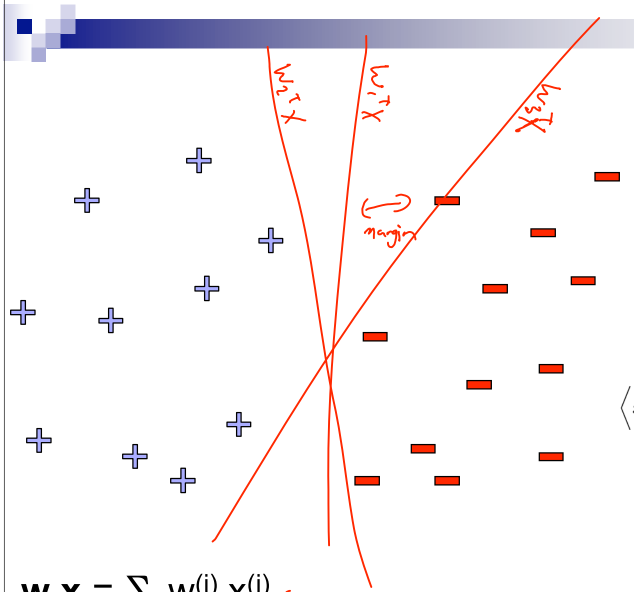
Support Vector Machines

Machine Learning – 10701/15781

Carlos Guestrin
Carnegie Mellon University

October 17th, 2007

Linear classifiers – Which line is better?



Data:

$$\left\langle x_1^{(1)}, \ldots, x_1^{(m)}, y_1 \right\rangle$$
$$\vdots$$
$$\left\langle x_n^{(1)}, \ldots, x_n^{(m)}, y_n \right\rangle$$

Example i:

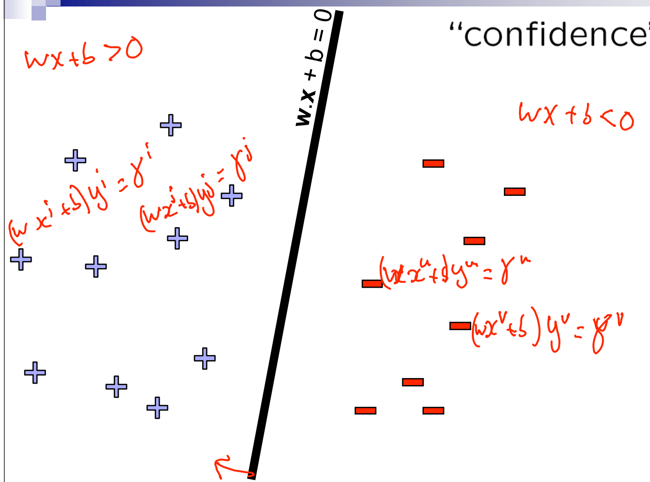
$\left\langle x_i^{(1)}, \ldots, x_i^{(m)} \right\rangle$ — m features

$y_i \in \{-1, +1\}$ — class

w.x = $\sum_j w^{(j)} x^{(j)}$

Pick the one with the largest margin!

"confidence" $= \left(\mathbf{w}.\mathbf{x}_j + b\right) y_j$



max min γ^i (over w,b; i={1..n})

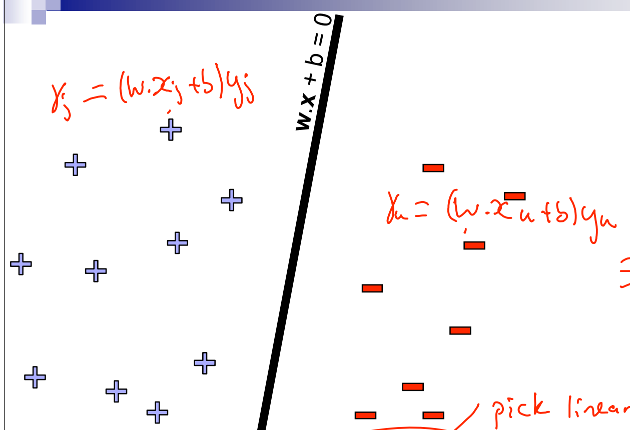
⇓

pick plane where worst point has highest confidence

$\mathbf{w}.\mathbf{x} = \sum_j w^{(j)} x^{(j)}$

Maximize the margin

$\gamma_1 = 0.1 \geq \gamma$
$\gamma_2 = 0.2 \geq \gamma$
$\gamma_3 = 0.05 \geq \gamma$



