Cmput 466/551

Support Vector Machines: Max Margin, …

Covering chapters (HTF): 4.5, 12

R Greiner
Department of Computing Science
University of Alberta

Much of this is taken from Andrew W. Moore; CMU
http://www.cs.cmu.edu/~awm/tutorials

+ Nello Cristianini, Ron Meir, Ron Parr, Barnabas Poczos

Error Function

Jump

Given data $\{ [x^{(i)}, y^{(i)}] \}_{i=1..m}$, optimize...

- 1. Classification error
  Perceptron Training

$$err_{Class}(w) = \frac{1}{m}\sum_{i=1}^{m} I[\, y^{(i)} \neq sign(w^T x^{(i)})\,]$$

- 2. Mean-squared error (LMS)
  Direct (Gradient Descent)

$$err_{MSE}(w) = \frac{1}{m}\sum_{i=1}^{m} \frac{1}{2}[\, y^{(i)} - w^T\, x^{(i)}\,]^2$$

- 3. (Log) Conditional Probability (LR)
  MSE Gradient Descent; LCL Gradient Descent

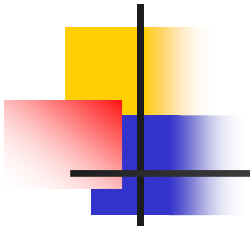
$$LCL(w) = \frac{1}{m}\sum_{i=1}^{m} \log P_w(\, y^{(i)} \mid x^{(i)})$$

- 4. (Log) Joint Probability (LDA; FDA)
  Direct Computation

$$LL(w) = \frac{1}{m}\sum_{i=1}^{m} \log P_w(\, y^{(i)}, x^{(i)})$$

- 5. Hinge Loss (SVM)

$$HL(w) = \frac{1}{m}\sum_{i} \lambda_i \left(y^{(i)} \boldsymbol{w}^T \boldsymbol{x}^{(i)} - 1\right)_+ +$$



A Little History

- Support Vector Machines (SVM)
  - introduced in COLT-92
  - greatly developed since then
- Now, a large and diverse community:
  - machine learning
  - optimization
  - statistics
  - neural networks
  - functional analysis, etc etc etc.
- Successful applications in many fields (bioinformatics, text, handwriting recognition, etc)
- Kernel Machines: large class of learning algs
  - SVM = a particular instance
- http://www.kernel-machines.org

Skip in class

Log in

- Home
- Frequently Asked Questions
- Publications
- Books
- Software
- Annual Workshop
- JMLR
- Links

News

- Machine Learning Summer School / Course On The Analysis On Patterns 2007-02-12
- New Kernel-Machines.org server 2007-01-30
- Call for participation: The 2006 kernel workshop, "10 years of kernel machines" 2006-10-06
- Three Workshops on Kernel Methods at NIPS 2005 2005-09-08
- Nips 2004 Workshop on Learning With Structured Outputs 2004-10-22

More news...

Books

Books on SVMs and Other Kernel Machines

last modified 2007-01-31 12:45

- Vladimir Vapnik. Estimation of Dependences Based on Empirical Data. Springer Verlag, 2006, 2nd edition.

  The second edition of Vapnik's classic on learning theory, including several new chapters on the history of events and on non-inductive inference.

- Grace Wahba. **Spline Models for Observational Data**. SIAM CBMS-NSF Regional Conference Series in Applied Mathematics vol. 59, Philadelphia, 1990.

  Discusses (reproducing) kernel methods in nonparametric regression. Not easy reading for machine learning researchers, but containing fundamental material about precedents of today's kernel machines (169 pages, $33.5).

- Vladimir Vapnik. **The Nature of Statistical Learning Theory**. Springer, NY, 1995.

  An overview of statistical learning theory, containing no proofs, but most of the crucial theorems and milestones of learning theory. With a detailed chapter on SVMs for pattern recognition and regression (1st edition: 188 pages, $65; 2nd edition: 304 pages, $70).

- Vladimir Vapnik. **Statistical Learning Theory**. Wiley, NY, 1998.

  The comprehensive treatment of statistical learning theory, including a large amount of material on SVMs (768 pages, $120).

- Bernhard Schölkopf, Chris Burges, and Alex Smola (eds). Advances in Kernel Methods - Support Vector Learning MIT Press, Cambridge, MA, 1999.

  A collection of articles written by experts in the field. Includes an introductory tutorial, overviews of the theory of SVMs, contributions on novel algorithms, and three chapters on SVM implementations (392 pages, $53).

