

Support Vector Machines

Note to other teachers and users of these slides. Andrew would be delighted if you found this source material useful in giving your own lectures. Feel free to use these slides verbatim, or to modify them to fit your own needs. PowerPoint originals are available. If you make use of a significant portion of these slides in your own lecture, please include this message, or the following link to the source repository of Andrew's tutorials: <http://www.cs.cmu.edu/~awm/tutorials>. Comments and corrections gratefully received.

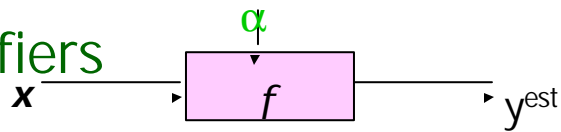
Andrew W. Moore
Associate Professor
School of Computer Science
Carnegie Mellon University

www.cs.cmu.edu/~awm
awm@cs.cmu.edu
412-268-7599

Copyright © 2001, Andrew W. Moore

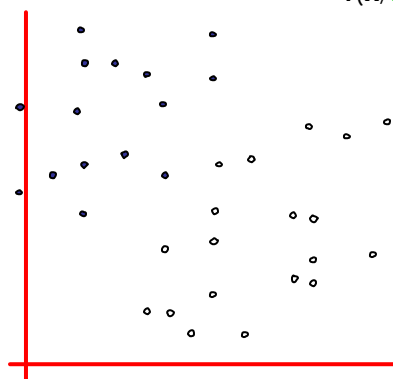
Nov 23rd, 2001

Linear Classifiers



$$f(x, w, b) = \text{sign}(w \cdot x - b)$$

- denotes +1
- denotes -1

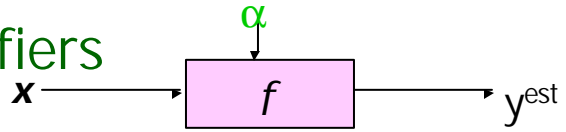


How would you classify this data?

Copyright © 2001, Andrew W. Moore

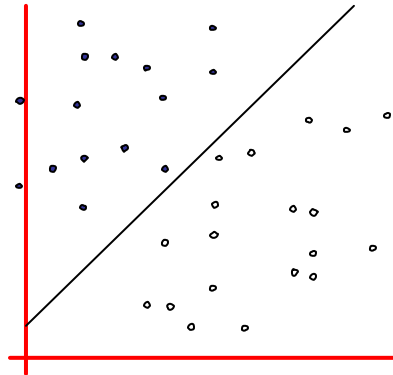
Support Vector Machines: Slide 2

Linear Classifiers



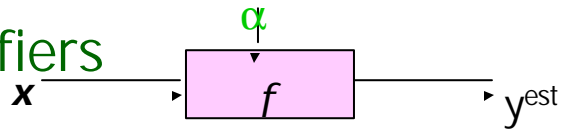
- denotes +1
- denotes -1

$$f(x, w, b) = \text{sign}(w \cdot x - b)$$



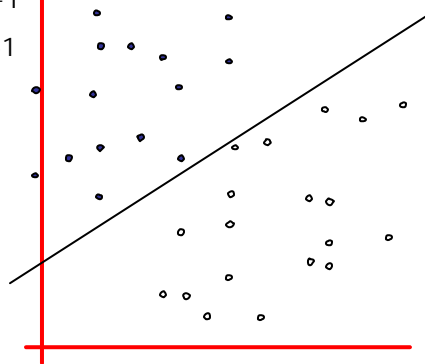
How would you classify this data?

Linear Classifiers



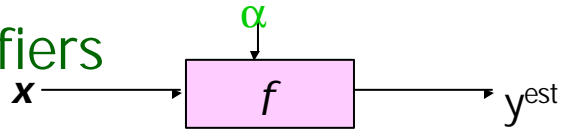
- denotes +1
- denotes -1

$$f(x, w, b) = \text{sign}(w \cdot x - b)$$



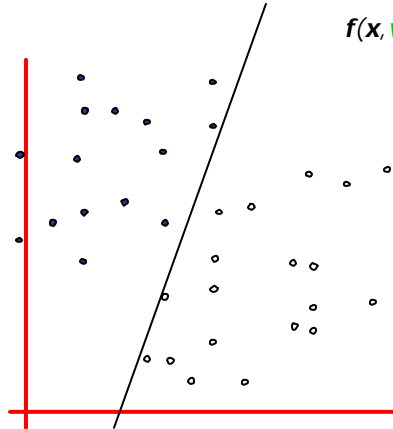
How would you classify this data?

Linear Classifiers



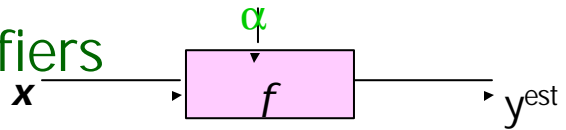
$$f(x, w, b) = \text{sign}(w \cdot x - b)$$

- denotes +1
- denotes -1



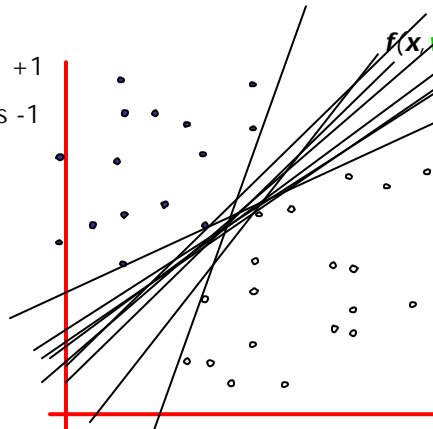
How would you classify this data?

Linear Classifiers



$$f(x, w, b) = \text{sign}(w \cdot x - b)$$

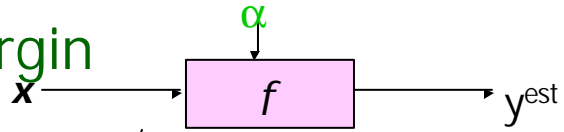
- denotes +1
- denotes -1



Any of these would be fine..

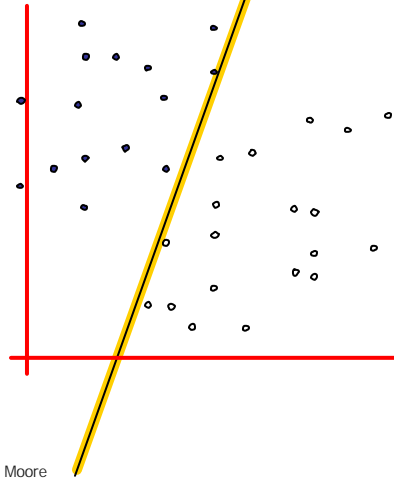
..but which is best?

Classifier Margin



$$f(\mathbf{x}, \mathbf{w}, b) = \text{sign}(\mathbf{w} \cdot \mathbf{x} - b)$$

- denotes +1
- denotes -1

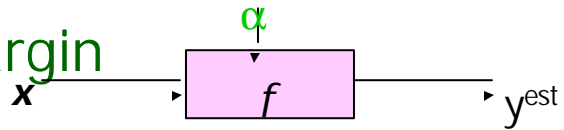


Define the **margin** of a linear classifier as the width that the boundary could be increased by before hitting a datapoint.

Copyright © 2001, Andrew W. Moore

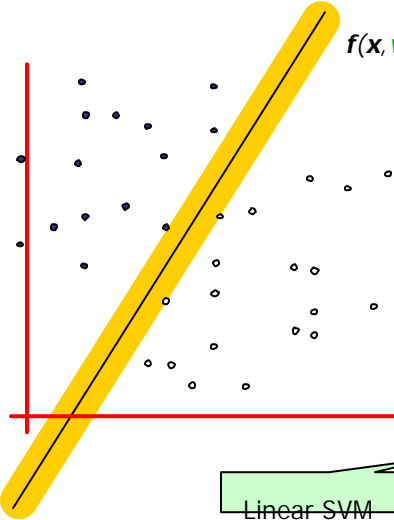
Support Vector Machines: Slide 7

Maximum Margin



$$f(\mathbf{x}, \mathbf{w}, b) = \text{sign}(\mathbf{w} \cdot \mathbf{x} - b)$$

- denotes +1
- denotes -1



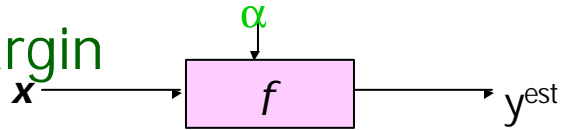
The **maximum margin linear classifier** is the linear classifier with the, um, maximum margin.