$\max_{w,b} \min_i (w.x_i + b) y_i$

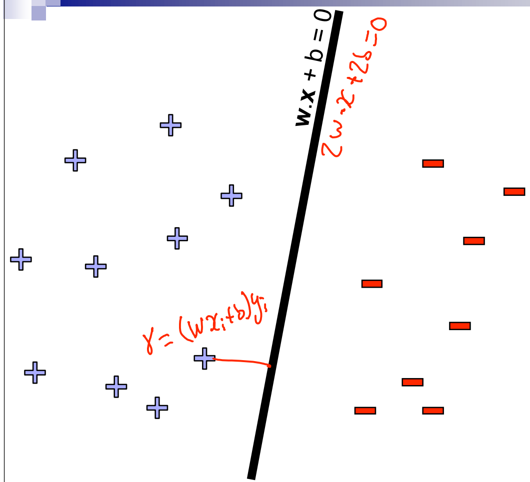
$\equiv \max_{w,b} \min_i \gamma_i$, $(w.x_i + b) y_i \geq \gamma_i$

$\equiv \max_{w,b} \gamma$, $(w.x_i + b) y_i \geq \gamma \; \forall i$

pick linear classifier

$$\text{maximize}_{\gamma,\mathbf{w},b} \quad \gamma$$
$$\left(\mathbf{w}.\mathbf{x}_j + b\right) y_j \geq \gamma, \;\; \forall j \in \text{Dataset}$$

But there are a many planes…



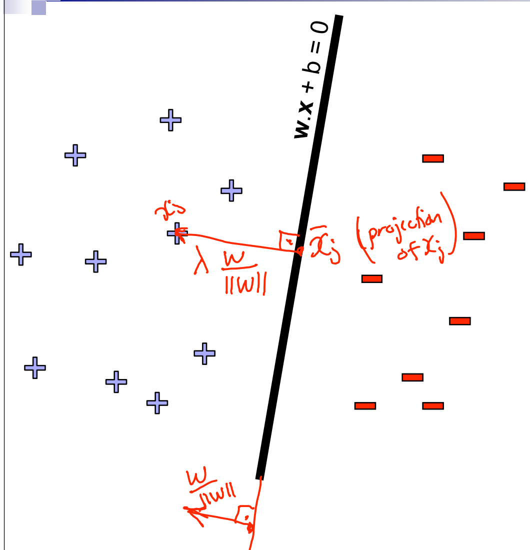
what if: new plane $\bar{w} = 2w$, $\bar{b} = 2b$

↳ now $\bar{\gamma} = 2\gamma$

I can be as confident as I want by increasing w

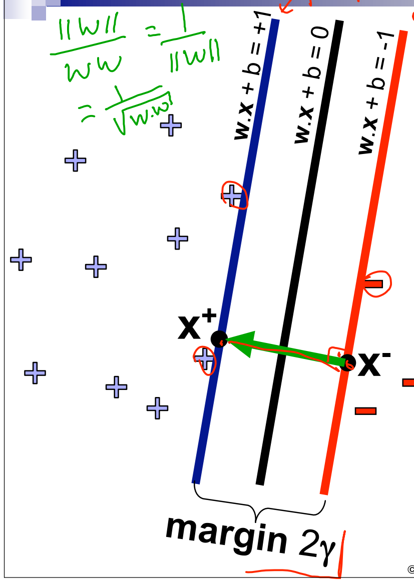
Review: Normal to a plane

$$\mathbf{x}_j = \bar{\mathbf{x}}_j + \lambda_j \frac{\mathbf{w}}{||\mathbf{w}||}$$



Normalized margin – Canonical hyperplanes

$$\mathbf{x}_j = \bar{\mathbf{x}}_j + \lambda \frac{\mathbf{w}}{||\mathbf{w}||}$$