- Nello Cristianini and John Shawe-Taylor. An Introduction to Support Vector Machines. Cambridge University Pres Cambridge, UK, 2000.

  An introduction to SVMs which is concise yet comprehensive in its description of the theoretical foundations of large margin algorithms (189 pages, $45).

- Alex Smola, Peter Bartlett, Bernhard Schölkopf, and Dale Schuurmans (eds). Advances in Large Margin Classifier MIT Press, Cambridge, MA, 2000.

  A collection of articles dealing with one of the main ideas of SVMs, large margin regularization. Contains an introduction, articles on new kernels, SVMs, and boosting algorithms (422 pages, $45).

Log in

Home
Frequently Asked Questions
Publications
Books
Software
Annual Workshop
JMLR
Links

News

› Machine Learning Summer School / Course On The Analysis On Patterns
2007-02-12

› New Kernel-Machines.org server
2007-01-30

› Call for participation: The 2006 kernel workshop, "10 years of kernel machines"
2006-10-06

› Three Workshops on Kernel Methods at NIPS 2005
2005-09-08

› Nips 2004 Workshop on Learning With Structured Outputs
2004-10-22

More news...

Software

Skip in class

Kernel-Machines.Org software links

last modified
2007-01-31 12:47

Gaussian Processes

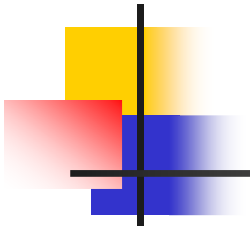
- GP Demo. Demonstration Software for Gaussian Processes by David MacKay (in OCTAVE).
- gpml. Matlab implementations of algorithms from Rasmussen & Williams "Gaussian Processes for Machine Learning the MIT Press 2006.
- LS-SVMlab. Matlab/C toolbox for least squares support vector machines.
- MAP-1. Package for MAP estimation by Carl Rasmussen.
- MC-1. Package for MAP estimation by Carl Rasmussen.
- Flexible Bayesian Modelling. Package by Radford Neal. It includes programs for Neural Networks, Gaussian Processes, and Mixture Models.
- Netlab. Matlab toolbox including Gaussian Process Regression, Mixture models and Neural Networks.
- Sparse Gaussian Processes. Matlab Toolbox for Sparse Inference using Gaussian Processes.
- Tpros and Cpros. Package by Mark Gibbs.

Mathematical Programming

- CPLEX. Barrier/QP Solver.
- LOQO. Linear and Quadratic Optimization Package by Robert Vanderbei.
- MINOS. Linear and Quadratic Solver.

Support Vectors

- Nearest Point Algorithm. by Sathiya Keerthi (in FORTRAN).
- SVM Java Applet. by Chris Burges et al.
- BSVM. A decomposition method for bound-constrained SVM formulations.
- QP SVM Classification and Regression. Fortran Implementation.
- CLISP/LibSVM. A module for using LibSVM from GNU CLISP (an ANSI Common Lisp implementation).
- Chunking Code. by C. Saunders, M. O. Stitson, J. Weston, L. Bottou, B. Schölkopf, and A. Smola at Royal Holloway, AT&T, and GMD FIRST (Documentation).
- cSVM. SVM for classification tasks with model selection.
- 2D SVM Interactive Demo. runs under Matlab 6 and produces nice pictures - useful for courses.
- DTREG. by Phillip H. Sherrod.
- Interior Point Optimizer for SVM Pattern Recognition. by Alex Smola.
- Equbits Foresight. Commerical SVM based Classification and Regression Application Designed for Drug Discovery.
- Gini-SVM. A multi-class Probabilistic regression software for large data sets.
- GiniSVM. Multi-class SVM Probability regression package.
- Gist. Gist contains software tools for support vector machine classification and for kernel principal components analysis. The SVM portion of Gist is available via an interactive web server.
- Parallel GPDT. Parallel and serial training of SVM.
- HeroSvm1.0. A high performance DLL for training SVM on a very large training set efficiently.
- SVM java implementation. This implementation is simple and easy to modify.
- LEARNSC. SVM, NN and FL MATLAB based user-friendly routines.
- LIBSVM. An SVM library with a graphic interface