This is the simplest kind of SVM (Called an LSVM)

Copyright © 2001, Andrew W. Moore

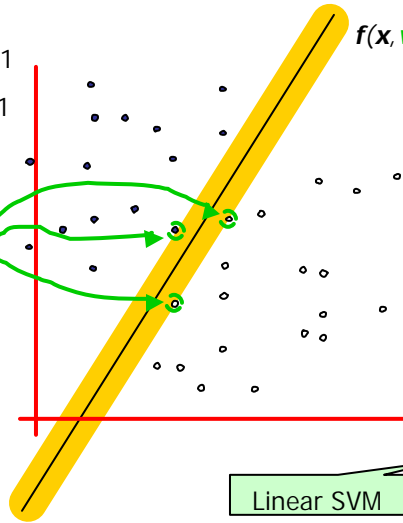
Support Vector Machines: Slide 8

Maximum Margin



- denotes +1
- denotes -1

Support Vectors are those datapoints that the margin pushes up against



$$f(\mathbf{x}, \mathbf{w}, b) = \text{sign}(\mathbf{w} \cdot \mathbf{x} - b)$$

The **maximum margin linear classifier** is the linear classifier with the, um, maximum margin.

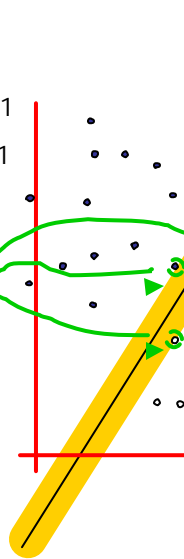
This is the simplest kind of SVM (Called an LSVM)

Linear SVM

Why Maximum Margin?

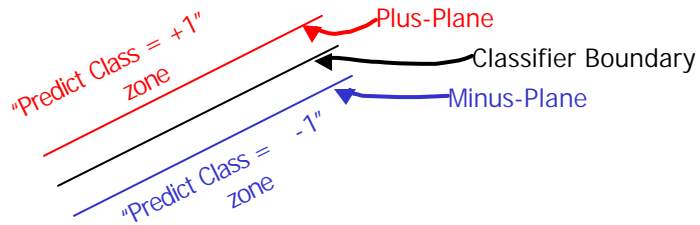
- denotes +1
- denotes -1

Support Vectors are those datapoints that the margin pushes up against



1. Intuitively this feels safest.
2. If we've made a small error in the location of the boundary (it's been jolted in its perpendicular direction) this gives us least chance of causing a misclassification.
3. LOOCV is easy since the model is immune to removal of any non-support-vector datapoints.
4. There's some theory (using VC dimension) that is related to (but not the same as) the proposition that this is a good thing.
5. Empirically it works very very well.

Specifying a line and margin

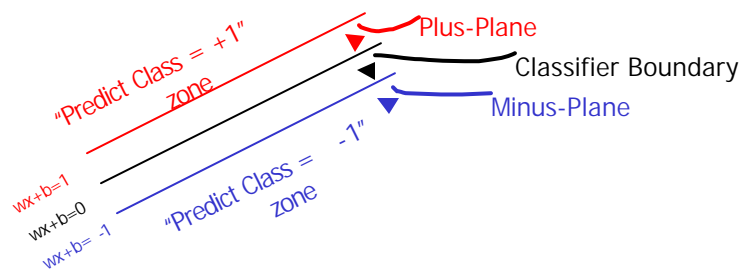


- How do we represent this mathematically?
- ...in m input dimensions?

Copyright © 2001, Andrew W. Moore

Support Vector Machines: Slide 11

Specifying a line and margin



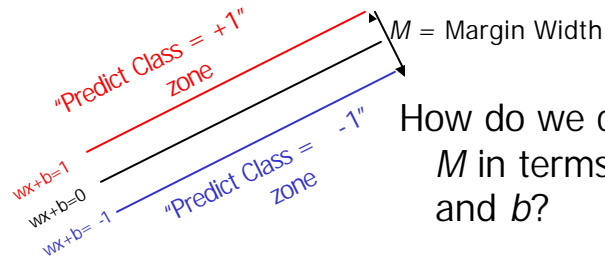
- Plus-plane = $\{ \mathbf{x} : \mathbf{w} \cdot \mathbf{x} + b = +1 \}$
- Minus-plane = $\{ \mathbf{x} : \mathbf{w} \cdot \mathbf{x} + b = -1 \}$

Classify as.. $+1$ if $\mathbf{w} \cdot \mathbf{x} + b \geq 1$
 -1 if $\mathbf{w} \cdot \mathbf{x} + b \leq -1$
 Universe explodes if $-1 < \mathbf{w} \cdot \mathbf{x} + b < 1$

Copyright © 2001, Andrew W. Moore

Support Vector Machines: Slide 12

Computing the margin width

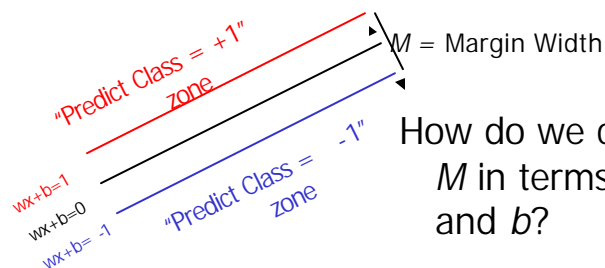


How do we compute M in terms of \mathbf{w} and b ?

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

Claim: The vector \mathbf{w} is perpendicular to the Plus Plane. **Why?**

Computing the margin width



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

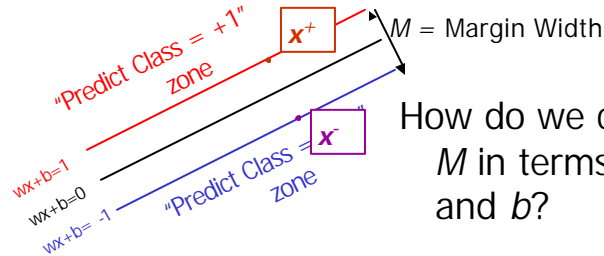
- Plus-plane = $\{ \mathbf{x} : \mathbf{w} \cdot \mathbf{x} + b = +1 \}$
- Minus-plane = $\{ \mathbf{x} : \mathbf{w} \cdot \mathbf{x} + b = -1 \}$

Claim: The vector \mathbf{w} is perpendicular to the Plus Plane. **Why?**

Let \mathbf{u} and \mathbf{v} be two vectors on the Plus Plane. What is $\mathbf{w} \cdot (\mathbf{u} - \mathbf{v})$?

And so of course the vector \mathbf{w} is also perpendicular to the Minus Plane

Computing the margin width



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

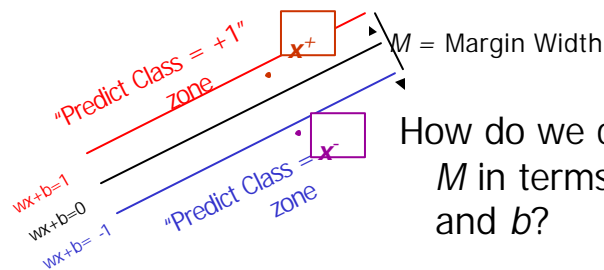
- Plus-plane = $\{ \mathbf{x} : \mathbf{w} \cdot \mathbf{x} + b = +1 \}$
- Minus-plane = $\{ \mathbf{x} : \mathbf{w} \cdot \mathbf{x} + b = -1 \}$
- The vector \mathbf{w} is perpendicular to the Plus Plane
- Let \mathbf{x}^- be any point on the minus plane
- Let \mathbf{x}^+ be the closest plus-plane-point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

Any location in \mathbb{R}^m : not necessarily a datapoint

Copyright © 2001, Andrew W. Moore

Support Vector Machines: Slide 15

Computing the margin width



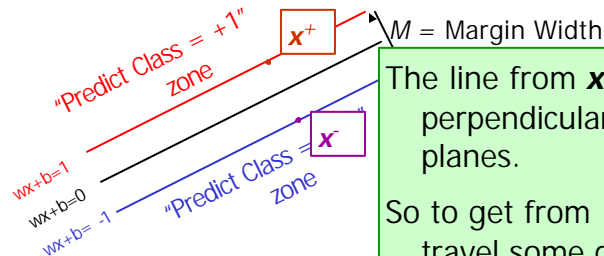
How do we compute M in terms of \mathbf{w} and b ?