$x^+ = x^- + \lambda \frac{w}{||w||}$

$x^+ w + b = 1$

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

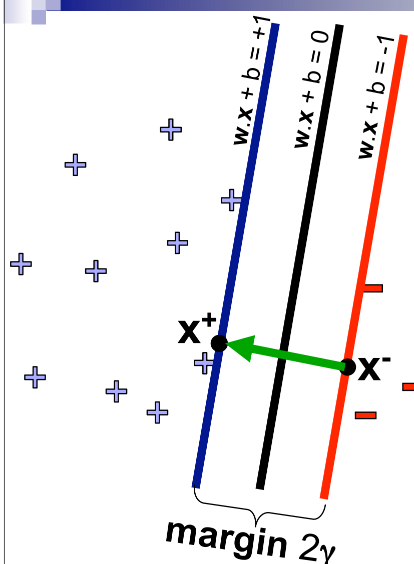
$x^- w + b + \lambda \frac{ww}{||w||} = 1$

$\lambda = 2\gamma$

$\lambda = 2\frac{||w||}{ww} \Rightarrow \gamma = \frac{1}{\sqrt{ww}}$

$\frac{||w||}{ww} = \frac{1}{||w||} = \frac{1}{\sqrt{w.w}}$

Normalized margin – Canonical hyperplanes



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

$$\mathbf{w}.\mathbf{x}^+ + b = 1$$

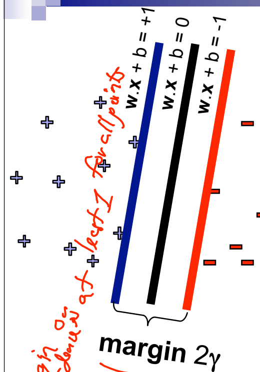
$$\mathbf{w}.(\mathbf{x}^- + \lambda \frac{\mathbf{w}}{||\mathbf{w}||}) + b = 1$$

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

$$\gamma = \frac{1}{\sqrt{\mathbf{w}.\mathbf{w}}}$$

Margin maximization using canonical hyperplanes

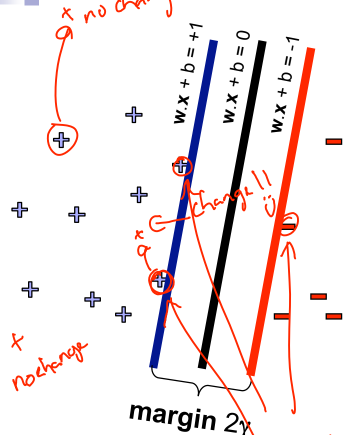
$$\gamma = \frac{1}{\sqrt{\mathbf{w}.\mathbf{w}}}$$



$$\text{maximize}_{\gamma,\mathbf{w},b} \quad \gamma$$
$$\left(\mathbf{w}.\mathbf{x}_j + b\right) y_j \geq \gamma, \ \forall j \in \text{Dataset}$$

$$\text{minimize}_{\mathbf{w},b} \quad \mathbf{w}.\mathbf{w}$$
$$\left(\mathbf{w}.\mathbf{x}_j + b\right) y_j \geq 1, \ \forall j \in \text{Dataset}$$

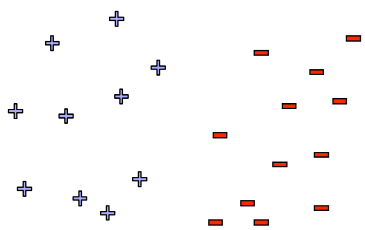
Support vector machines (SVMs)



$$\text{minimize}_{\mathbf{w},b} \quad \mathbf{w}.\mathbf{w}$$
$$\left(\mathbf{w}.\mathbf{x}_j + b\right) y_j \geq 1, \ \forall j$$

- Solve efficiently by quadratic programming (QP)
  - Well-studied solution algorithms
- Hyperplane defined by support vectors

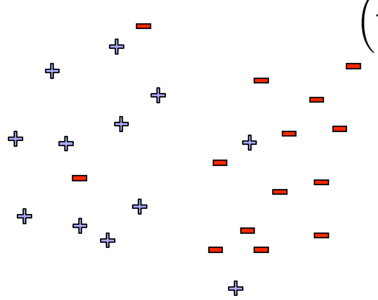
What if the data is not linearly separable?



