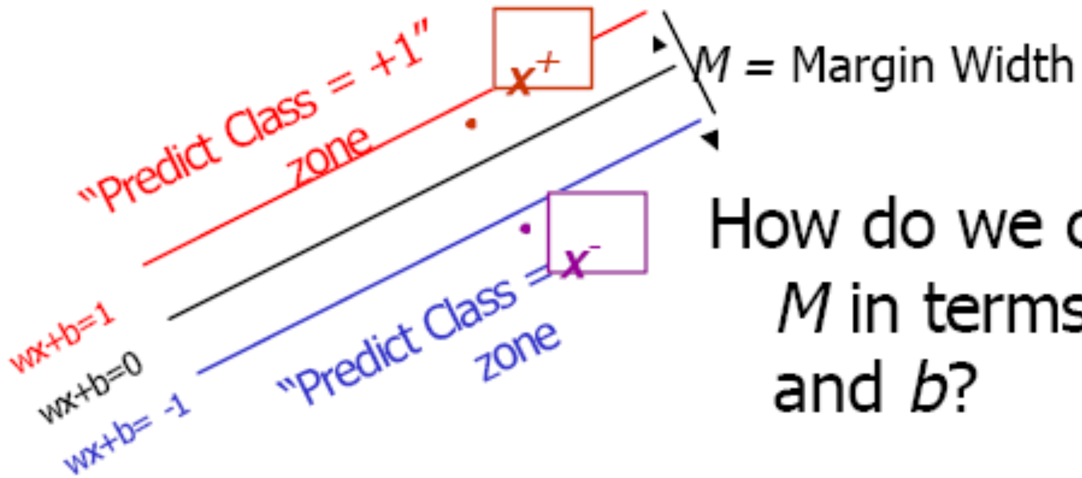
How do we compute *M* in terms of **w** and *b*?

- Plus-plane = $\{ \mathbf{x} : \mathbf{w} \,.\, \mathbf{x} + b = +1 \}$
- Minus-plane = $\{ \mathbf{x} : \mathbf{w} \,.\, \mathbf{x} + b = -1 \}$
- The vector **w** is perpendicular to the Plus Plane
- Let $\mathbf{x}^-$ be any point on the minus plane
- Let $\mathbf{x}^+$ be the closest plus-plane-point to $\mathbf{x}^-$.

Any location in R^m: not necessarily a datapoint

A. Moore, 2003

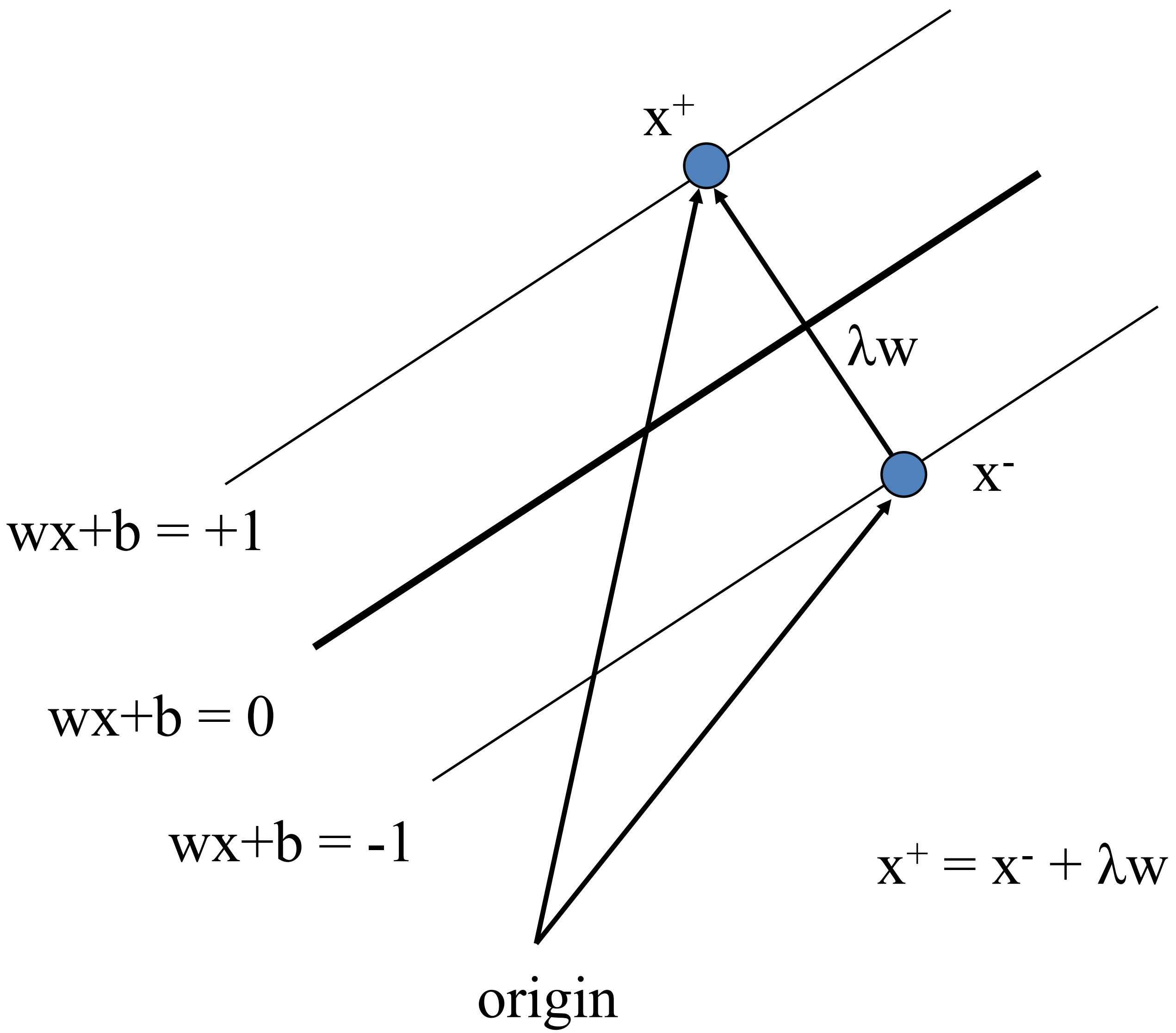
Computing the margin width



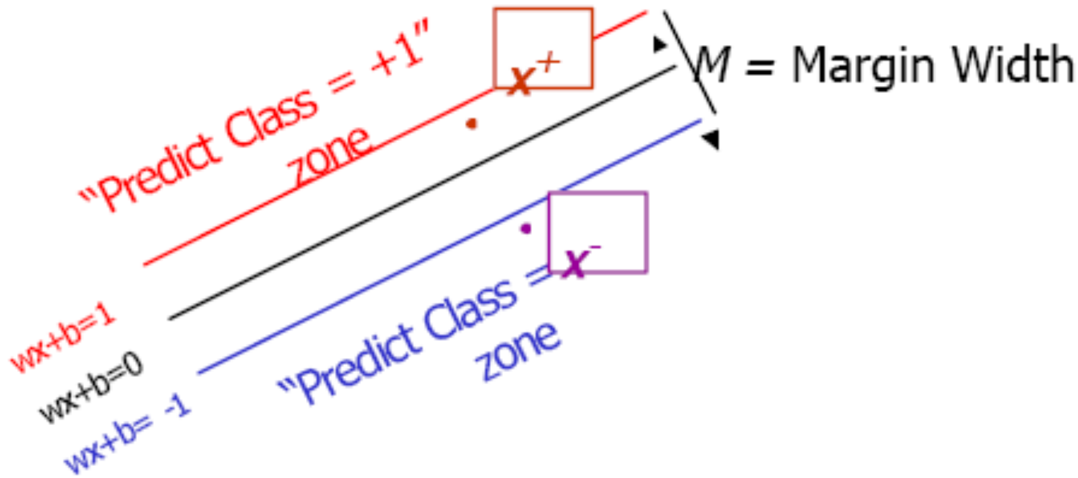
How do we compute *M* in terms of ***w*** and *b*?

- Plus-plane = $\{ \mathbf{x} : \mathbf{w} \cdot \mathbf{x} + b = +1 \}$
- Minus-plane = $\{ \mathbf{x} : \mathbf{w} \cdot \mathbf{x} + b = -1 \}$
- The vector **w** is perpendicular to the Plus Plane
- Let $\mathbf{x}^-$ be any point on the minus plane
- Let $\mathbf{x}^+$ be the closest plus-plane-point to $\mathbf{x}^-$.
- Claim: $\mathbf{x}^+ = \mathbf{x}^- + \lambda \mathbf{w}$ for some value of λ. Why?