- Plus-plane = $\{ \mathbf{x} : \mathbf{w} \cdot \mathbf{x} + b = +1 \}$
- Minus-plane = $\{ \mathbf{x} : \mathbf{w} \cdot \mathbf{x} + b = -1 \}$
- The vector \mathbf{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:** $\mathbf{x}^+ = \mathbf{x}^- + \mathbf{I} \mathbf{w}$ for some value of \mathbf{I} . **Why?**

Copyright © 2001, Andrew W. Moore

Support Vector Machines: Slide 16

Computing the margin width



The line from \mathbf{x}^- to \mathbf{x}^+ is perpendicular to the planes.

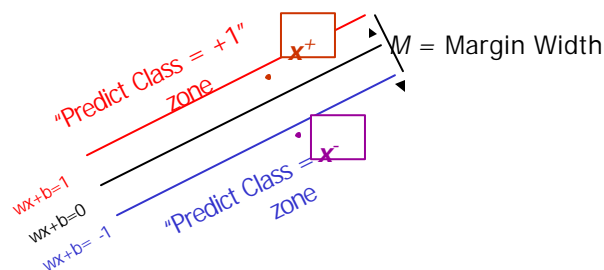
So to get from \mathbf{x}^- to \mathbf{x}^+ travel some distance in direction \mathbf{w} .

- Plus-plane = $\{ \mathbf{x} : \mathbf{w} \cdot \mathbf{x} + b = 1 \}$
- Minus-plane = $\{ \mathbf{x} : \mathbf{w} \cdot \mathbf{x} + b = -1 \}$
- The vector \mathbf{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:** $\mathbf{x}^+ = \mathbf{x}^- + \mathbf{l} \mathbf{w}$ for some value of \mathbf{l} . **Why?**

Copyright © 2001, Andrew W. Moore

Support Vector Machines: Slide 17

Computing the margin width



What we know:

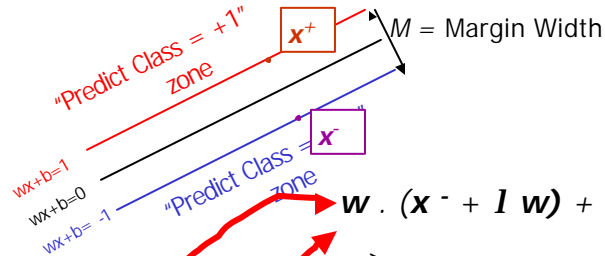
- $\mathbf{w} \cdot \mathbf{x}^+ + b = +1$
- $\mathbf{w} \cdot \mathbf{x}^- + b = -1$
- $\mathbf{x}^+ = \mathbf{x}^- + \mathbf{l} \mathbf{w}$
- $|\mathbf{x}^+ - \mathbf{x}^-| = M$

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

Copyright © 2001, Andrew W. Moore

Support Vector Machines: Slide 18

Computing the margin width



What we know:

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

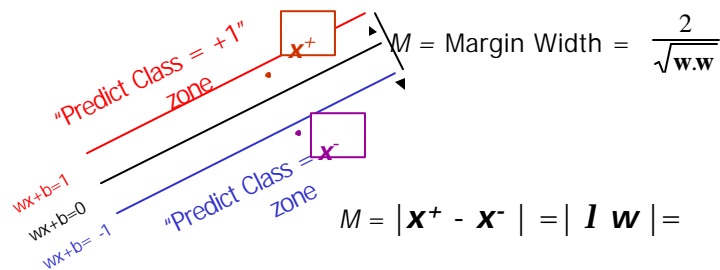
It's now easy to get M
in terms of \mathbf{w} and b

$$\begin{aligned} \mathbf{w} \cdot (\mathbf{x}^- + \mathbf{l} \mathbf{w}) + b &= 1 \\ \Rightarrow \mathbf{w} \cdot \mathbf{x}^- + b + \mathbf{l} \mathbf{w} \cdot \mathbf{w} &= 1 \\ \Rightarrow -1 + \mathbf{l} \mathbf{w} \cdot \mathbf{w} &= 1 \\ \Rightarrow ? &= \frac{2}{\mathbf{w} \cdot \mathbf{w}} \end{aligned}$$

Copyright © 2001, Andrew W. Moore

Support Vector Machines: Slide 19

Computing the margin width



What we know:

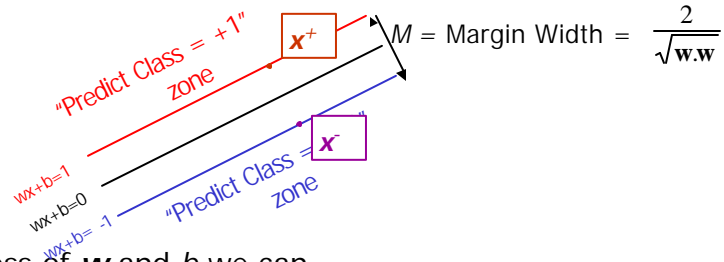
- $\mathbf{w} \cdot \mathbf{x}^+ + b = +1$
- $\mathbf{w} \cdot \mathbf{x}^- + b = -1$
- $\mathbf{x}^+ = \mathbf{x}^- + \mathbf{l} \mathbf{w}$
- $|\mathbf{x}^+ - \mathbf{x}^-| = M$
- $? = \frac{2}{\mathbf{w} \cdot \mathbf{w}}$

$$\begin{aligned} M &= |\mathbf{x}^+ - \mathbf{x}^-| = |\mathbf{l} \mathbf{w}| = \\ &= ? |\mathbf{w}| = ? \sqrt{\mathbf{w} \cdot \mathbf{w}} \\ &= \frac{2\sqrt{\mathbf{w} \cdot \mathbf{w}}}{\mathbf{w} \cdot \mathbf{w}} = \frac{2}{\sqrt{\mathbf{w} \cdot \mathbf{w}}} \end{aligned}$$

Copyright © 2001, Andrew W. Moore

Support Vector Machines: Slide 20

Learning the Maximum Margin Classifier



Given a guess of \mathbf{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 \mathbf{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?

Copyright © 2001, Andrew W. Moore

Support Vector Machines: Slide 21

Learning via Quadratic Programming

- QP is a well-studied class of optimization algorithms to maximize a quadratic function of some real-valued variables subject to linear constraints.