Use features of features of features of features….

 11

What if the data is still not linearly separable?



$$\text{minimize}_{\mathbf{w},b} \quad \mathbf{w}.\mathbf{w}$$

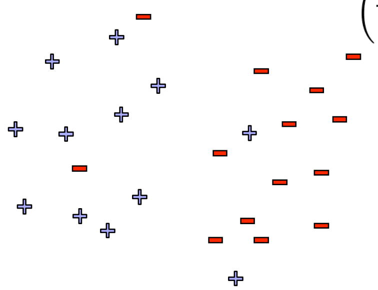
$$\left(\mathbf{w}.\mathbf{x}_j + b\right) y_j \geq 1 \qquad , \forall j$$

- Minimize **w.w** and number of training mistakes
  - Tradeoff two criteria?
- Tradeoff #(mistakes) and **w.w**
  - 0/1 loss
  - Slack penalty *C*
  - Not QP anymore
  - Also doesn't distinguish near misses and really bad mistakes

 12

Slack variables – Hinge loss

$$\text{minimize}_{\mathbf{w},b} \quad \mathbf{w}.\mathbf{w}$$
$$\left(\mathbf{w}.\mathbf{x}_j + b\right) y_j \geq 1 \qquad , \forall j$$



- If margin $\geq$ 1, don't care
- If margin < 1, pay linear penalty

Side note: What's the difference between SVMs and logistic regression?

SVM:

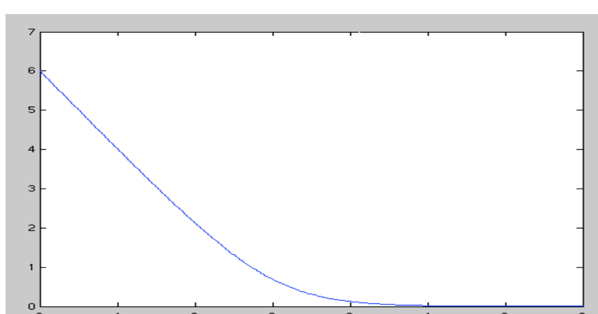
$$\text{minimize}_{\mathbf{w},b} \quad \mathbf{w}.\mathbf{w} + C \sum_j \xi_j$$
$$\left(\mathbf{w}.\mathbf{x}_j + b\right) y_j \geq 1 - \xi_j, \ \forall j$$
$$\xi_j \geq 0, \ \forall j$$

Logistic regression:

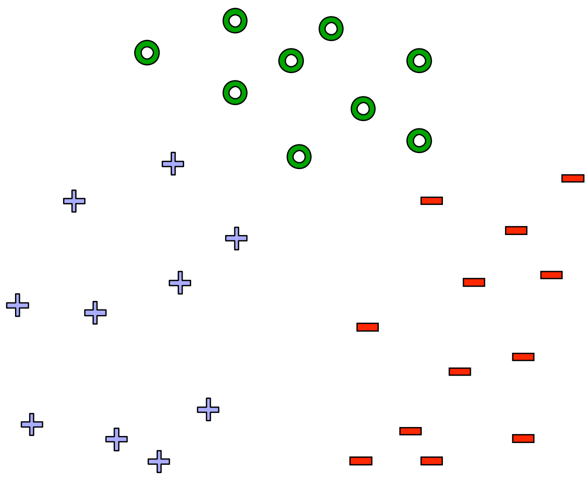
$$P(Y = 1 \mid x, \mathbf{w}) = \frac{1}{1 + e^{-(\mathbf{w}.\mathbf{x}+b)}}$$

Log loss:

$$-\ln P(Y = 1 \mid x, \mathbf{w}) = \ln\left(1 + e^{-(\mathbf{w}.\mathbf{x}+b)}\right)$$

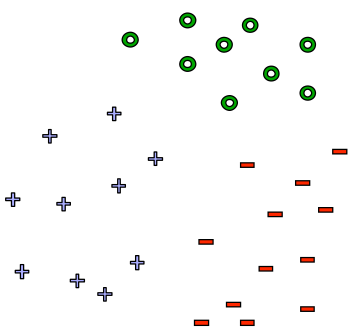


What about multiple classes?