A. Moore, 2003


x^+
λw
x^-
$wx+b = +1$
$wx+b = 0$
$wx+b = -1$
$x^+ = x^- + \lambda w$
origin

Computing the margin width



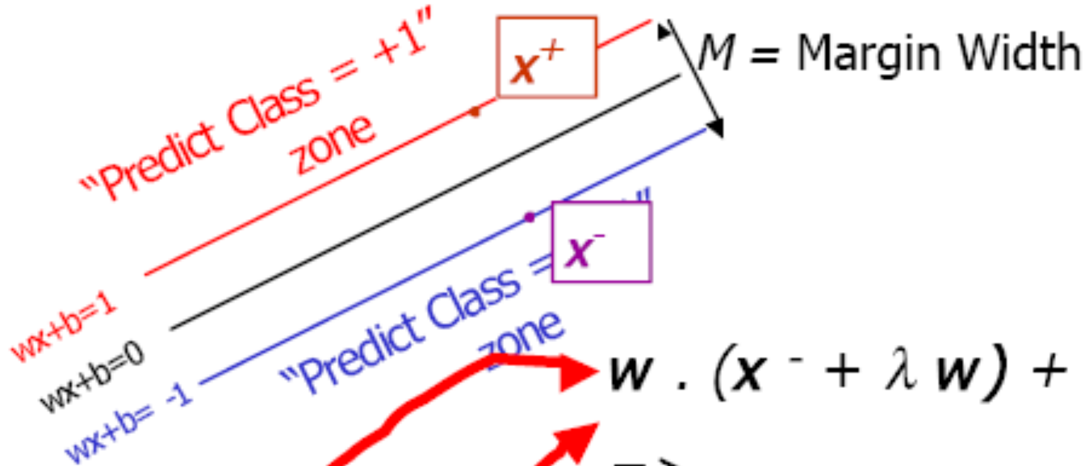
What we know:

- $\mathbf{w} \cdot \mathbf{x}^+ + b = +1$
- $\mathbf{w} \cdot \mathbf{x}^- + b = -1$
- $\mathbf{x}^+ = \mathbf{x}^- + \lambda \mathbf{w}$
- $|\mathbf{x}^+ - \mathbf{x}^-| = M$

It's now easy to get *M* in terms of ***w*** and *b*

A. Moore, 2003

Computing the margin width



What we know:

- $\mathbf{w} \cdot \mathbf{x}^+ + b = +1$
- $\mathbf{w} \cdot \mathbf{x}^- + b = -1$
- $\mathbf{x}^+ = \mathbf{x}^- + \lambda\, \mathbf{w}$
- $|\mathbf{x}^+ - \mathbf{x}^-| = M$

It's now easy to get M in terms of $\mathbf{w}$ and b

$$\mathbf{w} \cdot (\mathbf{x}^- + \lambda\, \mathbf{w}) + b = 1$$

=>

$$\mathbf{w} \cdot \mathbf{x}^- + b + \lambda\, \mathbf{w} \cdot \mathbf{w} = 1$$

=>

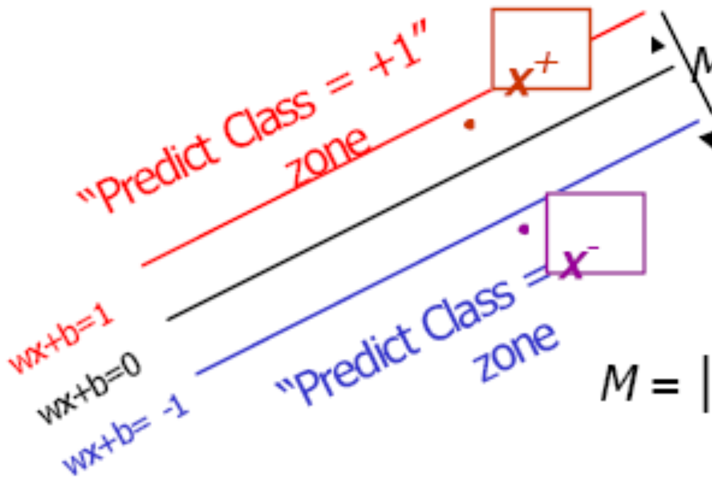
$$-1 + \lambda\, \mathbf{w} \cdot \mathbf{w} = 1$$

=>

$$\lambda = \frac{2}{\mathbf{w} \cdot \mathbf{w}}$$

A. Moore, 2003

Computing the margin width



M = Margin Width = $\dfrac{2}{\sqrt{\mathbf{w}.\mathbf{w}}}$

What we know:

- $\mathbf{w} \,.\, \mathbf{x}^+ + b = +1$
- $\mathbf{w} \,.\, \mathbf{x}^- + b = -1$
- $\mathbf{x}^+ = \mathbf{x}^- + \lambda\, \mathbf{w}$
- $|\mathbf{x}^+ - \mathbf{x}^-| = M$
- $\lambda = \dfrac{2}{\mathbf{w}.\mathbf{w}}$

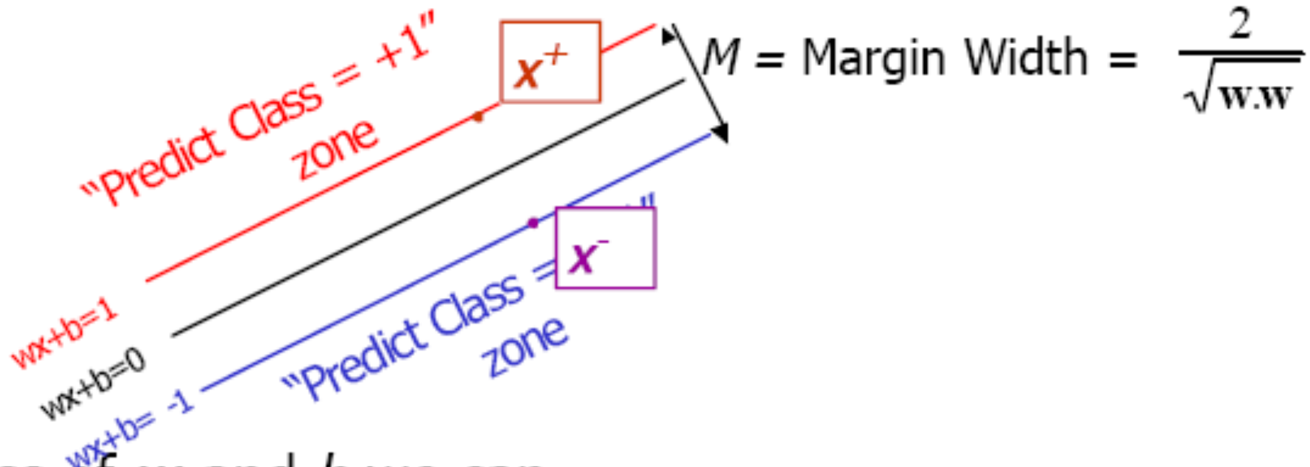
$$M = |\mathbf{x}^+ - \mathbf{x}^-| = |\lambda\, \mathbf{w}| =$$

$$= \lambda\, |\mathbf{w}| = \lambda \sqrt{\mathbf{w}.\mathbf{w}}$$

$$= \frac{2\sqrt{\mathbf{w}.\mathbf{w}}}{\mathbf{w}.\mathbf{w}} = \frac{2}{\sqrt{\mathbf{w}.\mathbf{w}}}$$

A. Moore, 2003

Learning the Maximum Margin Classifier



Given a guess of ***w*** and *b* we can

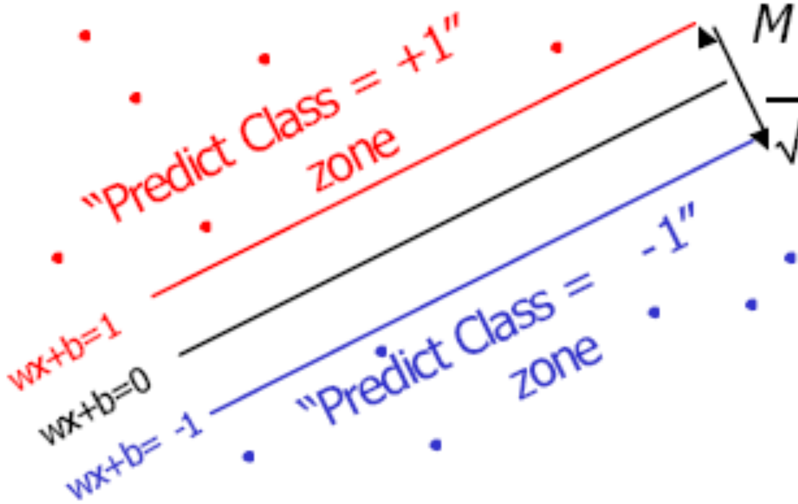
- Compute whether all data points in the correct half-planes
- Compute the width of the margin

So now we just need to write a program to search the space of **w**'s and *b*'s to find the widest margin that matches all the datapoints. *How?*

Gradient descent? Simulated Annealing? Matrix Inversion? EM? Newton's Method?