Outline

- Foundations
  - ~~Linear Programming;Primal/Dual~~
  - Constrained Optimization
    - Lagrange Multipliers
    - KKT
  - Perceptron Factoids
    - Dual Representation
- "Best" Linear Separator: Max Margin!
- Coping with Non-Linearly Separated Data
  - Slack Variables
- Kernel Trick
- Regression

Skip in class

Background: Linear Programming

- **Linear Programming**
  - Given $\mathbf{c} \in \Re^n$, $\mathbf{A} \in \Re^{n \times m}$, $\mathbf{b} \in \Re^m$
  - find $\mathbf{x}^* = \text{argmax}_{\mathbf{x}} \mathbf{c}^T \mathbf{x}$
  - subject to
    - $\mathbf{A}\,\mathbf{x} \leq \mathbf{b}$ -- ie, $\mathbf{a}_i^T \mathbf{x} \leq b_i$ for i = 1 .. m
    - $\mathbf{x} \geq \mathbf{0}$ -- ie, $x_j \geq 0$ for j = 1 .. n

- ∃ fast algorithms for solving linear programs …including
  - simplex algorithm
  - Karmarkar's algorithm

Duality

Skip in class

- Given $\mathbf{c} \in \Re^m$, $\mathbf{A} \in \Re^{m \times n}$, $\mathbf{b} \in \Re^n$:

- **Primal**
  - find $\mathbf{x}^* = \text{argmax}_{\mathbf{x}}\ \mathbf{c}^T \mathbf{x}$
  - subject to
    - $\mathbf{A}\mathbf{x} \leq \mathbf{b}$
    - $\mathbf{x} \geq \mathbf{0}$

- **Equivalent Dual**
  - find $\mathbf{w}^* = \text{argmin}_{\mathbf{w}}\ \mathbf{b}^T \mathbf{w}$
  - subject to
    - $\mathbf{A}^T \mathbf{w} \geq \mathbf{c}$
    - $\mathbf{w} \geq \mathbf{0}$

Strong duality result:

If $\boldsymbol{x}^*$ is an optimal solution for the primal, then the dual has optimal solution $\boldsymbol{w}^*$ s.t:

$$\mathbf{c}^T\mathbf{x}^* = \mathbf{b}^T\mathbf{x}^*$$

If A is m×n
then, $|\mathbf{x}|=m$ $|\mathbf{w}|=n$ …

If m≫n, then $|\mathbf{x}| \gg |\mathbf{w}|$

Proof: Lagrange multipliers…

Skip in class

Primal ⇔ Dual

Primal Problem

- Maximize $7x_1 + 3x_2 + 2x_3$
- Subject to

$$\begin{aligned} 2x_1 + 5x_2 + 0x_3 &\geq 20 \\ 8x_1 + 3x_2 - 1x_3 &\geq -10 \end{aligned}$$

$x_1, x_2, x_3 \geq 0$

b

Dual Problem

- Minimize $20w_1 + -10w_2$
- Subject to

$$\begin{aligned} 2w_1 + 8w_2 &\leq 7 \\ 5w_1 + 3w_2 &\leq 3 \\ 0w_1 - 1w_2 &\leq 2 \end{aligned}$$