©2005-2007 Carlos Guestrin 15

One against All

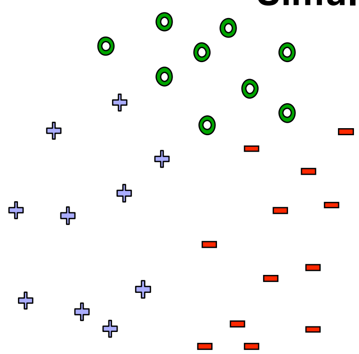


Learn 3 classifiers:

©2005-2007 Carlos Guestrin 16

Learn 1 classifier: Multiclass SVM

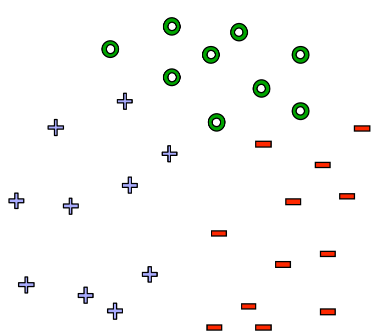
Simultaneously learn 3 sets of weights



$$\mathbf{w}^{(y_j)}.\mathbf{x}_j + b^{(y_j)} \geq \mathbf{w}^{(y')}.\mathbf{x}_j + b^{(y')} + 1, \ \forall y' \neq y_j, \ \forall j$$

Learn 1 classifier: Multiclass SVM

$$\text{minimize}_{\mathbf{w},b} \quad \sum_y \mathbf{w}^{(y)}.\mathbf{w}^{(y)} + C \sum_j \xi_j$$
$$\mathbf{w}^{(y_j)}.\mathbf{x}_j + b^{(y_j)} \geq \mathbf{w}^{(y')}.\mathbf{x}_j + b^{(y')} + 1 - \xi_j, \ \forall y' \neq y_j, \ \forall j$$
$$\xi_j \geq 0, \ \forall j$$



What you need to know

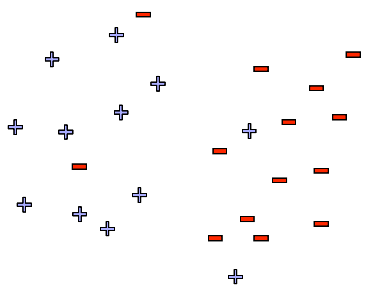
- Maximizing margin
- Derivation of SVM formulation
- Slack variables and hinge loss
- Relationship between SVMs and logistic regression
  - 0/1 loss
  - Hinge loss
  - Log loss
- Tackling multiple class
  - One against All
  - Multiclass SVMs

SVMs, Duality and the Kernel Trick

Machine Learning – 10701/15781
Carlos Guestrin
Carnegie Mellon University

October 17th, 2007

SVMs reminder



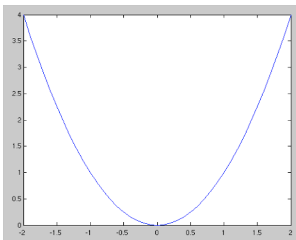
$$\text{minimize}_{\mathbf{w},b} \quad \mathbf{w}.\mathbf{w} + C\sum_j \xi_j$$
$$\left(\mathbf{w}.\mathbf{x}_j + b\right) y_j \geq 1 - \xi_j, \ \forall j$$
$$\xi_j \geq 0, \ \forall j$$

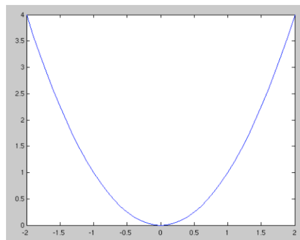
Today's lecture

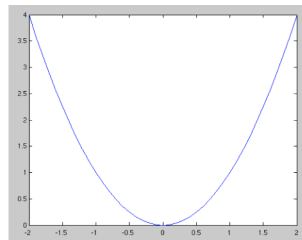
- Learn one of the most interesting and exciting recent advancements in machine learning
  - The "kernel trick"
  - High dimensional feature spaces at no extra cost!
- But first, a detour
  - Constrained optimization!