Copyright © 2001, Andrew W. Moore

Support Vector Machines: Slide 22

Quadratic Programming

Find $\arg \max_{\mathbf{u}} c + \mathbf{d}^T \mathbf{u} + \frac{\mathbf{u}^T \mathbf{R} \mathbf{u}}{2}$ ← Quadratic criterion

Subject to

$$\begin{aligned} a_{11}u_1 + a_{12}u_2 + \dots + a_{1m}u_m &\leq b_1 \\ a_{21}u_1 + a_{22}u_2 + \dots + a_{2m}u_m &\leq b_2 \\ &\vdots \\ a_{n1}u_1 + a_{n2}u_2 + \dots + a_{nm}u_m &\leq b_n \end{aligned}$$

} n additional linear inequality constraints

And subject to

$$\begin{aligned} a_{(n+1)1}u_1 + a_{(n+1)2}u_2 + \dots + a_{(n+1)m}u_m &= b_{(n+1)} \\ a_{(n+2)1}u_1 + a_{(n+2)2}u_2 + \dots + a_{(n+2)m}u_m &= b_{(n+2)} \\ &\vdots \\ a_{(n+e)1}u_1 + a_{(n+e)2}u_2 + \dots + a_{(n+e)m}u_m &= b_{(n+e)} \end{aligned}$$

} e additional linear equality constraints

Copyright © 2001, Andrew W. Moore

Support Vector Machines: Slide 23

Quadratic Programming

Find $\arg \max_{\mathbf{u}} c + \mathbf{d}^T \mathbf{u} + \frac{\mathbf{u}^T \mathbf{R} \mathbf{u}}{2}$ ← Quadratic criterion

Subject to

$$\begin{aligned} a_{11}u_1 + a_{12}u_2 + \dots + a_{1m}u_m &\leq b_1 \\ a_{21}u_1 + a_{22}u_2 + \dots + a_{2m}u_m &\leq b_2 \\ &\vdots \\ a_{n1}u_1 + a_{n2}u_2 + \dots + a_{nm}u_m &\leq b_n \end{aligned}$$

} n additional linear inequality constraints

And subject to

$$\begin{aligned} a_{(n+1)1}u_1 + a_{(n+1)2}u_2 + \dots + a_{(n+1)m}u_m &= b_{(n+1)} \\ a_{(n+2)1}u_1 + a_{(n+2)2}u_2 + \dots + a_{(n+2)m}u_m &= b_{(n+2)} \\ &\vdots \\ a_{(n+e)1}u_1 + a_{(n+e)2}u_2 + \dots + a_{(n+e)m}u_m &= b_{(n+e)} \end{aligned}$$

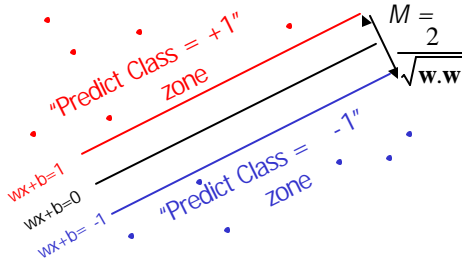
} e additional linear equality constraints

There exist algorithms for finding such constrained quadratic optima much more efficiently and reliably than gradient ascent.
 (But they are very fiddly...you probably don't want to write one yourself)

Copyright © 2001, Andrew W. Moore

Support Vector Machines: Slide 24

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

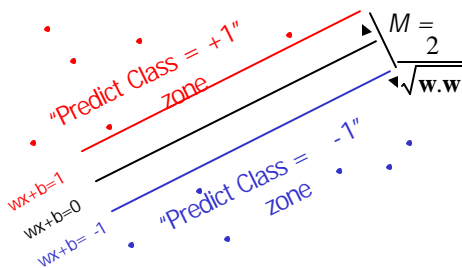
Assume R datapoints, each (\mathbf{x}_k, y_k) where $y_k = +/- 1$

What should our quadratic optimization criterion be?

How many constraints will we have?

What should they be?

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 $\mathbf{w} \cdot \mathbf{w}$

How many constraints will we have? R

What should they be?

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

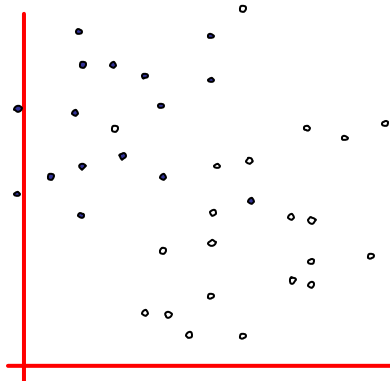
$$\mathbf{w} \cdot \mathbf{x}_k + b \leq -1 \text{ if } y_k = -1$$

Uh-oh!

This is going to be a problem!

What should we do?

- denotes +1
- denotes -1



Copyright © 2001, Andrew W. Moore

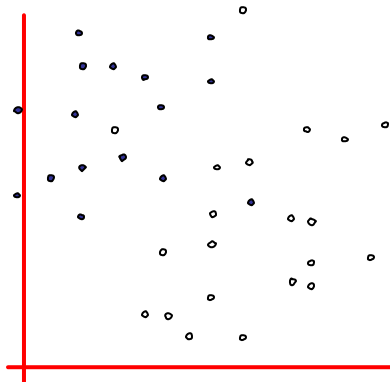
Support Vector Machines: Slide 27

Uh-oh!

This is going to be a problem!

What should we do?

- denotes +1
- denotes -1



Idea 1:

Find minimum $\mathbf{w} \cdot \mathbf{w}$, while minimizing number of training set errors.

Problem: Two things to minimize makes for an ill-defined optimization

Copyright © 2001, Andrew W. Moore

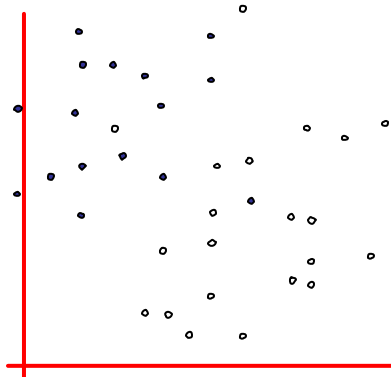
Support Vector Machines: Slide 28

Uh-oh!

This is going to be a problem!

What should we do?

- denotes +1
- denotes -1



Idea 1.1:

Minimize

$$w \cdot w + C (\# \text{train errors})$$

Tradeoff parameter

There's a serious practical problem that's about to make us reject this approach. Can you guess what it is?

Copyright © 2001, Andrew W. Moore

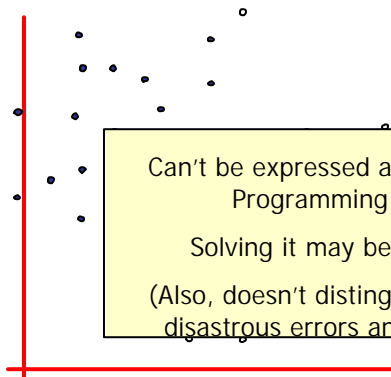
Support Vector Machines: Slide 29

Uh-oh!

This is going to be a problem!

What should we do?

- denotes +1
- denotes -1



Idea 1.1:

Minimize

$$w \cdot w + C (\# \text{train errors})$$

Tradeoff parameter

Can't be expressed as a Quadratic Programming problem.
Solving it may be too slow.
(Also, doesn't distinguish between disastrous errors and near misses)

us reject this approach. Can you guess what it is?



Copyright © 2001, Andrew W. Moore

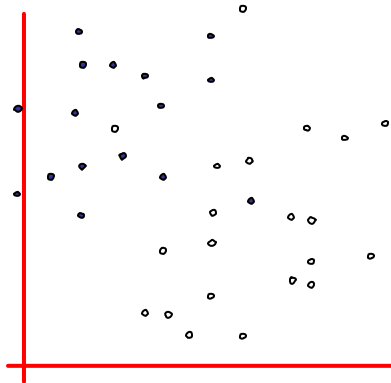
Support Vector Machines: Slide 30

Uh-oh!

This is going to be a problem!

What should we do?

- denotes +1
- denotes -1



Idea 2.0:

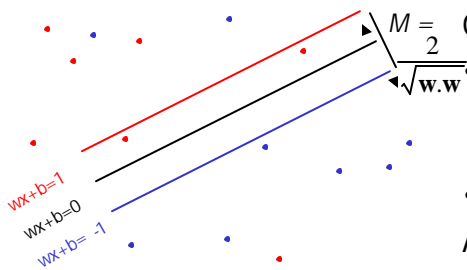
Minimize

$\mathbf{w} \cdot \mathbf{w} + C$ (distance of error points to their correct place)

Copyright © 2001, Andrew W. Moore

Support Vector Machines: Slide 31

Learning Maximum Margin with Noise



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

- Compute sum of distances of points to their correct zones

- Compute the margin width

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

What should our quadratic optimization criterion be?