$w_1, w_2 \geq 0$

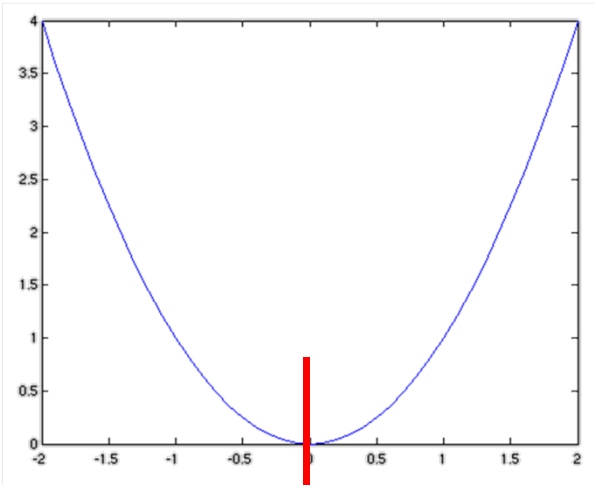
c

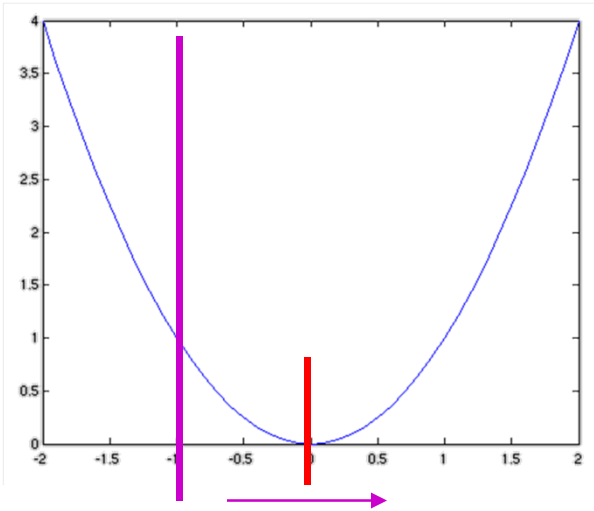
Constrained optimization

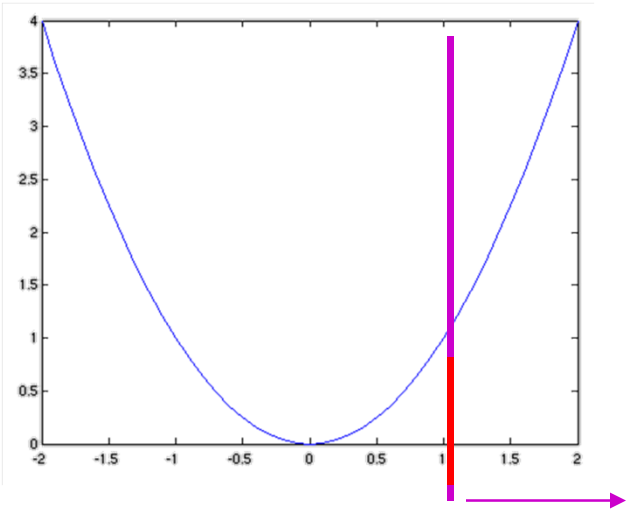
$\min_x x^2$

$\min_x x^2$
s.t. $x \geq -1$

$\min_x x^2$
s.t. $x \geq 1$







min @ x=0

Unconstrained

min @ x=0

Constraint **ir**relevant

min @ x=+1

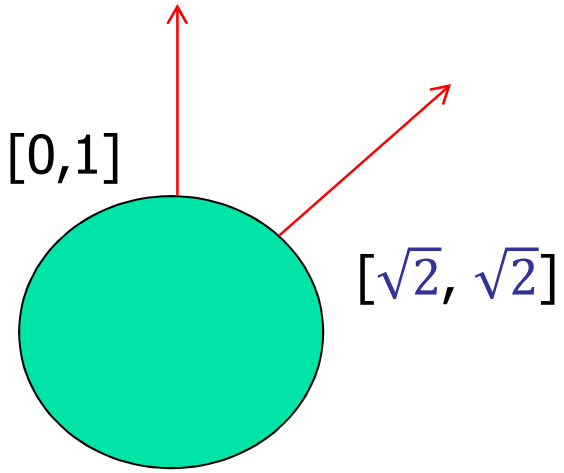
Constraint **relevant**

Gradient … partial derivative

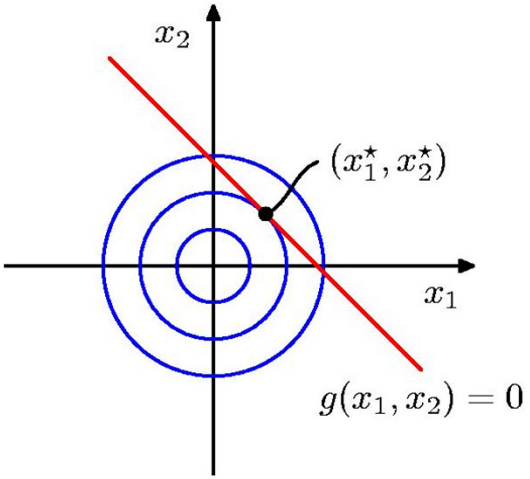
- $\nabla f(\mathbf{x}) = \left[\frac{\partial f(\mathbf{x})}{\partial x_1}, \ldots, \frac{\partial f(\mathbf{x})}{\partial x_n}\right]^T$
  - This vector is TANGENT to $f(\mathbf{x})$
- Eg:
  - $f(x, y) = x^2 + 3xy - 5x$
    - $\nabla f(x,y) = \left[\frac{\partial f(x,y)}{\partial x}, \frac{\partial f(x,y)}{\partial y}\right] = [2x + 3y - 5, \quad 3x]$
    - If evaluated at [x,y] = [1, 3],
      then this value is $\nabla f([1,3]) = [6, 3]$