Constrained optimization

$$\min_x \; x^2$$
$$\text{s.t.} \quad x \geq b$$

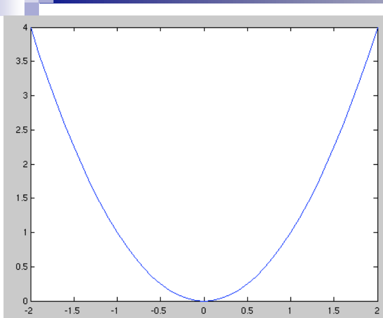






Lagrange multipliers – Dual variables

$$\min_x \; x^2$$
$$\text{s.t.} \quad x \geq b$$



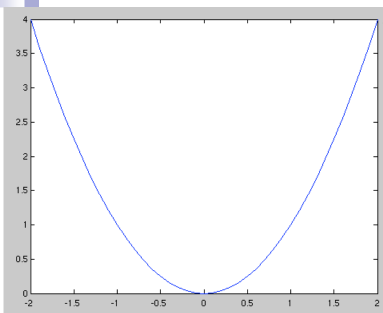
Moving the constraint to objective function
Lagrangian:

$$L(x, \alpha) = x^2 - \alpha(x - b)$$
$$\text{s.t.} \quad \alpha \geq 0$$

Solve:

$$\min_x \max_\alpha \; L(x, \alpha)$$
$$\text{s.t.} \quad \alpha \geq 0$$

Lagrange multipliers – Dual variables



Solving: $\min_x \max_\alpha \; x^2 - \alpha(x - b)$

s.t. $\alpha \geq 0$

Dual SVM derivation (1) – the linearly separable case

$$\text{minimize}_{\mathbf{w},b} \quad \frac{1}{2}\mathbf{w}.\mathbf{w}$$
$$\left(\mathbf{w}.\mathbf{x}_j + b\right) y_j \geq 1, \;\; \forall j$$

Dual SVM derivation (2) – the linearly separable case

$L(\mathbf{w},\alpha) = \frac{1}{2}\mathbf{w}.\mathbf{w} - \sum_j \alpha_j \left[\left(\mathbf{w}.\mathbf{x}_j + b\right) y_j - 1\right]$
$\alpha_j \geq 0, \ \forall j$

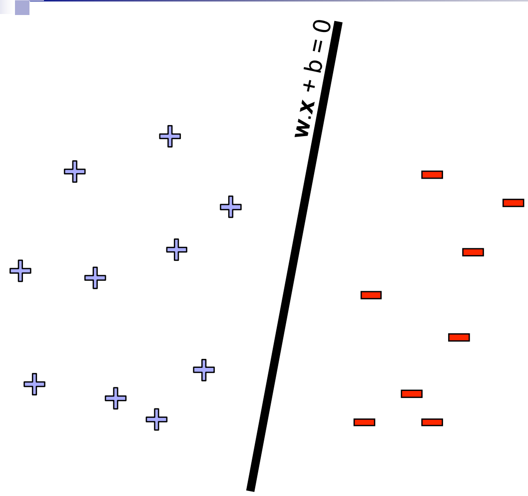
$$\mathbf{w} = \sum_j \alpha_j y_j \mathbf{x}_j$$

$\text{minimize}_{\mathbf{w}} \quad \frac{1}{2}\mathbf{w}.\mathbf{w}$
$\left(\mathbf{w}.\mathbf{x}_j + b\right) y_j \geq 1, \ \forall j$

$$b = y_k - \mathbf{w}.\mathbf{x}_k$$

for any k where $\alpha_k > 0$

Dual SVM interpretation



$$\mathbf{w} = \sum_j \alpha_j y_j \mathbf{x}_j$$

Dual SVM formulation – the linearly separable case

$$\text{maximize}_\alpha \quad \sum_i \alpha_i - \frac{1}{2}\sum_{i,j}\alpha_i\alpha_j y_i y_j \mathbf{x}_i\mathbf{x}_j$$