A. Moore, 2003

Learning the Maximum Margin Classifier



$M = \frac{2}{\sqrt{\mathbf{w}.\mathbf{w}}}$

Given guess of $\mathbf{w}$, b we can

- Compute whether all data points are in the correct half-planes
- Compute the margin width

Assume R datapoints, each $(\mathbf{x}_k, y_k)$ where $y_k = +/- 1$

What should our quadratic optimization criterion be?

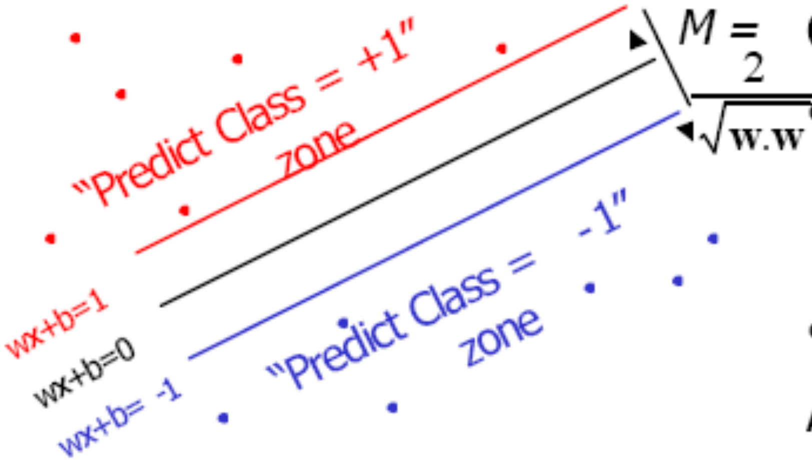
How many constraints will we have?

What should they be?

Constrained Optimization Problem

A. Moore, 2003

Learning the Maximum Margin Classifier



Given guess of $\mathbf{w}$, b we can

- Compute whether all data points are in the correct half-planes
- Compute the margin width

Assume R datapoints, each $(\mathbf{x}_k, y_k)$ where y_k = +/- 1

What should our quadratic optimization criterion be?

Minimize $\mathbf{w}.\mathbf{w}$

How many constraints will we have? R

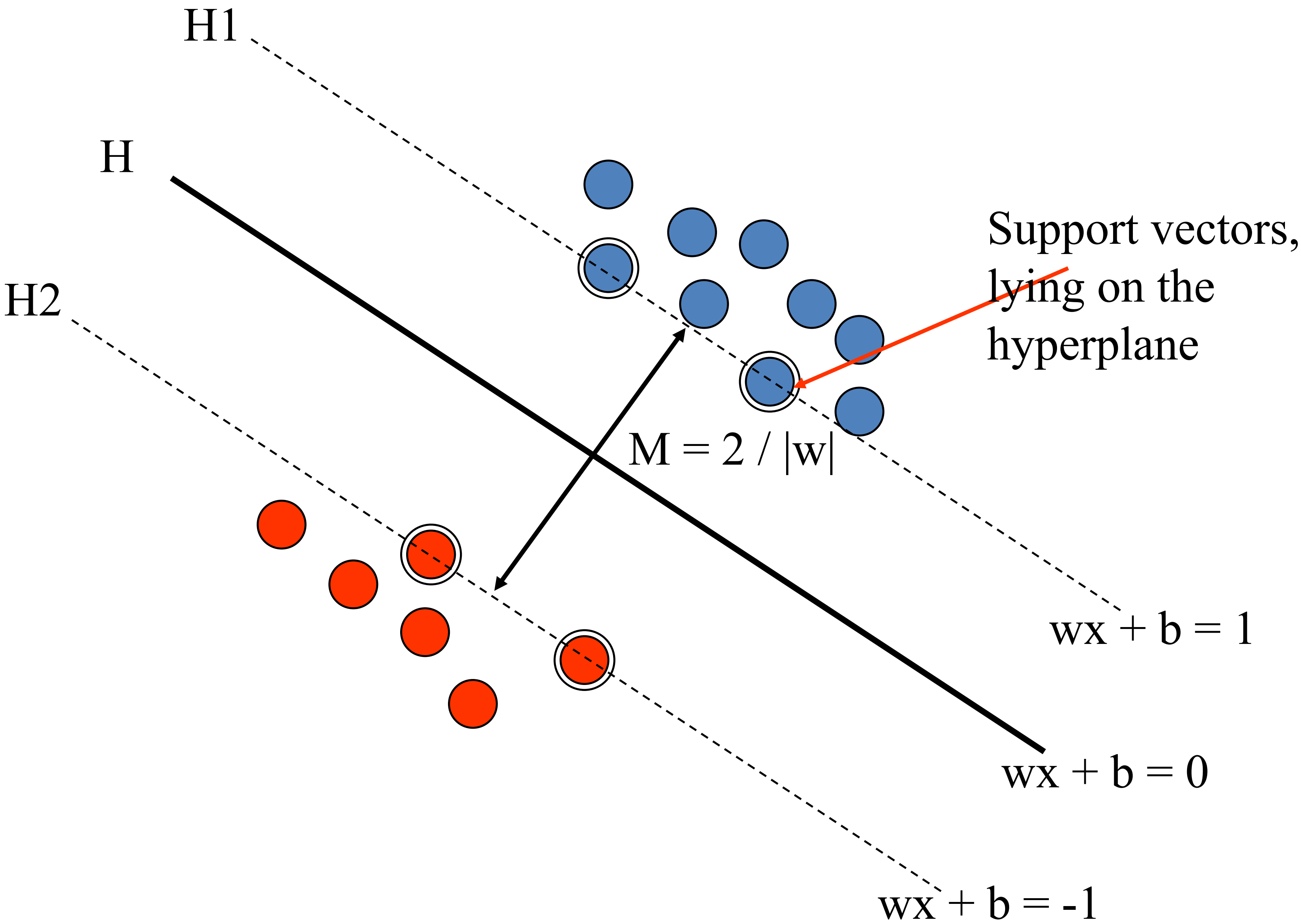
What should they be?

$\mathbf{w} \cdot \mathbf{x}_k + b >= 1$ if $y_k = 1$

$\mathbf{w} \cdot \mathbf{x}_k + b <= -1$ if $y_k = -1$

A. Moore, 2003

Margin and Support Vectors



Relationship with Vapnik Chevonenkis (VC) Dimension Learning Theory

Expectation of Test Error

$$R(a) = \int \frac{1}{2} | y - f(X, a) | \, p(X, y) dX dy$$

$R(a)$ is called expected risk / loss, the same as before except the ½ ratio.

Empirical Risk $R_{emp}(a)$ is defined to be the measured mean error rate on l training examples.

$$R_{emp}(a) = \frac{1}{2l} \sum_{i=1}^{l} | y_i - f(X_i, a) |$$

Vapnik Risk Bound

- ½$|y_i - f(X_i, a)|$ is also called the loss. It can only take the values 0 and 1.
- Choose η such that 0 <= η <= 1. With probability 1 – η, the following bound holds (Vapnik, 1995)

$$R(a) \leq R_{emp}(a) + \sqrt{\left(\frac{h(\log(2l/h)+1)-\log(\eta/4)}{l}\right)}$$

Where h is a non-negative integer called the Vapnik Chevonenkis (VC) dimension, and is a measure of the notion of capacity. The second part of the right is called VC confidence.

Insights about Risk Bound

- Independent of $p(X,y)$.
- Often not possible to compute the left hand side.
- Easily compute right hand side if h is known.
- **Structural Risk Minimization**: Given sufficiently small η, taking the machine which minimizes the right hand side and gives the lowest upper bound on the actual risk.
- **Question**: how does the bound change according to η?

VC Dimension

- VC dimension is a property of a set of functions { $f(a)$ }. Here we consider functions that correspond to two-class pattern recognition case, so that $f(X,a) \in \{-1, +1\}$.