Gradient … partial derivative

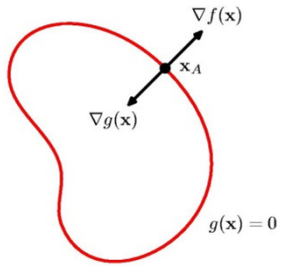
- $\nabla f(\mathbf{x}) = \left[\frac{\partial f(\mathbf{x})}{\partial x_1}, \ldots, \frac{\partial f(\mathbf{x})}{\partial x_n}\right]^T$
- This vector is DIRECTION where $f(\mathbf{x})$ increases fastest
- Eg: $f(x, y) = x^2 + y^2$
  - $\nabla f([x, y]) = [2x, 2y]$
  - … so $\nabla f([\sqrt{2}, \sqrt{2}]) = [2\sqrt{2}, 2\sqrt{2}]$
  - $\nabla f([0, 1]) = [0, 2]$



Lagrange Multiplier 101

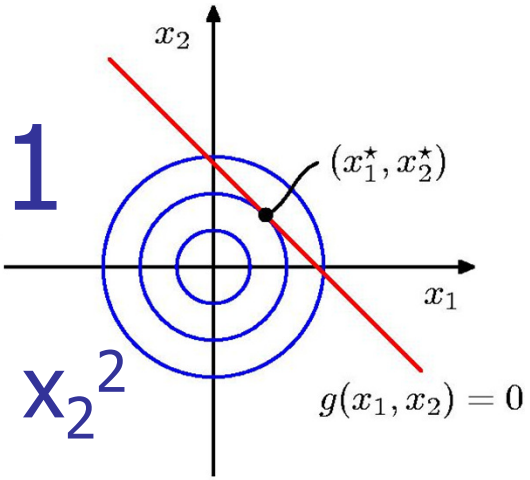


- Challenge: $\text{argmax}_{\mathbf{x}}\ f(\mathbf{x}) = -x_1^2 - x_2^2$
  s.t. $g(\mathbf{x}) = x_1 + x_2 - 1 = 0$
- Consider optimum $\mathbf{x}^* = (x^*_1, x^*_2)$
  - On $g(\mathbf{x}) = 0$ line, by def'n!
  - Note $\nabla f(\mathbf{x}^*) \perp g(\mathbf{x}^*)$
    - Otherwise, could walk along $g(\mathbf{x})=0$ to get larger $f(\mathbf{x})$ value
- In fact, $f(\mathbf{x}^*)$ is tangent to $g(\mathbf{x}^*)$
  $\Rightarrow \nabla f(\mathbf{x}^*)$ is || to $\nabla g(\mathbf{x}^*)$
  $\Rightarrow \exists\lambda$ s.t. $\nabla f(\mathbf{x}^*) + \lambda \nabla g(\mathbf{x}^*) = 0$



- Write $L(\mathbf{x}, \lambda) = f(\mathbf{x}) + \lambda g(\mathbf{x})$
- Note $\nabla_{\mathbf{x}} L(\mathbf{x}, \lambda) = \nabla_{\mathbf{x}} f(\mathbf{x}) + \lambda \nabla_{\mathbf{x}} g(\mathbf{x})$ ... = 0 @ $\mathbf{x}^*$
  $\nabla_{\lambda} L(\mathbf{x}, \lambda) = g(\mathbf{x})$ ... = 0 $\Rightarrow g(\mathbf{x}) = 0$ satisfied