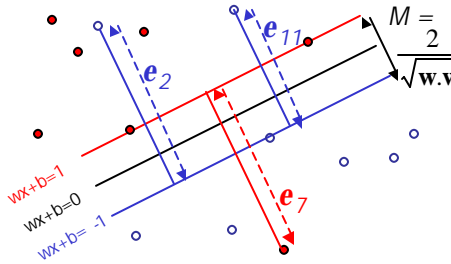
How many constraints will we have?

What should they be?

Copyright © 2001, Andrew W. Moore

Support Vector Machines: Slide 32

Learning Maximum Margin with Noise



- Given guess of \mathbf{w} , b we can
- Compute sum of distances of points to their correct zones
 - Compute the margin width
- Assume R datapoints, each (\mathbf{x}_k, y_k) where $y_k = +/- 1$

What should our quadratic optimization criterion be?

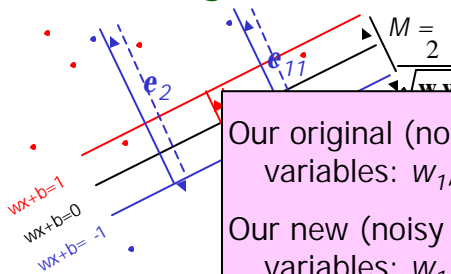
Minimize
$$\frac{1}{2} \mathbf{w} \cdot \mathbf{w} + C \sum_{k=1}^R e_k$$

How many constraints will we have? R

What should they be?

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

Learning Maximum Margin with Noise



- Given guess of \mathbf{w} , b we can
- Compute sum of distances
- th
- Our original (noiseless data) QP had $m+1$ variables: w_1, w_2, \dots, w_m and b .
- Our new (noisy data) QP has $m+1+R$ variables: $w_1, w_2, \dots, w_m, b, e_1, e_2, \dots, e_R$
- (\mathbf{x}_k, y_k) where $y_k = +/- 1$

What should our quadratic optimization criterion be?

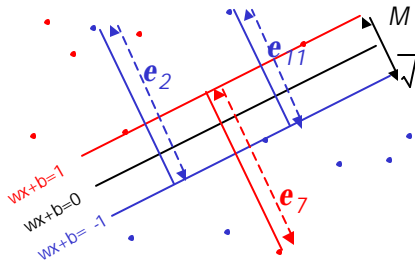
Minimize
$$\frac{1}{2} \mathbf{w} \cdot \mathbf{w} + C \sum_{k=1}^R e_k$$

How many constraints will we have? R

What should they be?

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

Learning Maximum Margin with Noise



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

- Compute sum of distances of points to their correct zones

- 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
$$\frac{1}{2} \mathbf{w} \cdot \mathbf{w} + C \sum_{k=1}^R e_k$$

How many constraints will we have? R

What should they be?

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

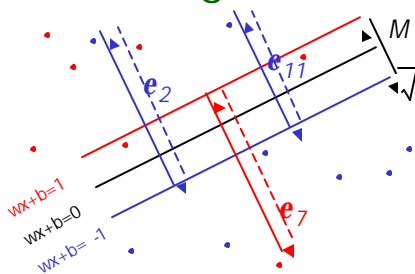
$$\mathbf{w} \cdot \mathbf{x}_k + b \leq -1 + e_k \text{ if } y_k = -1$$

There's a bug in this QP. Can you spot it?

Copyright © 2001, Andrew W. Moore

Support Vector Machines: Slide 35

Learning Maximum Margin with Noise



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

- Compute sum of distances of points to their correct zones

- 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
$$\frac{1}{2} \mathbf{w} \cdot \mathbf{w} + C \sum_{k=1}^R e_k$$

How many constraints will we have? $2R$

What should they be?

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

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

$$e_k \geq 0 \text{ for all } k$$

Copyright © 2001, Andrew W. Moore

Support Vector Machines: Slide 36

An Equivalent QP

$$\text{Maximize } \sum_{k=1}^R a_k + \sum_{k=1}^R \sum_{l=1}^R a_k a_l Q_{kl} \text{ where } Q_{kl} = y_k y_l (\mathbf{x}_k \cdot \mathbf{x}_l)$$

$$\text{Subject to these constraints: } 0 \leq a_k \leq C \quad \forall k \quad \sum_{k=1}^R a_k y_k = 0$$

Then define:

$$\mathbf{w} = \sum_{k=1}^R a_k y_k \mathbf{x}_k$$

$$b = y_K (1 - e_K) - \mathbf{x}_K \cdot \mathbf{w}$$

where $K = \arg \max_k a_k$

Then classify with:

$$f(\mathbf{x}, \mathbf{w}, b) = \text{sign}(\mathbf{w} \cdot \mathbf{x} - b)$$

Copyright © 2001, Andrew W. Moore

Support Vector Machines: Slide 37

An Equivalent QP

$$\text{Maximize } \sum_{k=1}^R a_k + \sum_{k=1}^R \sum_{l=1}^R a_k a_l Q_{kl} \text{ where } Q_{kl} = y_k y_l (\mathbf{x}_k \cdot \mathbf{x}_l)$$

$$\text{Subject to these constraints: } 0 \leq a_k \leq C \quad \forall k \quad \sum_{k=1}^R a_k y_k = 0$$

Then define:

$$\mathbf{w} = \sum_{k=1}^R a_k y_k \mathbf{x}_k$$

$$b = y_K (1 - e_K) - \mathbf{x}_K \cdot \mathbf{w}$$

where $K = \arg \max_k a_k$

Datapoints with $a_k > 0$ will be the support vectors.

..so this sum only needs to be over the support vectors.

Copyright © 2001, Andrew W. Moore

Support Vector Machines: Slide 38

An Equivalent QP

Maximize $\sum_{k=1}^R a_k$ $y_k y_l (\mathbf{x}_k \cdot \mathbf{x}_l)$

Subject to $a_k \geq 0$

Then $\mathbf{w} = \sum_{k=1}^R a_k \mathbf{x}_k$

$b = y_k (1 - \text{margin}_k)$

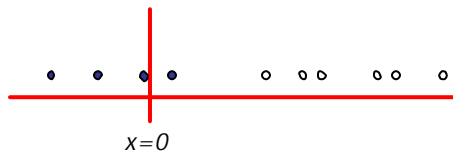
where $K = \text{diag}(\mathbf{A})$

Why did I tell you about this equivalent QP?

- It's a formulation that QP packages can optimize more quickly
- Because of further jaw-dropping developments you're about to learn.

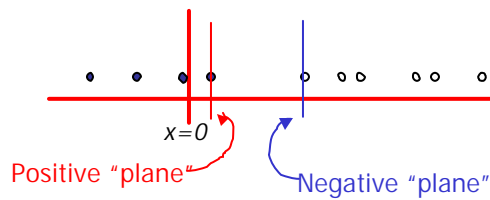
Suppose we're in 1-dimension

What would SVMs do with this data?



Suppose we're in 1-dimension

Not a big surprise



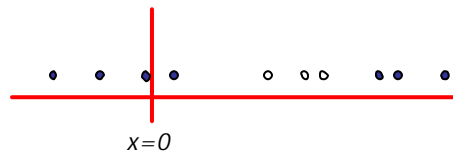
Copyright © 2001, Andrew W. Moore

Support Vector Machines: Slide 41

Harder 1-dimensional dataset

That's wiped the smirk off SVM's face.

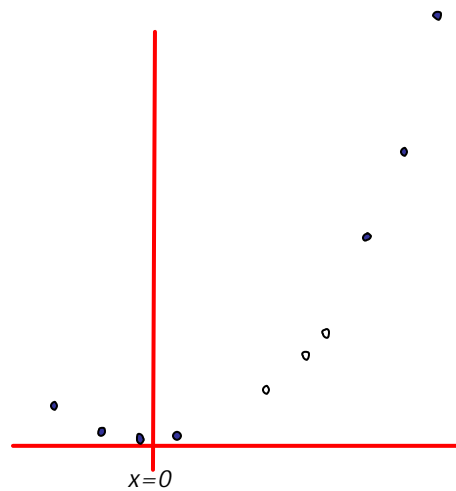
What can be done about this?