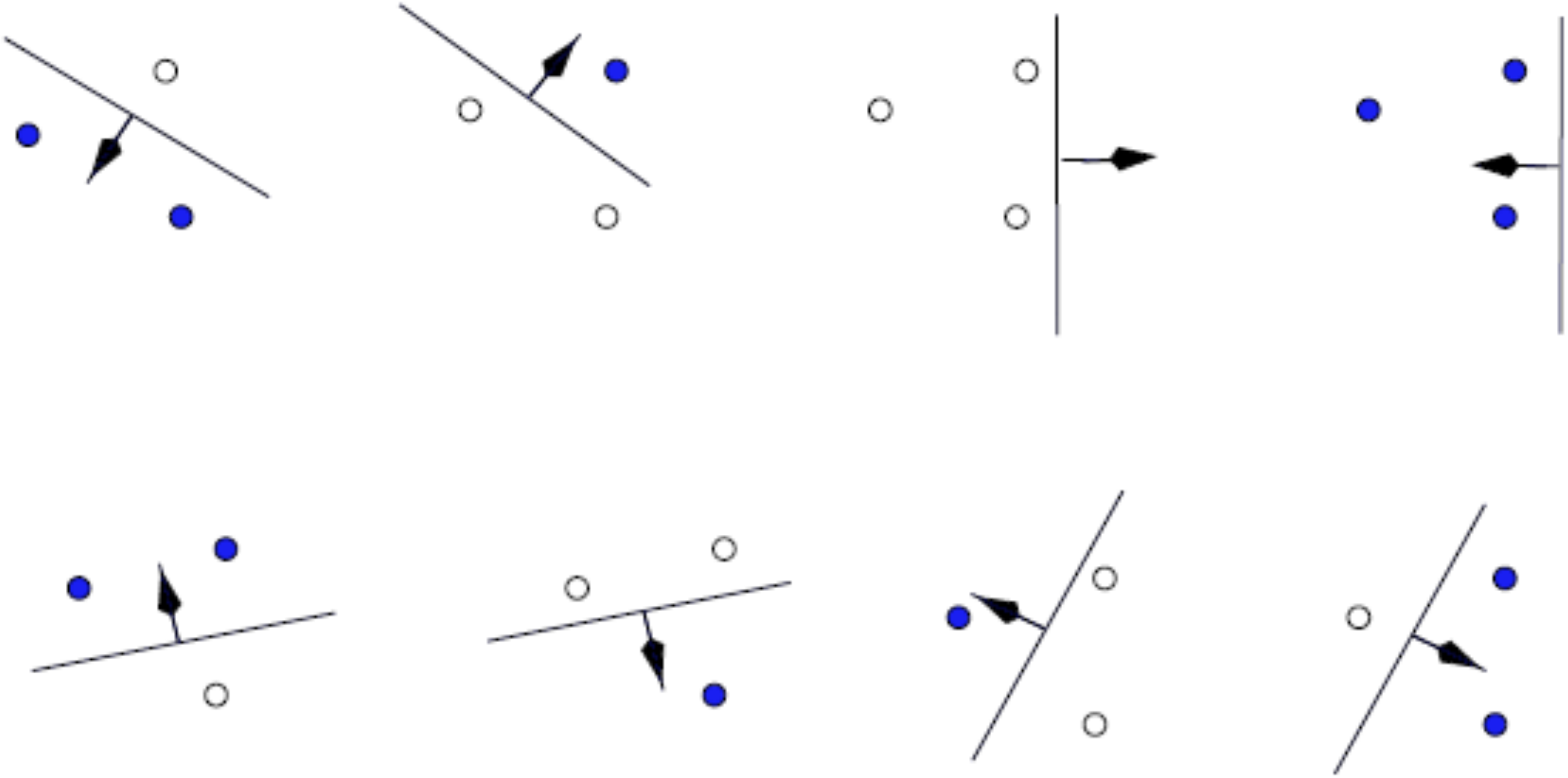
- If a given set of l points can be labeled in all possible 2^l ways, and **for each labeling**, a member of set $\{f(a)\}$ can be found to correctly assign those labels, we say that set of points is shattered by that set of functions.

VC Dimension

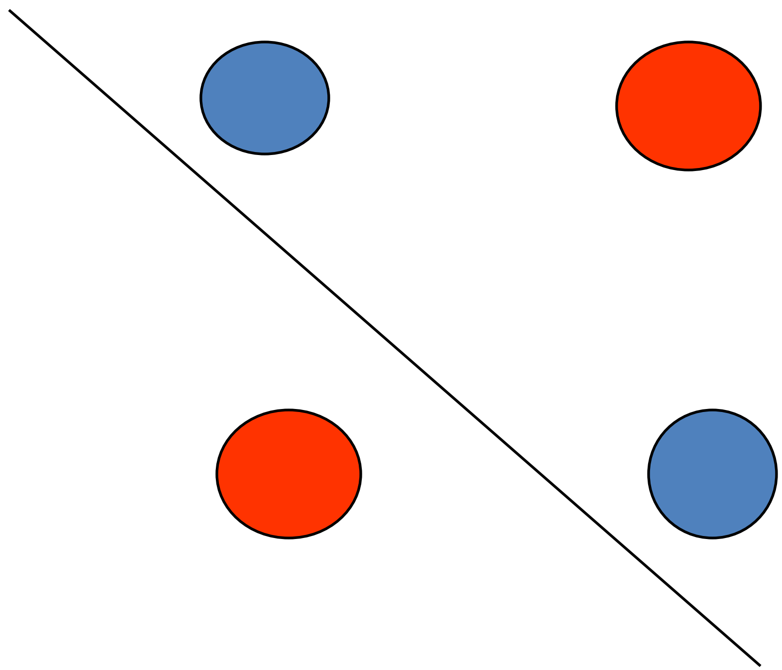
- VC dimension for a set of functions $\{f(a)\}$ is defined as the maximum number of training points that can be shattered by $\{f(a)\}$.

- If the VC dimension is *h*, then there exists at least one set of *h* points that can be shattered. But not necessary for every set of *h* points.

A linear function has VC dimension 3



8 possible labeling of 3 points can be separated by lines.



Simply can not separate the labeling of these four points using a line. So the VC dimension of a line is 3.

VC Dimension and the Number of Parameters

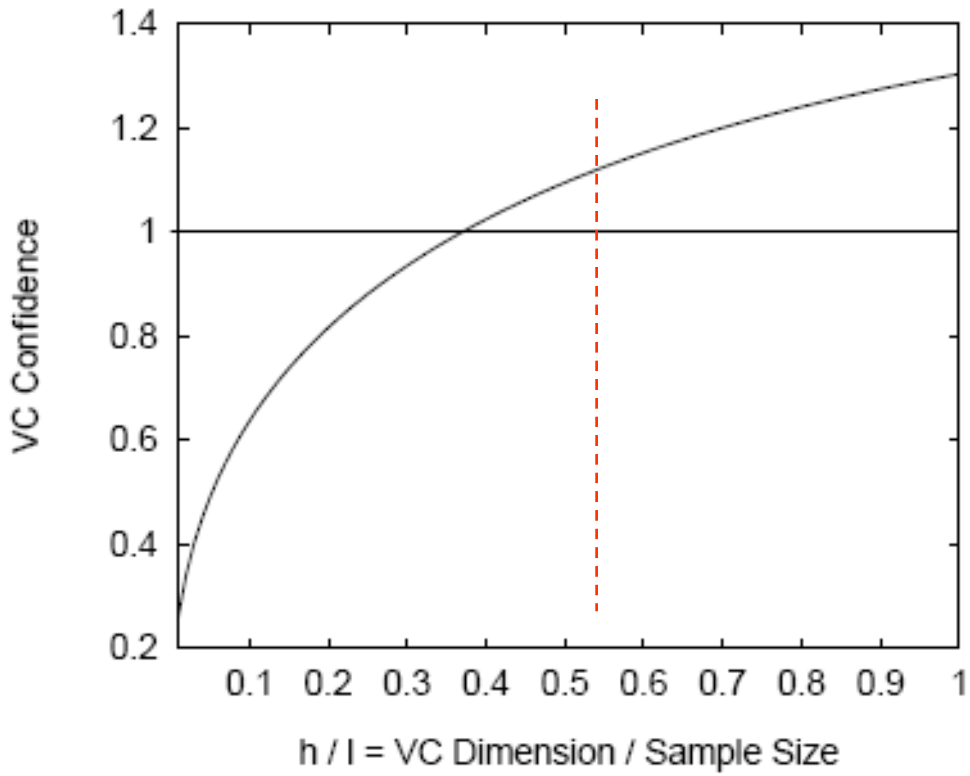
- Intuitively, more parameters ➔ higher VC dimension?
- However, 1 parameter function can have infinite VC dimension. (see Burge's tutorial)

$$f(x, \alpha) \equiv \theta(\sin(\alpha x)), \quad x, \alpha \in \mathbf{R}.$$

If $\sin(ax) > 0$, $f(x,a) = 1$, -1 otherwise

VC Confidence and VC Dimension *h*

$$R(a) \leq R_{emp}(a) + \sqrt{\left(\frac{h(\log(2l/h)+1) - \log(\eta/4)}{l}\right)}$$