Lagrange Multiplier 101

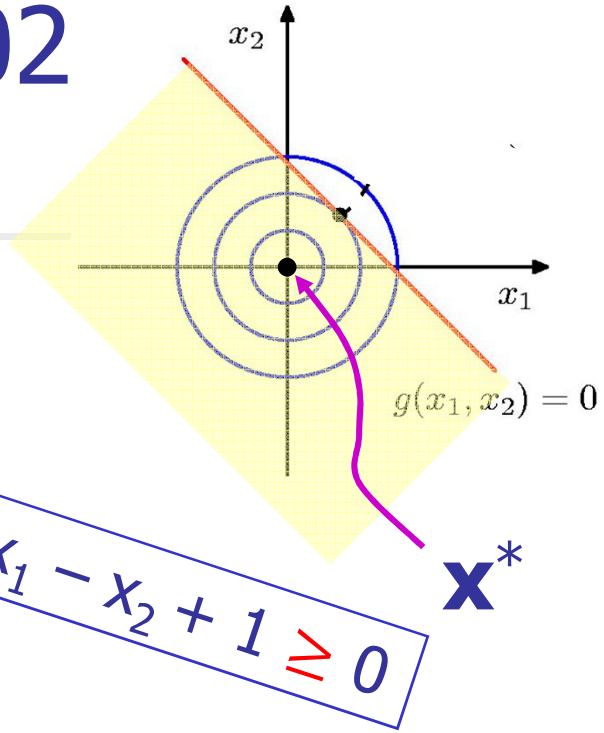


- Challenge: $\text{argmax}_{\mathbf{x}}\ f(\mathbf{x}) = -x_1^2 - x_2^2$
  s.t. $g(\mathbf{x}) = x_1 + x_2 - 1 = 0$
- $L(\mathbf{x}, \lambda) = f(\mathbf{x}) + \lambda g(\mathbf{x})$
- $\nabla_{x1} L(\mathbf{x}, \lambda) = -2x_1 + \lambda = 0$

  $\nabla_{x2} L(\mathbf{x}, \lambda) = -2x_2 + \lambda = 0$

  $\nabla_{\lambda} L(\mathbf{x}, \lambda) = x_1 + x_2 - 1 = 0$
  - Soln: $x_1 = x_2 = ½$
  - Also $\lambda = 1$ (who cares…)

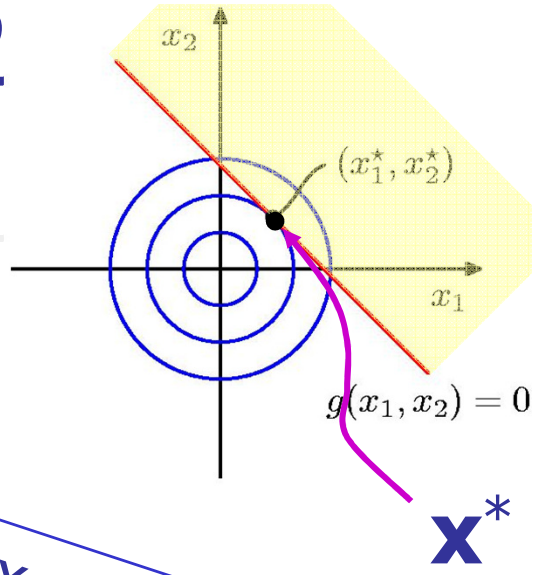
Lagrange Multiplier 102 InEqualities…



- $\text{argmax}_{\mathbf{x}}\ f(\mathbf{x})$ s.t. $g(\mathbf{x}) \geq 0$
- Two cases:
  - At optimal $\mathbf{x}^*$, $g(\mathbf{x}^*) > 0$
    - "inactive constraint"
    - Just need $\nabla f(\mathbf{x}) = 0$
    - … corresponds to $\nabla f(\mathbf{x}) + \lambda \nabla g(\mathbf{x}) = 0$ when $\lambda=0$