Copyright © 2001, Andrew W. Moore

Support Vector Machines: Slide 42

Harder 1-dimensional dataset



Remember how
permitting non-
linear basis
functions made
linear regression
so much nicer?

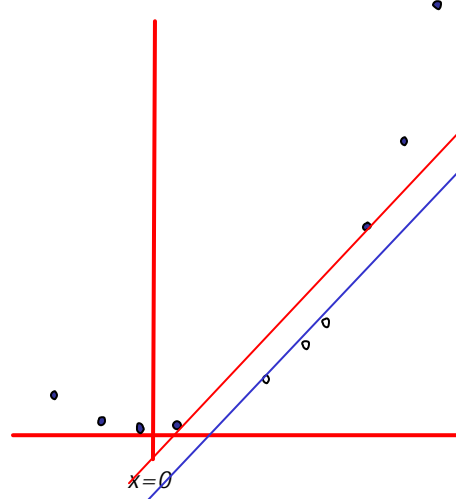
Let's permit them
here too

$$\mathbf{z}_k = (x_k, x_k^2)$$

Copyright © 2001, Andrew W. Moore

Support Vector Machines: Slide 43

Harder 1-dimensional dataset



Remember how
permitting non-
linear basis
functions made
linear regression
so much nicer?

Let's permit them
here too

$$\mathbf{z}_k = (x_k, x_k^2)$$

Copyright © 2001, Andrew W. Moore

Support Vector Machines: Slide 44

Common SVM basis functions

$\mathbf{z}_k =$ (polynomial terms of \mathbf{x}_k of degree 1 to q)

$\mathbf{z}_k =$ (radial basis functions of \mathbf{x}_k)

$$\mathbf{z}_k[j] = f_j(\mathbf{x}_k) = \text{KernelFn}\left(\frac{|\mathbf{x}_k - \mathbf{c}_j|}{KW}\right)$$

$\mathbf{z}_k =$ (sigmoid functions of \mathbf{x}_k)

This is sensible.

Is that the end of the story?

No...there's one more trick!

Copyright © 2001, Andrew W. Moore

Support Vector Machines: Slide 45

$\mathbf{F}(\mathbf{x}) =$

1	<div style="display: flex; flex-direction: column; align-items: center;"> <div style="margin-bottom: 5px;">} Constant Term</div> <div style="margin-bottom: 5px;">} Linear Terms</div> <div style="margin-bottom: 5px;">} Pure Quadratic Terms</div> <div style="margin-bottom: 5px;">} Quadratic Cross-Terms</div> </div>
$\sqrt{2}x_1$	
$\sqrt{2}x_2$	
⋮	
$\sqrt{2}x_m$	
x_1^2	
x_2^2	
⋮	
x_m^2	
$\sqrt{2}x_1x_2$	
$\sqrt{2}x_1x_3$	
⋮	
$\sqrt{2}x_1x_m$	
$\sqrt{2}x_2x_3$	
⋮	
$\sqrt{2}x_1x_m$	
⋮	
$\sqrt{2}x_{m-1}x_m$	

Quadratic Basis Functions

Number of terms (assuming m input dimensions) = $(m+2)\text{-choose-}2$

= $(m+2)(m+1)/2$

= (as near as makes no difference) $m^2/2$

You may be wondering what those $\sqrt{2}$'s are doing.

- You should be happy that they do no harm
- You'll find out why they're there soon.

Copyright © 2001, Andrew W. Moore

Support Vector Machines: Slide 46

QP with basis functions

Maximize $\sum_{k=1}^R a_k + \sum_{k=1}^R \sum_{l=1}^R a_k a_l Q_{kl}$ where $Q_{kl} = y_k y_l (\mathbf{F}(\mathbf{x}_k) \cdot \mathbf{F}(\mathbf{x}_l))$

Subject to these constraints: $0 \leq a_k \leq C \quad \forall k \quad \sum_{k=1}^R a_k y_k = 0$

Then define:

$$\mathbf{w} = \sum_{k \text{ s.t. } a_k > 0} a_k y_k \mathbf{F}(\mathbf{x}_k)$$

$$b = y_K (1 - e_K) - \mathbf{x}_K \cdot \mathbf{w}_K$$

where $K = \arg \max_k a_k$

Then classify with:

$$f(\mathbf{x}, \mathbf{w}, b) = \text{sign}(\mathbf{w} \cdot \mathbf{f}(\mathbf{x}) - b)$$

Copyright © 2001, Andrew W. Moore

Support Vector Machines: Slide 47

QP with basis functions

Maximize $\sum_{k=1}^R a_k + \sum_{k=1}^R \sum_{l=1}^R a_k a_l Q_{kl}$ where $Q_{kl} = y_k y_l (\mathbf{F}(\mathbf{x}_k) \cdot \mathbf{F}(\mathbf{x}_l))$

Subject to these constraints: $0 \leq a_k \leq$

Then define:

$$\mathbf{w} = \sum_{k \text{ s.t. } a_k > 0} a_k y_k \mathbf{F}(\mathbf{x}_k)$$

$$b = y_K (1 - e_K) - \mathbf{x}_K \cdot \mathbf{w}_K$$

where $K = \arg \max_k a_k$

We must do $R^2/2$ dot products to get this matrix ready.

Each dot product requires $m^2/2$ additions and multiplications

The whole thing costs $R^2 m^2 / 4$.
Yeeks!

...or does it?

$$f(\mathbf{x}, \mathbf{w}, b) = \text{sign}(\mathbf{w} \cdot \mathbf{f}(\mathbf{x}) - b)$$

Copyright © 2001, Andrew W. Moore

Support Vector Machines: Slide 48

Quadratic Dot Products

$$\mathbf{F}(\mathbf{a}) \cdot \mathbf{F}(\mathbf{b}) =$$

$$\begin{pmatrix} 1 \\ \sqrt{2}a_1 \\ \sqrt{2}a_2 \\ \vdots \\ \sqrt{2}a_m \\ a_1^2 \\ a_2^2 \\ \vdots \\ a_m^2 \\ \sqrt{2}a_1a_2 \\ \sqrt{2}a_1a_3 \\ \vdots \\ \sqrt{2}a_1a_m \\ \sqrt{2}a_2a_3 \\ \vdots \\ \sqrt{2}a_1a_m \\ \vdots \\ \sqrt{2}a_{m-1}a_m \end{pmatrix} \cdot \begin{pmatrix} 1 \\ \sqrt{2}b_1 \\ \sqrt{2}b_2 \\ \vdots \\ \sqrt{2}b_m \\ b_1^2 \\ b_2^2 \\ \vdots \\ b_m^2 \\ \sqrt{2}b_1b_2 \\ \sqrt{2}b_1b_3 \\ \vdots \\ \sqrt{2}b_1b_m \\ \sqrt{2}b_2b_3 \\ \vdots \\ \sqrt{2}b_1b_m \\ \vdots \\ \sqrt{2}b_{m-1}b_m \end{pmatrix}$$

$$\begin{aligned} & \left. \begin{array}{l} 1 \\ + \\ \sum_{i=1}^m 2a_i b_i \end{array} \right\} \\ & + \\ & \left. \sum_{i=1}^m a_i^2 b_i^2 \right\} \\ & + \\ & \left. \sum_{i=1}^m \sum_{j=i+1}^m 2a_i a_j b_i b_j \right\} \end{aligned}$$

Copyright © 2001, Andrew W. Moore

Support Vector Machines: Slide 49

Quadratic Dot Products

$$\mathbf{F}(\mathbf{a}) \cdot \mathbf{F}(\mathbf{b}) =$$

$$1 + 2 \sum_{i=1}^m a_i b_i + \sum_{i=1}^m a_i^2 b_i^2 + \sum_{i=1}^m \sum_{j=i+1}^m 2a_i a_j b_i b_j$$