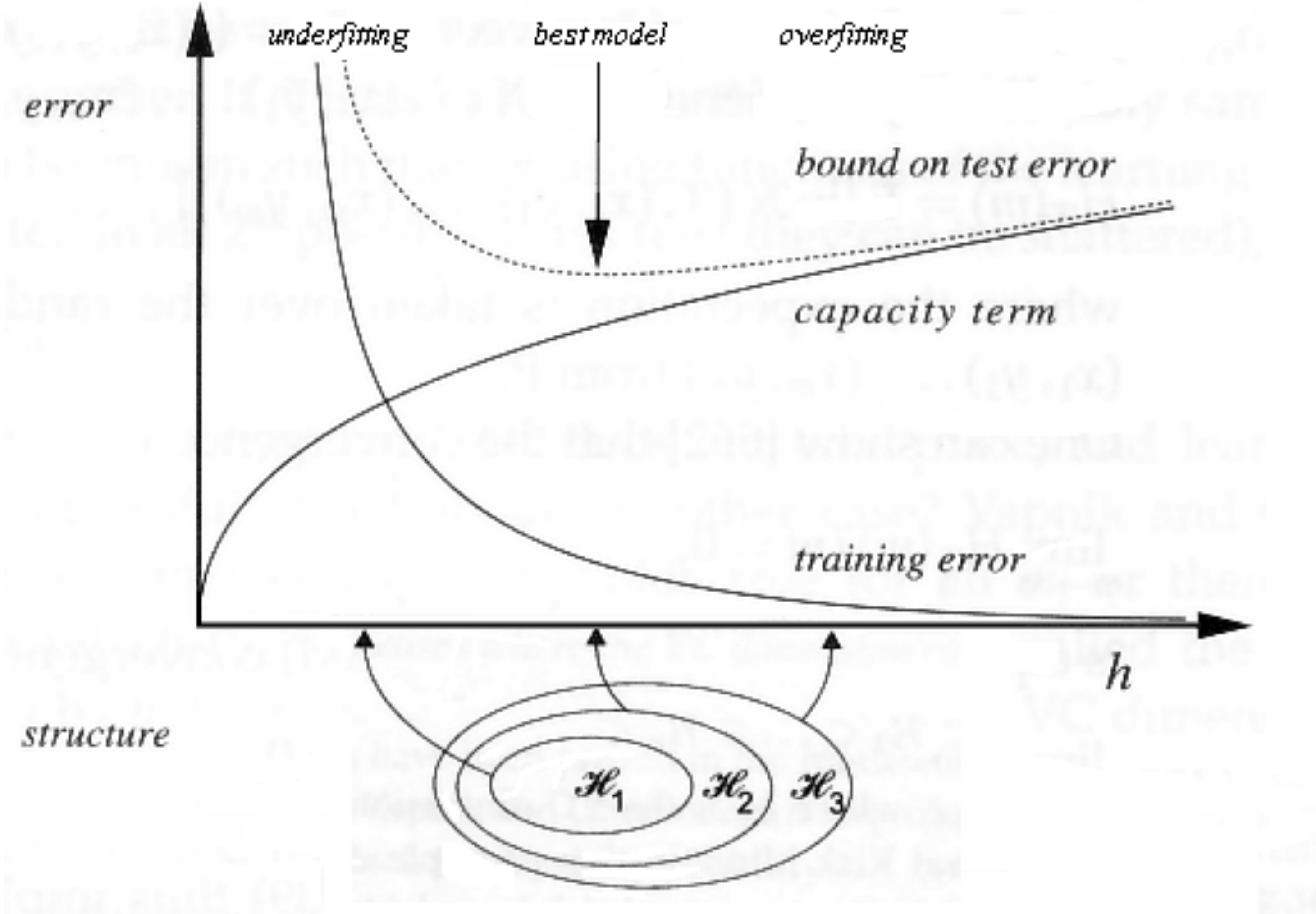


VC confidence is monotonic in h. (here $l = 10{,}000$, $\eta = 0.05$ (95%))

Structural Risk Minimization

$$R(a) \leq R_{emp}(a) + \sqrt{\left(\frac{h(\log(2l/h)+1) - \log(\eta/4)}{l}\right)}$$

Given some selection of learning machines whose empirical risk is zero, one wants to choose that learning machine whose associated set of functions has minimal VC dimension. This is called **Occam's Razor**: "**All things being equal, the simplest solution tends to be the best one**."



http://www.svms.org/srm/

Comments

- The risk bound equation gives a probabilistic upper bound on the actual risk. This does not prevent a particular machine with the same value for empirical risk, and whose function set has higher VC dimension from having better performance.
- For higher h value, the bound is guaranteed not tight.
- $h/l > 0.37$, VC confidence exceeds unity.

Example

- What is the VC dimension of one-nearest neighbor method?
- Nearest neighbor classifier can still perform well.
- For any classifier with an infinite VC dimension, the bound is not even valid.

Structure Risk Minimization for SVM

- Margin (M) is a measure of capacity / complexity of a linear support vector machine
- The objective is to find a linear hyperplane with maximum margin
- Maximum margin classifier

Maximum Margin Classifier

- The optimization problem:

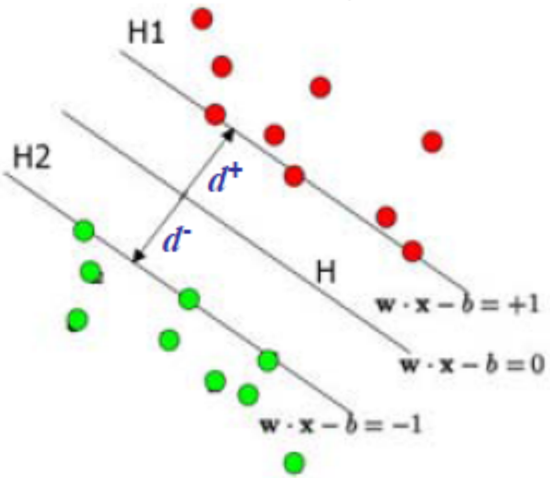
$$\max_{w,b} \quad \frac{1}{\|w\|}$$

$$\text{s.t} \quad y_i(w^T x_i + b) \geq 1, \quad \forall i$$

- The solution to this leads to the famous Support Vector Machines --- believed by many to be the best "off-the-shelf" supervised learning algorithm

Support Vector Machines Optimization

- A convex quadratic programming problem with linear constrains:



$$\max_{w,b} \quad \frac{1}{\|w\|}$$

$$\text{s.t} \quad y_i(w^T x_i + b) \geq 1, \quad \forall i$$

  - The attained margin is now given by $\frac{1}{\|w\|}$
  - Only a few of the classification constraints are relevant ➔ **support vectors**

- Constrained optimization
  - We can directly solve this using commercial quadratic programming (QP) code
  - But we want to take a more careful investigation of Lagrange duality, and the solution of the above in its dual form.

➔ deeper insight: support vectors, kernels …

➔ more efficient algorithm

Lagrange Optimization

- An mathematical optimization technique named after **Joseph Louis Lagrange**
- A method for finding local minima of a function of several variables subject to one or more constraints
- The method reduces a problem in ***n*** variables with ***k*** constraints to a *solvable* problem in ***n+k*** variables with no constraints.
- The method introduces a new unknown scalar variable, the **Lagrange multiplier**, for each constraint and forms a linear combination involving the multipliers as coefficients.

http://en.wikipedia.org/wiki/Lagrange_multipliers

Langrangian Duality

- The Primal Problem

Primal:

$$\begin{aligned} \min_w \quad & f(w) \\ \text{s.t.} \quad & g_i(w) \le 0, \quad i = 1, \ldots, k \\ & h_i(w) = 0, \quad i = 1, \ldots, l \end{aligned}$$

The generalized Lagrangian:

$$\mathcal{L}(w, \alpha, \beta) = f(w) + \sum_{i=1}^{k} \alpha_i g_i(w) + \sum_{i=1}^{l} \beta_i h_i(w)$$

the α's ($\alpha_i \geq 0$) and β's are called the Lagarangian multipliers

Lemma:

$$\max_{\alpha, \beta, \alpha_i \ge 0} \mathcal{L}(w, \alpha, \beta) = \begin{cases} f(w) & \text{if } w \text{ satisfies primal constraints} \\ \infty & \text{o/w} \end{cases}$$