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

$$\mathbf{w} = \sum_i \alpha_i y_i \mathbf{x}_i$$

$$b = y_k - \mathbf{w}.\mathbf{x}_k$$

for any k where $\alpha_k > 0$

Dual SVM derivation – the non-separable case

$$\text{minimize}_{\mathbf{w},b} \quad \frac{1}{2}\mathbf{w}.\mathbf{w} + C\sum_j \xi_j$$
$$\left(\mathbf{w}.\mathbf{x}_j + b\right) y_j \geq 1 - \xi_j, \ \forall j$$
$$\xi_j \geq 0, \ \forall j$$

Dual SVM formulation – the non-separable case

$$\text{maximize}_\alpha \quad \sum_i \alpha_i - \frac{1}{2}\sum_{i,j} \alpha_i \alpha_j y_i y_j \mathbf{x}_i \mathbf{x}_j$$

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

$$\mathbf{w} = \sum_i \alpha_i y_i \mathbf{x}_i$$

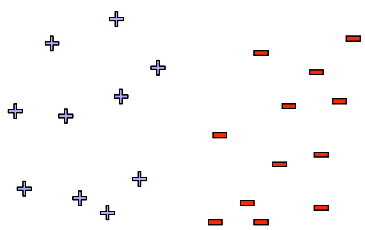
$$b = y_k - \mathbf{w}.\mathbf{x}_k$$

for any k where $C > \alpha_k > 0$

Why did we learn about the dual SVM?

- There are some quadratic programming algorithms that can solve the dual faster than the primal
- But, more importantly, the "**kernel trick**"!!!
  - Another little detour…

Reminder from last time: What if the data is not linearly separable?



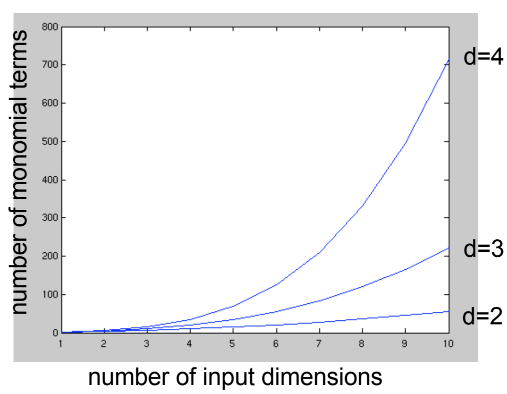
Use features of features of features of features….

$$\Phi(\mathbf{x}) : R^m \mapsto F$$

Feature space can get really large really quickly!

Higher order polynomials

$$\text{num. terms} = \binom{d+m-1}{d} = \frac{(d+m-1)!}{d!(m-1)!}$$



m – input features
d – degree of polynomial

grows fast!
d = 6, m = 100
about 1.6 billion terms

Dual formulation only depends on dot-products, not on **w**!

$$\text{maximize}_\alpha \quad \sum_i \alpha_i - \frac{1}{2}\sum_{i,j}\alpha_i\alpha_j y_i y_j \mathbf{x}_i\mathbf{x}_j$$

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

$$\text{maximize}_\alpha \quad \sum_i \alpha_i - \frac{1}{2}\sum_{i,j}\alpha_i\alpha_j y_i y_j K(\mathbf{x}_i, \mathbf{x}_j)$$

$$K(\mathbf{x}_i, \mathbf{x}_j) = \Phi(\mathbf{x}_i)\cdot\Phi(\mathbf{x}_j)$$
$$\sum_i \alpha_i y_i = 0$$
$$C \geq \alpha_i \geq 0$$

Dot-product of polynomials

$\Phi(\mathbf{u})\cdot\Phi(\mathbf{v}) =$ polynomials of degree d

Finally: the "kernel trick"!

maximize$_\alpha$ $\sum_i \alpha_i - \frac{1}{2}\sum_{i,j} \alpha_i\alpha_j y_i y_j K(\mathbf{x}_i, \mathbf{x}_j)$