Just out of casual, innocent, interest, let's look at another function of \mathbf{a} and \mathbf{b} :

$$(\mathbf{a} \cdot \mathbf{b} + 1)^2$$

$$= (\mathbf{a} \cdot \mathbf{b})^2 + 2\mathbf{a} \cdot \mathbf{b} + 1$$

$$= \left(\sum_{i=1}^m a_i b_i \right)^2 + 2 \sum_{i=1}^m a_i b_i + 1$$

$$= \sum_{i=1}^m \sum_{j=1}^m a_i b_i a_j b_j + 2 \sum_{i=1}^m a_i b_i + 1$$

$$= \sum_{i=1}^m (a_i b_i)^2 + 2 \sum_{i=1}^m \sum_{j=i+1}^m a_i b_i a_j b_j + 2 \sum_{i=1}^m a_i b_i + 1$$

Copyright © 2001, Andrew W. Moore

Support Vector Machines: Slide 50

Quadratic Dot Products

$$\mathbf{F}(\mathbf{a}) \cdot \mathbf{F}(\mathbf{b}) = 1 + 2 \sum_{i=1}^m a_i b_i + \sum_{i=1}^m a_i^2 b_i^2 + \sum_{i=1}^m \sum_{j=i+1}^m 2a_i a_j b_i b_j$$

Just out of casual, innocent, interest, let's look at another function of \mathbf{a} and \mathbf{b} :

$$\begin{aligned} & (\mathbf{a} \cdot \mathbf{b} + 1)^2 \\ &= (\mathbf{a} \cdot \mathbf{b})^2 + 2\mathbf{a} \cdot \mathbf{b} + 1 \\ &= \left(\sum_{i=1}^m a_i b_i \right)^2 + 2 \sum_{i=1}^m a_i b_i + 1 \\ &= \sum_{i=1}^m \sum_{j=1}^m a_i b_i a_j b_j + 2 \sum_{i=1}^m a_i b_i + 1 \\ &= \sum_{i=1}^m (a_i b_i)^2 + 2 \sum_{i=1}^m \sum_{j=i+1}^m a_i b_i a_j b_j + 2 \sum_{i=1}^m a_i b_i + 1 \end{aligned}$$

They're the same!

And this is only $O(m)$ to compute!

QP with Quadratic basis functions

$$\text{Maximize } \sum_{k=1}^R a_k + \sum_{k=1}^R \sum_{l=1}^R a_k a_l Q_{kl} \text{ where } Q_{kl} = y_k y_l (\mathbf{F}(\mathbf{x}_k) \cdot \mathbf{F}(\mathbf{x}_l))$$

Subject to these constraints: $0 \leq a_k \leq 1$

We must do $R^2/2$ dot products to get this matrix ready.

Each dot product now only requires m additions and multiplications

Then define:

$$\mathbf{w} = \sum_{k \text{ s.t. } a_k > 0} a_k y_k \mathbf{F}(\mathbf{x}_k)$$

$$b = y_K (1 - e_K) - \mathbf{x}_K \cdot \mathbf{w}_K$$

where $K = \arg \max_k a_k$

Then classify with:

$$f(\mathbf{x}, \mathbf{w}, b) = \text{sign}(\mathbf{w} \cdot \mathbf{f}(\mathbf{x}) - b)$$

Higher Order Polynomials

Poly-nomial	$f(\mathbf{x})$	Cost to build Q_{kl} matrix traditionally	Cost if 100 inputs	$f(\mathbf{a}) \cdot f(\mathbf{b})$	Cost to build Q_{kl} matrix sneakily	Cost if 100 inputs
Quadratic	All $m^2/2$ terms up to degree 2	$m^2 R^2 / 4$	$2,500 R^2$	$(\mathbf{a} \cdot \mathbf{b} + 1)^2$	$m R^2 / 2$	$50 R^2$
Cubic	All $m^3/6$ terms up to degree 3	$m^3 R^2 / 12$	$83,000 R^2$	$(\mathbf{a} \cdot \mathbf{b} + 1)^3$	$m R^2 / 2$	$50 R^2$
Quartic	All $m^4/24$ terms up to degree 4	$m^4 R^2 / 48$	$1,960,000 R^2$	$(\mathbf{a} \cdot \mathbf{b} + 1)^4$	$m R^2 / 2$	$50 R^2$

Copyright © 2001, Andrew W. Moore

Support Vector Machines: Slide 53

OP with Quintic basis functions

We must do $R^2/2$ dot products to get this matrix ready.
 In 100-d, each dot product now needs 103 operations instead of 75 million
 But there are still worrying things lurking away.
What are they?

$$Q_{kl} = y_k y_l (\mathbf{F}(\mathbf{x}_k) \cdot \mathbf{F}(\mathbf{x}_l))$$

constraints:

$$\sum_{k=1}^R a_k y_k = 0$$

Then define:

$$\mathbf{w} = \sum_{k \text{ s.t. } a_k > 0} a_k y_k \mathbf{F}(\mathbf{x}_k)$$

$$b = y_K (1 - e_K) - \mathbf{x}_K \cdot \mathbf{w}_K$$

where $K = \arg \max_k a_k$

Then classify with:

$$f(\mathbf{x}, \mathbf{w}, b) = \text{sign}(\mathbf{w} \cdot \mathbf{f}(\mathbf{x}) - b)$$

Copyright © 2001, Andrew W. Moore

Support Vector Machines: Slide 54

QP with Quintic basis functions

We must do $R^2/2$ dot products to get this matrix ready.

In 100-d, each dot product now needs 103 operations instead of 75 million

But there are still worrying things lurking away. What are they?

constraints:

$$Q_{kl} = y_k y_l (\mathbf{F}(\mathbf{x}_k) \cdot \mathbf{F}(\mathbf{x}_l))$$

$$\forall k \quad \sum_{l=1}^R a_l y_l = 0$$

- The fear of overfitting with this enormous number of terms

- The evaluation phase (doing a set of predictions on a test set) will be very expensive (*why?*)

Then define:

$$\mathbf{w} = \sum_{k \text{ s.t. } a_k > 0} a_k y_k \mathbf{F}(\mathbf{x}_k)$$

$$b = y_K (1 - e_K) - \mathbf{x}_K \cdot \mathbf{w}_K$$

where $K = \arg \max_k a_k$

Then classify with:

$$f(\mathbf{x}, \mathbf{w}, b) = \text{sign}(\mathbf{w} \cdot \mathbf{f}(\mathbf{x}) - b)$$

Copyright © 2001, Andrew W. Moore

Support Vector Machines: Slide 55

QP with Quintic basis functions

We must do $R^2/2$ dot products to get this matrix ready.

In 100-d, each dot product now needs 103 operations instead of 75 million

But there are still worrying things lurking away. What are they?

constraints:

$$Q_{kl} = y_k y_l (\mathbf{F}(\mathbf{x}_k) \cdot \mathbf{F}(\mathbf{x}_l))$$

$$\forall k \quad \sum_{l=1}^R a_l y_l = 0$$

The use of Maximum Margin *magically* makes this not a problem

- The fear of overfitting with this enormous number of terms

- The evaluation phase (doing a set of predictions on a test set) will be very expensive (*why?*)

Then define:

$$\mathbf{w} = \sum_{k \text{ s.t. } a_k > 0} a_k y_k \mathbf{F}(\mathbf{x}_k)$$

$$b = y_K (1 - e_K) - \mathbf{x}_K \cdot \mathbf{w}_K$$

where $K = \arg \max_k a_k$

Because each $\mathbf{w} \cdot \mathbf{f}(\mathbf{x})$ (see below) needs 75 million operations. *What can be done?*

Then classify with:

$$f(\mathbf{x}, \mathbf{w}, b) = \text{sign}(\mathbf{w} \cdot \mathbf{f}(\mathbf{x}) - b)$$

Copyright © 2001, Andrew W. Moore

Support Vector Machines: Slide 56

QP with Quintic basis functions

We must do $R^2/2$ dot products to get this matrix ready.