A re-written Primal:

$$\min_w \max_{\alpha, \beta, \alpha_i \ge 0} \mathcal{L}(w, \alpha, \beta)$$

Lagrangian Duality

- Recall the Primal Problem:

$$\min_w \max_{\alpha,\beta,\alpha_i \geq 0} \mathcal{L}(w,\alpha,\beta)$$

- The Dual Problem:

$$\max_{\alpha,\beta,\alpha_i \geq 0} \min_w \mathcal{L}(w,\alpha,\beta)$$

- **Theorem (weak duality):**

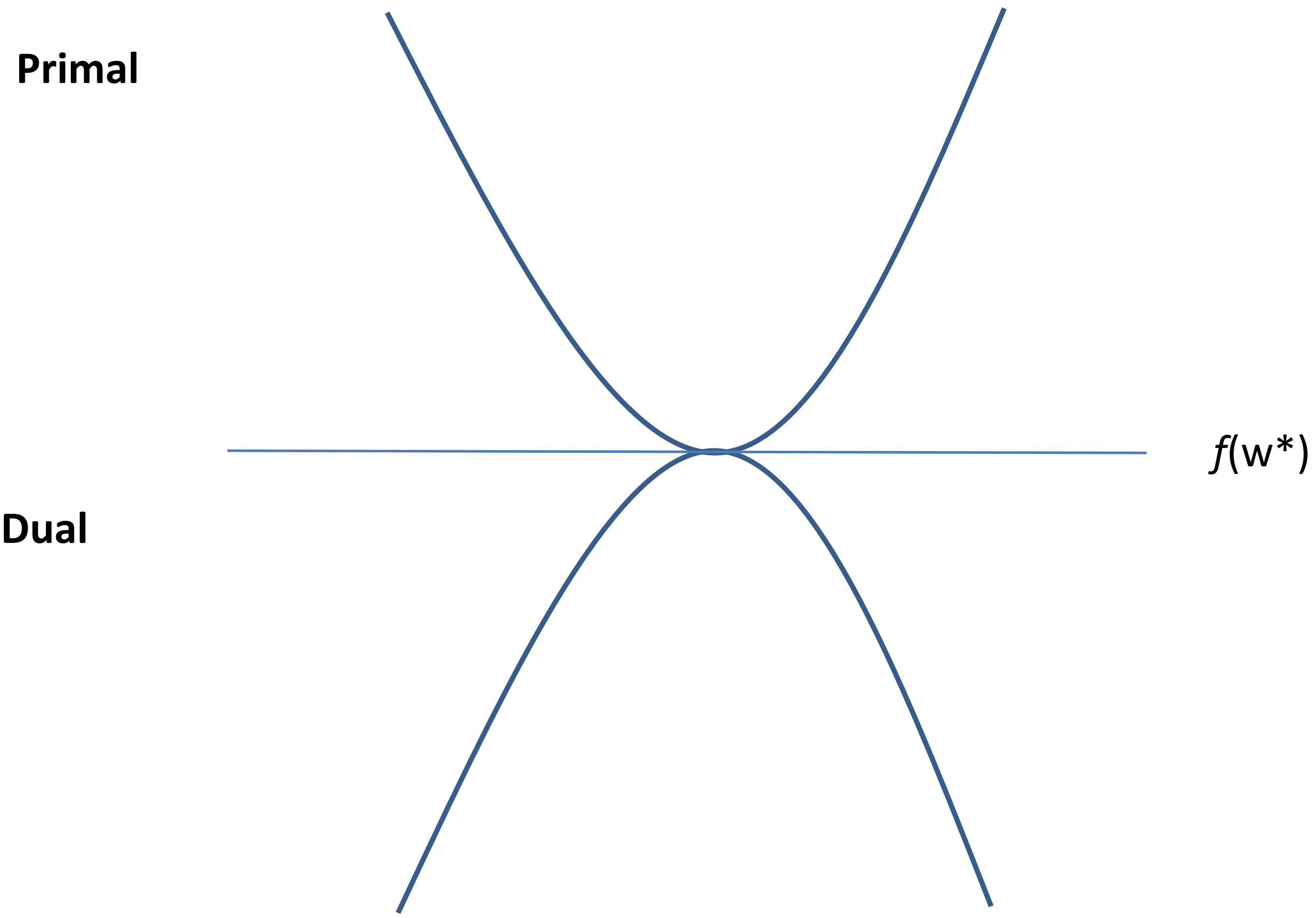
$$d^* = \max_{\alpha,\beta,\alpha_i \geq 0} \min_w \mathcal{L}(w,\alpha,\beta) \leq \min_w \max_{\alpha,\beta,\alpha_i \geq 0} \mathcal{L}(w,\alpha,\beta) = p^*$$

- **Theorem (strong duality):**

  Iff there exist a saddle point of $\mathcal{L}(w,\alpha,\beta)$, we have

$$d^* = p^*$$

Primal and Dual Problems



KKT Condition

- If there exists some saddle point of $\mathcal{L}$, then the saddle point satisfies the following "Karush-Kuhn-Tucker" (KKT) conditions:

$$\frac{\partial}{\partial w_i}\mathcal{L}(w,\alpha,\beta)=0,\quad i=1,\dots,k$$

$$\frac{\partial}{\partial \beta_i}\mathcal{L}(w,\alpha,\beta)=0,\quad i=1,\dots,l$$

$$\alpha_i g_i(w)=0,\quad i=1,\dots,m$$ Complementary slackness

$$g_i(w)\le 0,\quad i=1,\dots,m$$ Primal feasibility

$$\alpha_i \ge 0,\quad i=1,\dots,m$$ Dual feasibility

- **Theorem**: If w^*, α^* and β^* satisfy the KKT condition, then it is also a solution to the primal and the dual problems.

Solve Maximum Margin Classifier

- Recall our opt problem:

$$\begin{aligned} &\max_{w,b} \quad \frac{1}{\|w\|} \\ &\text{s.t} \quad y_i(w^T x_i + b) \geq 1, \quad \forall i \end{aligned}$$

- This is equivalent to

$$\begin{aligned} &\min_{w,b} \quad \frac{1}{2} w^T w \\ &\text{s.t} \quad 1 - y_i(w^T x_i + b) \leq 0, \quad \forall i \end{aligned} \qquad (*)$$

- Write the Lagrangian:

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

  - Recall that (*) can be reformulated as $\min_{w,b} \max_{\alpha_i \geq 0} \mathcal{L}(w,b,\alpha)$
    Now we solve its **dual problem**: $\max_{\alpha_i \geq 0} \min_{w,b} \mathcal{L}(w,b,\alpha)$

The Dual Problem

$$\max_{\alpha_i \geq 0} \min_{w,b} \mathcal{L}(w,b,\alpha)$$

- We minimize $\mathcal{L}$ with respect to w and b first:

$$\nabla_w \mathcal{L}(w,b,\alpha) = w - \sum_{i=1}^{m} \alpha_i y_i x_i = 0, \qquad (*)$$

$$\nabla_b \mathcal{L}(w,b,\alpha) = \sum_{i=1}^{m} \alpha_i y_i = 0, \qquad (**)$$

Note that (*) implies:

$$w = \sum_{i=1}^{m} \alpha_i y_i x_i \qquad (***)$$

- Plus (***) back to $\mathcal{L}$, and using (**), we have:

$$\mathcal{L}(w,b,\alpha) = \sum_{i=1}^{m} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{m} \alpha_i \alpha_j y_i y_j (\mathbf{x}_i^T \mathbf{x}_j)$$

Proof

$$\frac{1}{2}w^T w - \sum_{i=1}^{m} a_i(y_i(w^T x_i + b) \; - 1)$$

Replace w with $\sum_{i=1}^{m} a_i y_i x_i$, we get:

$$=\frac{1}{2}\sum_{i=1}^{m} a_i y_i x_i \sum_{i=1}^{m} a_i y_i x_i - \sum_{i=1}^{m} a_i(y_i((\sum_{i=1}^{m} a_i y_i x_i)x_i + b) - 1)$$

$$= \frac{1}{2}\sum_{i=1}^{m} a_i y_i x_i \sum_{i=1}^{m} a_i y_i x_i - (\sum_{i=1}^{m} a_i y_i(\sum_{i=1}^{m} a_i y_i x_i)x_i + \sum_{i=1}^{m} a_i y_i \, b - \sum_{i=1}^{m} a_i)$$

$$= -\frac{1}{2}\sum_{i,j=1}^{m} a_i a_j y_i y_j x_i^T x_j \quad + \sum_{i=1}^{m} a_i$$