$K(\mathbf{x}_i, \mathbf{x}_j) = \Phi(\mathbf{x}_i) \cdot \Phi(\mathbf{x}_j)$

$\sum_i \alpha_i y_i = 0$

$C \geq \alpha_i \geq 0$

$$\mathbf{w} = \sum_i \alpha_i y_i \Phi(\mathbf{x}_i)$$

$$b = y_k - \mathbf{w}.\Phi(\mathbf{x}_k)$$

for any k where $C > \alpha_k > 0$

- Never represent features explicitly
  - Compute dot products in closed form
- Constant-time high-dimensional dot-products for many classes of features
- Very interesting theory – Reproducing Kernel Hilbert Spaces
  - Not covered in detail in 10701/15781, more in 10702

Polynomial kernels

- All monomials of degree d in O(d) operations:

$\Phi(\mathbf{u}) \cdot \Phi(\mathbf{v}) = (\mathbf{u} \cdot \mathbf{v})^d =$ polynomials of degree d

- How about all monomials of degree up to d?
  - Solution 0:
  - Better solution:

Common kernels

- Polynomials of degree d

$$K(\mathbf{u},\mathbf{v}) = (\mathbf{u}\cdot\mathbf{v})^d$$

- Polynomials of degree up to d

- Gaus $K(\mathbf{u},\mathbf{v}) = (\mathbf{u}\cdot\mathbf{v}+1)^d$

- Sigmoid $K(\mathbf{u},\mathbf{v}) = \exp\left(-\frac{||\mathbf{u}-\mathbf{v}||}{2\sigma^2}\right)$

$$K(\mathbf{u},\mathbf{v}) = \tanh(\eta\mathbf{u}\cdot\mathbf{v}+\nu)$$

Overfitting?

- Huge feature space with kernels, what about overfitting???
  - Maximizing margin leads to sparse set of support vectors
  - Some interesting theory says that SVMs search for simple hypothesis with large margin
  - Often robust to overfitting

What about at classification time

- For a new input $\mathbf{x}$, if we need to represent $\Phi(\mathbf{x})$, we are in trouble!
- Recall classifier: sign($\mathbf{w}.\Phi(\mathbf{x})$+b)
- Using kernels we are cool!

$$K(\mathbf{u},\mathbf{v}) = \Phi(\mathbf{u}) \cdot \Phi(\mathbf{v})$$

$$\mathbf{w} = \sum_i \alpha_i y_i \Phi(\mathbf{x}_i)$$

$$b = y_k - \mathbf{w}.\Phi(\mathbf{x}_k)$$

for any k where $C > \alpha_k > 0$

SVMs with kernels

- Choose a set of features and kernel function
- Solve dual problem to obtain support vectors α_i
- At classification time, compute:

$$\mathbf{w} \cdot \Phi(\mathbf{x}) = \sum_i \alpha_i y_i K(\mathbf{x},\mathbf{x}_i)$$

$$b = y_k - \sum_i \alpha_i y_i K(\mathbf{x}_k,\mathbf{x}_i)$$

for any k where $C > \alpha_k > 0$

Classify as

$$sign\left(\mathbf{w} \cdot \Phi(\mathbf{x}) + b\right)$$

What's the difference between SVMs and Logistic Regression?

	SVMs	Logistic Regression
Loss function		
High dimensional features with kernels		

Kernels in logistic regression

$$P(Y = 1 \mid x, \mathbf{w}) = \frac{1}{1 + e^{-(\mathbf{w}\cdot\Phi(\mathbf{x})+b)}}$$

- Define weights in terms of support vectors:

$$\mathbf{w} = \sum_i \alpha_i \Phi(\mathbf{x}_i)$$

$$P(Y = 1 \mid x, \mathbf{w}) = \frac{1}{1 + e^{-(\sum_i \alpha_i \Phi(\mathbf{x}_i)\cdot\Phi(\mathbf{x})+b)}}$$
$$= \frac{1}{1 + e^{-(\sum_i \alpha_i K(\mathbf{x},\mathbf{x}_i)+b)}}$$

- Derive simple gradient descent rule on α_i

What's the difference between SVMs and Logistic Regression? (Revisited)

	SVMs	Logistic Regression
Loss function	Hinge loss	Log-loss
High dimensional features with kernels	Yes!	Yes!

What you need to know

- Dual SVM formulation
  - How it's derived
- The kernel trick
- Derive polynomial kernel
- Common kernels
- Kernelized logistic regression
- Differences between SVMs and logistic regression