In 100-d, each dot product now needs 103 operations instead of 75 million

But there are still worrying things lurking away. What are they?

constraints:

$$Q_{ij} = y_i y_j (\mathbf{F}(\mathbf{x}_i) \cdot \mathbf{F}(\mathbf{x}_j))$$

The use of Maximum Margin **magically** makes this not a problem

$$\sum_k a_k y_k = 0$$

•The fear of overfitting with this enormous number of terms

•The evaluation phase (doing a set of predictions on a test set) will be very expensive (why?)

Then define:

$$\mathbf{w} = \sum_{k \text{ s.t. } a_k > 0} a_k y_k \mathbf{F}(\mathbf{x}_k)$$

Because each $\mathbf{w} \cdot \mathbf{f}(\mathbf{x})$ (see below) needs 75 million operations. What can be done?

$$\begin{aligned} \mathbf{w} \cdot \mathbf{F}(\mathbf{x}) &= \sum_{k \text{ s.t. } a_k > 0} a_k y_k \mathbf{F}(\mathbf{x}_k) \cdot \mathbf{F}(\mathbf{x}) \\ &= \sum_{k \text{ s.t. } a_k > 0} a_k y_k (\mathbf{x}_k \cdot \mathbf{x} + 1)^5 \end{aligned}$$

Then classify with:

$$f(\mathbf{x}, \mathbf{w}, b) = \text{sign}(\mathbf{w} \cdot \mathbf{f}(\mathbf{x}) - b)$$

Only S m operations (S =#support vectors)

Copyright © 2001, Andrew W. Moore

Support Vector Machines: Slide 57

QP with Quintic basis functions

We must do $R^2/2$ dot products to get this matrix ready.

In 100-d, each dot product now needs 103 operations instead of 75 million

But there are still worrying things lurking away. What are they?

constraints:

$$Q_{ij} = y_i y_j (\mathbf{F}(\mathbf{x}_i) \cdot \mathbf{F}(\mathbf{x}_j))$$

The use of Maximum Margin **magically** makes this not a problem

$$\sum_k a_k y_k = 0$$

•The fear of overfitting with this enormous number of terms

•The evaluation phase (doing a set of predictions on a test set) will be very expensive (why?)

Then define:

$$\mathbf{w} = \sum_{k \text{ s.t. } a_k > 0} a_k y_k \mathbf{F}(\mathbf{x}_k)$$

Because each $\mathbf{w} \cdot \mathbf{f}(\mathbf{x})$ (see below) needs 75 million operations. What can be done?

$$\begin{aligned} \mathbf{w} \cdot \mathbf{F}(\mathbf{x}) &= \sum_{k \text{ s.t. } a_k > 0} a_k y_k \mathbf{F}(\mathbf{x}_k) \cdot \mathbf{F}(\mathbf{x}) \\ &= \sum_{k \text{ s.t. } a_k > 0} a_k y_k (\mathbf{x}_k \cdot \mathbf{x} + 1)^5 \end{aligned}$$

When you see this many callout bubbles on a slide it's time to wrap the author in a blanket, gently take him away and murmur "someone's been at the PowerPoint for too long."

Only S m operations (S =#support vectors)

Copyright © 2001, Andrew W. Moore

Support Vector Machines: Slide 58

QP with Quintic basis functions

$$\text{Maximize } \sum_{k=1}^R a_k + \sum_{k=1}^R \sum_{l=1}^R a_k a_l Q_{kl} \text{ where}$$

$$\text{Subject to these constraints: } 0 \leq a_k \leq C$$

Then define:

$$\mathbf{w} = \sum_{k \text{ s.t. } a_k > 0} a_k y_k \mathbf{F}(\mathbf{x}_k)$$

$$\begin{aligned} \mathbf{w} \cdot \mathbf{F}(\mathbf{x}) &= \sum_{k \text{ s.t. } a_k > 0} a_k y_k \mathbf{F}(\mathbf{x}_k) \cdot \mathbf{F}(\mathbf{x}) \\ &= \sum_{k \text{ s.t. } a_k > 0} a_k y_k (\mathbf{x}_k \cdot \mathbf{x} + 1)^5 \end{aligned}$$

Only S m operations ($S = \#$ support vectors)

Andrew's opinion of why SVMs don't overfit as much as you'd think:

No matter what the basis function, there are really only up to R parameters: $\mathbf{a}_1, \mathbf{a}_2 \dots \mathbf{a}_R$ and usually most are set to zero by the Maximum Margin.

Asking for small $\mathbf{w} \cdot \mathbf{w}$ is like "weight decay" in Neural Nets and like Ridge Regression parameters in Linear regression and like the use of Priors in Bayesian Regression---all designed to smooth the function and reduce overfitting.

Then classify with:

$$f(\mathbf{x}, \mathbf{w}, b) = \text{sign}(\mathbf{w} \cdot \mathbf{f}(\mathbf{x}) - b)$$

Copyright © 2001, Andrew W. Moore

Support Vector Machines: Slide 59

SVM Kernel Functions

- $K(\mathbf{a}, \mathbf{b}) = (\mathbf{a} \cdot \mathbf{b} + 1)^d$ is an example of an SVM Kernel Function
- Beyond polynomials there are other very high dimensional basis functions that can be made practical by finding the right Kernel Function
 - Radial-Basis-style Kernel Function:

$$K(\mathbf{a}, \mathbf{b}) = \exp\left(-\frac{(\mathbf{a} - \mathbf{b})^2}{2s^2}\right)$$

- Neural-net-style Kernel Function:

$$K(\mathbf{a}, \mathbf{b}) = \tanh(\mathbf{k} \cdot \mathbf{a} \cdot \mathbf{b} - d)$$

σ, κ and δ are magic parameters that must be chosen by a model selection method such as CV or VCSRM*

*see last lecture

Copyright © 2001, Andrew W. Moore

Support Vector Machines: Slide 60

VC-dimension of an SVM

- Very very very loosely speaking there is some theory which under some different assumptions puts an upper bound on the VC dimension as

$$\left[\frac{\text{Diameter}}{\text{Margin}} \right]$$

- where
 - *Diameter* is the diameter of the smallest sphere that can enclose all the high-dimensional term-vectors derived from the training set.
 - *Margin* is the smallest margin we'll let the SVM use
- This can be used in SRM (Structural Risk Minimization) for choosing the polynomial degree, RBF σ , etc.
 - But most people just use Cross-Validation

Copyright © 2001, Andrew W. Moore

Support Vector Machines: Slide 61

SVM Performance

- Anecdotally they work very very well indeed.
- Example: They are currently the best-known classifier on a well-studied hand-written-character recognition benchmark
- Another Example: Andrew knows several reliable people doing practical real-world work who claim that SVMs have saved them when their other favorite classifiers did poorly.
- There is a lot of excitement and religious fervor about SVMs as of 2001.
- Despite this, some practitioners (including your lecturer) are a little skeptical.

Copyright © 2001, Andrew W. Moore

Support Vector Machines: Slide 62

Doing multi-class classification

- SVMs can only handle two-class outputs (i.e. a categorical output variable with arity 2).
- What can be done?
- Answer: with output arity N, learn N SVM's
 - SVM 1 learns "Output==1" vs "Output != 1"
 - SVM 2 learns "Output==2" vs "Output != 2"
 - :
 - SVM N learns "Output==N" vs "Output != N"
- Then to predict the output for a new input, just predict with each SVM and find out which one puts the prediction the furthest into the positive region.

References

- An excellent tutorial on VC-dimension and Support Vector Machines:
 - C.J.C. Burges. A tutorial on support vector machines for pattern recognition. *Data Mining and Knowledge Discovery*, 2(2):955-974, 1998.
 - <http://citeseer.nj.nec.com/burges98tutorial.html>
- The VC/SRM/SVM Bible:
 - Statistical Learning Theory by Vladimir Vapnik, Wiley-Interscience; 1998

What You Should Know

- Linear SVMs
- The definition of a maximum margin classifier
- What QP can do for you (but, for this class, you don't need to know how it does it)
- How Maximum Margin can be turned into a QP problem
- How we deal with noisy (non-separable) data
- How we permit non-linear boundaries
- How SVM Kernel functions permit us to pretend we're working with ultra-high-dimensional basis-function terms