The Dual Problem

- Now we have the following dual opt problem:

$$\max_\alpha \mathcal{J}(\alpha) = \sum_{i=1}^{m} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{m} \alpha_i \alpha_j y_i y_j (\mathbf{x}_i^T \mathbf{x}_j)$$

$$\text{s.t.} \quad \alpha_i \geq 0, \quad i = 1, \ldots, k$$

$$\sum_{i=1}^{m} \alpha_i y_i = 0.$$

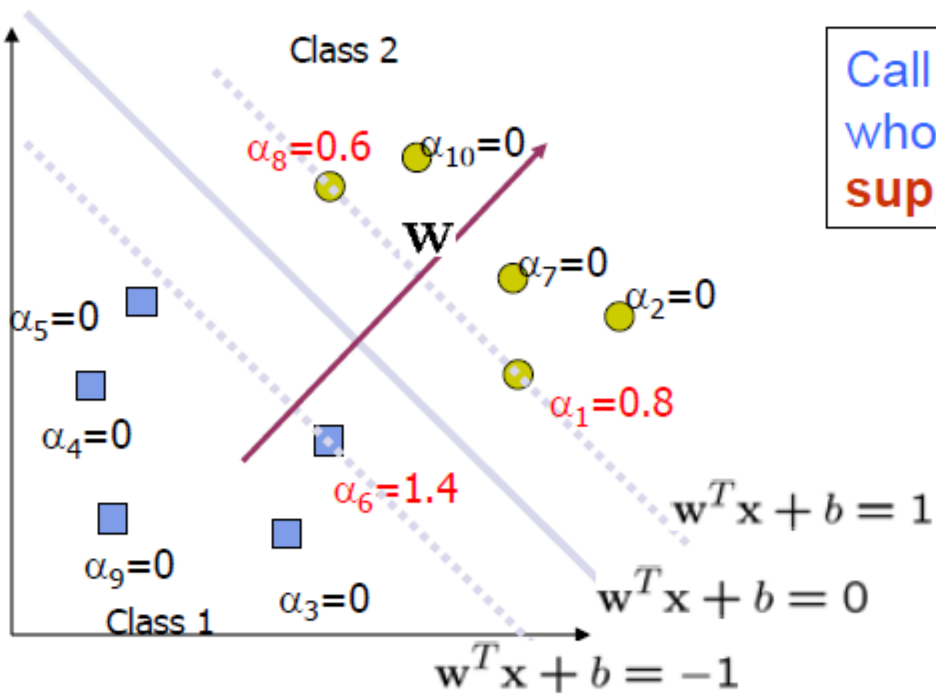
- This is, (again,) a **quadratic programming** problem.
  - A global maximum of α_i can always be found.
  - But what's the big deal??
  - Note two things:

1. w can be recovered by $\quad w = \sum_{i=1}^{m} \alpha_i y_i \mathbf{x}_i \quad$ See next …
2. The "kernel" $\quad \mathbf{x}_i^T \mathbf{x}_j \quad$ More later …

Support Vectors

- Note the KKT condition --- only a few α_i's can be nonzero!!

$$\alpha_i g_i(w) = 0, \quad i = 1, \ldots, m$$



Call the training data points whose α_i's are nonzero the **support vectors** (SV)

Support Vector Machines

- Once we have the Lagrange multipliers $\{\alpha_i\}$, we can reconstruct the parameter vector w as a weighted combination of the training examples:

$$w = \sum_{i \in SV} \alpha_i y_i \mathbf{x}_i$$

Question: how to get b?

- For testing with a new data z
  - Compute
  $$w^T z + b = \sum_{i \in SV} \alpha_i y_i \left(\mathbf{x}_i^T z \right) + b$$
  and classify z as class 1 if the sum is positive, and class 2 otherwise
  - Note: w need not be formed explicitly

How to Determine w and b

- Use quadratic programming to solve a_i and compute w is trivial. (use KKT condition (1))
- How to compute b?
- Use KKT condition (5), for any support vector (point $a_i > 0$), $y_i(w.x_i+b)-1 = 0$.
- We compute b in terms of a support vector. Better: we computer b in terms of all support vectors and take the average.

Interpretation of Support Vector Machines

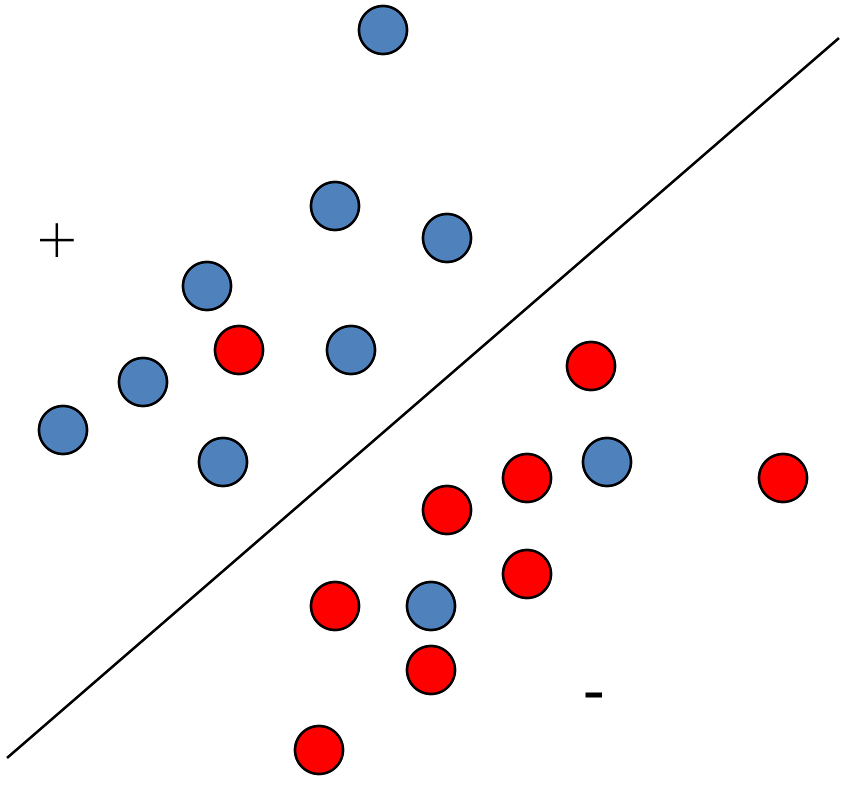
- The optimal $\boldsymbol{w}$ is a linear combination of a small number of data points. This "sparse" representation can be viewed as data compression as in the construction of kNN classifier

- To compute the weights $\{\alpha_i\}$, and to use support vector machines we need to specify only the inner products (or kernel) between the examples $\mathbf{x}_i^T \mathbf{x}_j$

- We make decisions by comparing each new example z with only the support vectors:

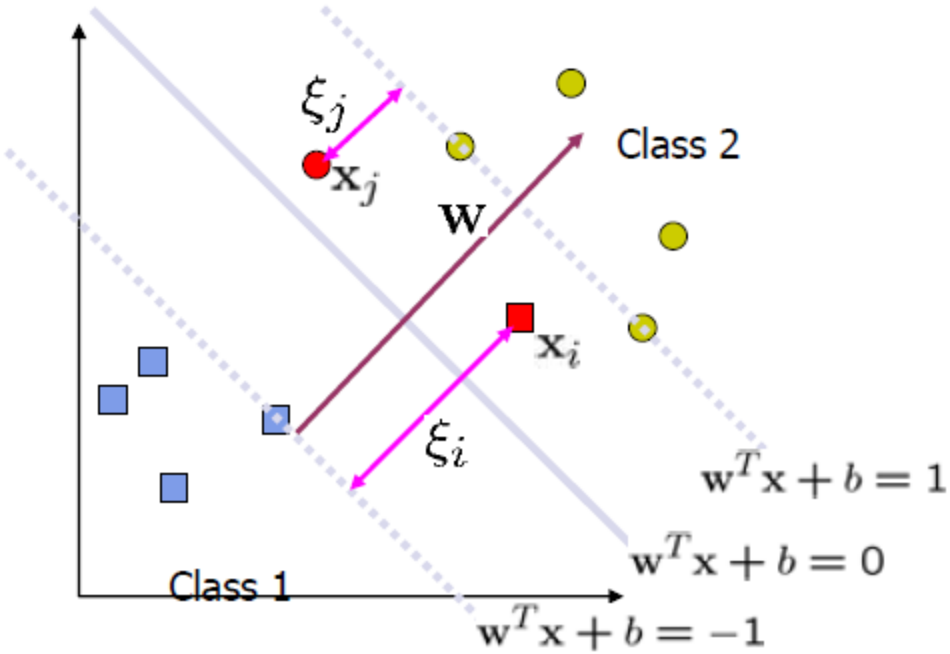
$$y^* = \operatorname{sign}\left(\sum_{i \in SV} \alpha_i y_i \left(\mathbf{x}_i^T z \right) + b \right)$$

Non-Separable Case



Can't satisfy the constraints $y_i(wx_i+b) >= 1$ for some data points? What can we do?

Non-Linearly Separable Problem



- We allow "error" ξ_i in classification; it is based on the output of the discriminant function $\boldsymbol{w}^T\boldsymbol{x}+b$
- ξ_i approximates the number of misclassified samples

Relax Constraints – Soft Margin

- Introduce positive slack variables ξ_i, i = 1, ..., l to relax constraints. ($\xi_i >= 0$)
- New constraints:
- $x_i.w + b >= +1 - \xi_i$ for $y_i = +1$
- $x_i.w + b <= -1 + \xi_i$ for $y_i = -1$
- Or $\mathbf{y_i(wx_i + b) >= 1 - \xi_i}$
- $\xi_i >= 0$
- For an classification error to happen, the corresponding ξ_i must exceed unity, so **$\Sigma\ \xi_i$ is an upper bound on the number of training errors**.

Soft Margin Hyperplane

- Now we have a slightly different opt problem:

$$\min_{w,b} \quad \frac{1}{2} w^T w + C \sum_{i=1}^{m} \xi_i$$

$$\text{s.t} \quad \begin{aligned} & y_i(w^T x_i + b) \geq 1 - \xi_i, \quad \forall i \\ & \xi_i \geq 0, \quad \forall i \end{aligned}$$

- ξ_i are "slack variables" in optimization
- Note that ξ_i=0 if there is no error for $\mathbf{x}_i$
- ξ_i is an upper bound of the number of errors
- C : tradeoff parameter between error and margin

Primal Optimization

$$L_P = \frac{1}{2}|w|^2 + C\sum_i \xi_i - \sum_{i=1}^{m} a_i(y_i(x_i.w+b)-1+\xi_i)) - \sum_{i=1}^{m} u_i\xi_i$$

$$\frac{\partial L_p}{\partial \xi_i} = C - a_i - u_i = 0$$

$$\Rightarrow a_i <= C$$

u_i is the Lagrange multipliers introduced to enforce Non-negativity of ξ_i

KKT Conditions

$$
\begin{array}{ll}
1.\partial_{\mathbf{w}}\mathcal{L}_P = 0 \quad \rightarrow & \mathbf{w} - \sum_i \alpha_i y_i \mathbf{x}_i = 0 \\
2.\partial_b\mathcal{L}_P = 0 \quad \rightarrow & \sum_i \alpha_i y_i = 0 \\
3.\partial_\xi\mathcal{L}_P = 0 \quad \rightarrow & C - \alpha_i - \mu_i = 0 \\
4.\text{constraint-1} & y_i(\mathbf{w}^T\mathbf{x}_i - b) - 1 + \xi_i \geq 0 \\
5.\text{constraint-2} & \xi_i \geq 0 \\
6.\text{multiplier condition-1} & \alpha_i \geq 0 \\
7.\text{multiplier condition-2} & \mu_i \geq 0 \\
8.\text{complementary slackness-1} & \alpha_i \left[y_i(\mathbf{w}^T\mathbf{x}_i - b) - 1 + \xi_i\right] = 0 \\
9.\text{complementary slackness-1} & \mu_i \xi_i = 0
\end{array}
$$

Max Welling, 2005

Proof of Soft Margin Optimization

$$L_P = \frac{1}{2}|w|^2 + C\sum_i \xi_i - \sum_{i=1}^{m} a_i(y_i(x_i.w+b)-1+\xi_i)) - \sum_{i=1}^{m} u_i\xi_i$$

$$= \frac{1}{2}|w|^2 - \sum_{i=1}^{m} a_i(y_i(x_i.w+b)-1) - \sum_{i=1}^{m}(C-a_i-u_i)\xi_i$$

$$= ?$$

The Optimization Problem

- The dual of this new constrained optimization problem is

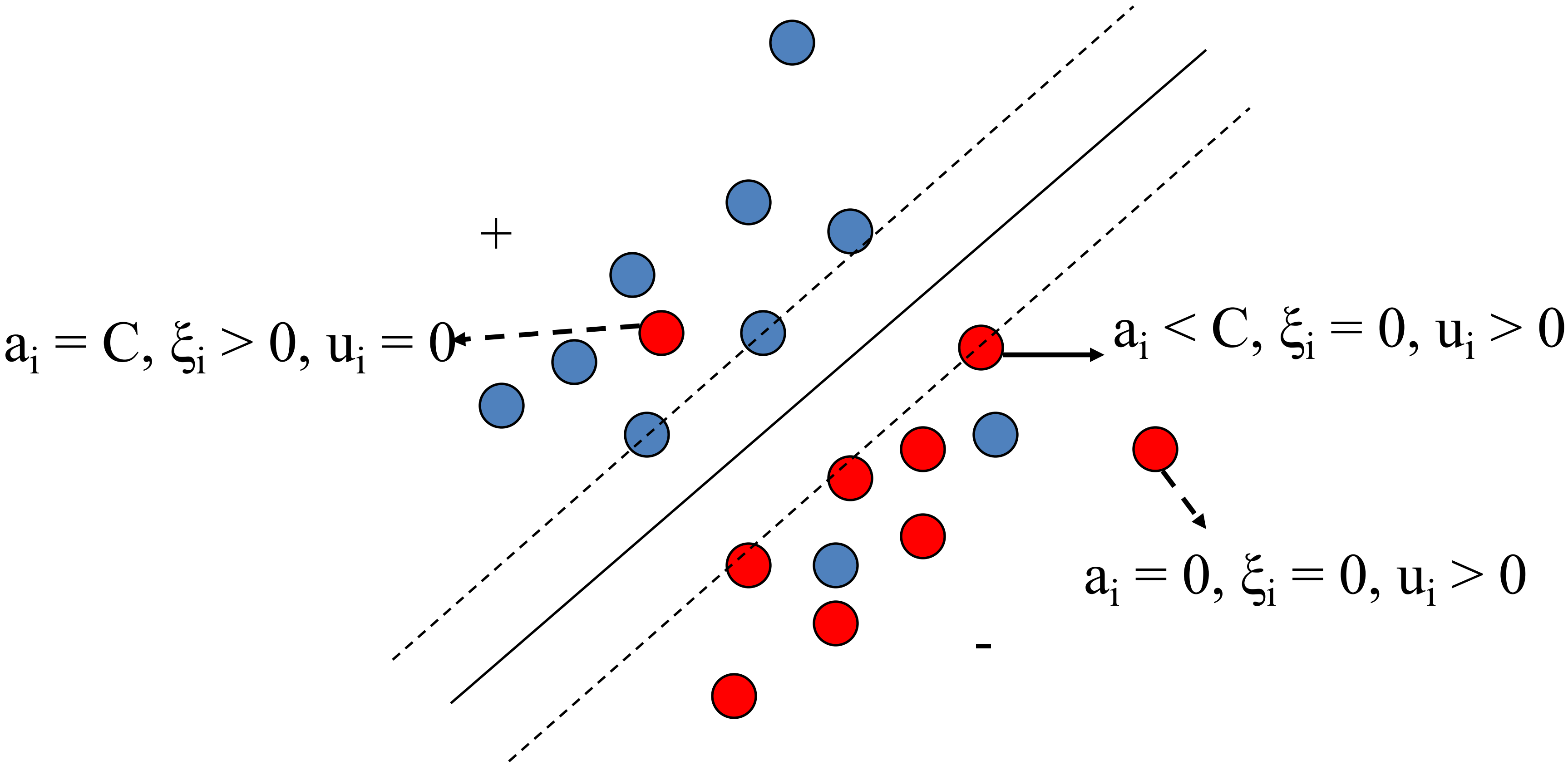
$$\max_{\alpha} \quad \mathcal{J}(\alpha) = \sum_{i=1}^{m} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{m} \alpha_i \alpha_j y_i y_j (\mathbf{x}_i^T \mathbf{x}_j)$$

$$\text{s.t.} \quad 0 \leq \alpha_i \leq C, \quad i = 1, \ldots, m$$

$$\sum_{i=1}^{m} \alpha_i y_i = 0.$$

- This is very similar to the optimization problem in the linear separable case, except that there is an upper bound *C* on α_i now
- Once again, a QP solver can be used to find α_i

Values of Multipliers



Solution of w and b

$$w = \sum_{i=1}^{Ns} a_i y_i x_i$$

Use complementary slackness to compute b. Choose a support vector ($0 < a_i < C$) to compute b, where $\xi_i = 0$. $\xi_i = 0$ is derived by combining equations 3 and 9.

New Objective Function

- Minimize $|w|^2/2 + C(\Sigma\xi_i)^k$.
- C is parameter to be chosen by the user, a larger C corresponding to assigning a higher penalty to errors.
- This is a convex programming problem for any positive integer k.

SVM Demo

https://www.youtube.com/watch?v=bqwAlpumoPM

http://cs.stanford.edu/people/karpathy/svmj
s/demo/