$g(x) = -x_1 - x_2 + 1 \geq 0$

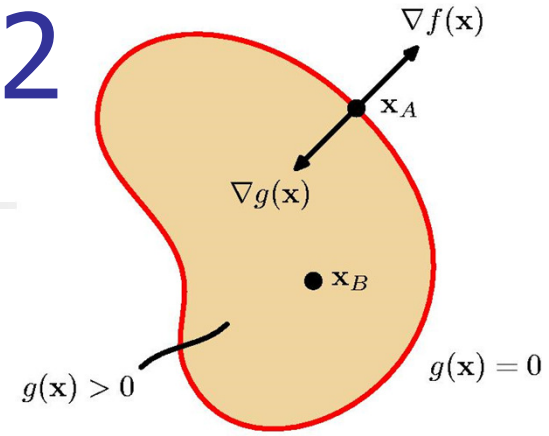
Lagrange Multiplier 102 InEqualities…



- $\text{argmax}_{\mathbf{x}}\ f(\mathbf{x})$ s.t. $g(\mathbf{x}) \geq 0$
- Two cases:
  - At optimal $\mathbf{x}^*$, $g(\mathbf{x}^*) > 0$
    - "inactive constraint"
    - Just need $\nabla f(\mathbf{x}) = 0$
    - … corresponds to $\nabla f(\mathbf{x}) + \lambda\, \nabla g(\mathbf{x}) = 0$ when $\lambda=0$
  - At optimal $\mathbf{x}^*$, $g(\mathbf{x}^*) = 0$
    - "active constraint"
    - need $\nabla f(\mathbf{x}) + \lambda\, \nabla g(\mathbf{x}) = 0$ with $\lambda \neq 0$
    - Actually… need $\lambda > 0$ as
      need $\nabla f(\mathbf{x})$ oriented AWAY from $g(x)>0$ region…
      $\nabla f(\mathbf{x}^*) = -\lambda\, \nabla g(\mathbf{x}^*)$ for some $\lambda > 0$

$g(x) = x_1 + x_2 - 1 \geq 0$

Lagrange Multiplier 102



- $\text{argmax}_{\mathbf{x}}\ f(\mathbf{x})$ s.t. $g(\mathbf{x}) \geq 0$
- At optimal $\mathbf{x}^*$, $g(\mathbf{x}^*) > 0$
  - $\nabla f(\mathbf{x}^*) + \lambda \nabla g(\mathbf{x}^*) = 0$ with $\lambda=0$
- At optimal $\mathbf{x}^*$, $g(\mathbf{x}^*) = 0$
  - $\nabla f(\mathbf{x}^*) + \lambda \nabla g(\mathbf{x}^*) = 0$ with $\lambda > 0$
- Either way… $\lambda\, g(\mathbf{x}^*) = 0$
- Summary:
  - $L(\mathbf{x}, \lambda) = f(\mathbf{x}) + \lambda g(\mathbf{x})$
  - Solve $\nabla_{x_i} L(\mathbf{x}, \lambda) = 0 \quad \nabla_{\lambda} L(\mathbf{x}, \lambda) = 0$
    s.t. $g(\mathbf{x}) \geq 0 \quad \lambda \geq 0 \quad \lambda\, g(\mathbf{x}) = 0$

Karush-Kuhn-Tucker (KKT):
If $g(x) > 0$, then $\lambda=0$

KKT Conditions: Inequality Case

- Karush-Kuhn-Tucker Theorem:
  If
  - function $f(x)$ has a minimum at x^* in the feasible set, and
  - $\nabla f(x^*)$ and $\nabla g_i(x^*)$, i=1,2,…,m exist,

  then $\exists$ m-dimensional vector λ such that

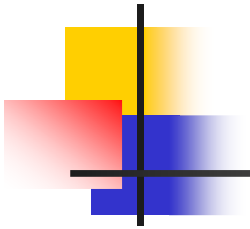
$$\lambda \geq 0$$

$$\nabla f(x^*) - \sum_{i=1}^{m} \lambda_i \nabla g_i(x^*) = 0$$

$$\lambda_i [\, g_i(x^*) \,] = 0, \quad \text{for } i=1,2,\ldots,m$$

- Each such (x^*, λ) is a KKT point;
  λ is the Dual Vector – aka the Lagrange Multipliers

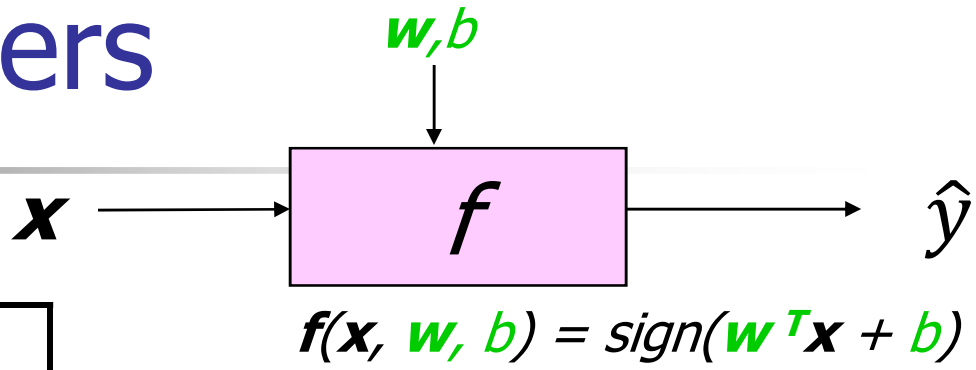
- These conditions are sufficient if dealing with a convex programming problem



Outline

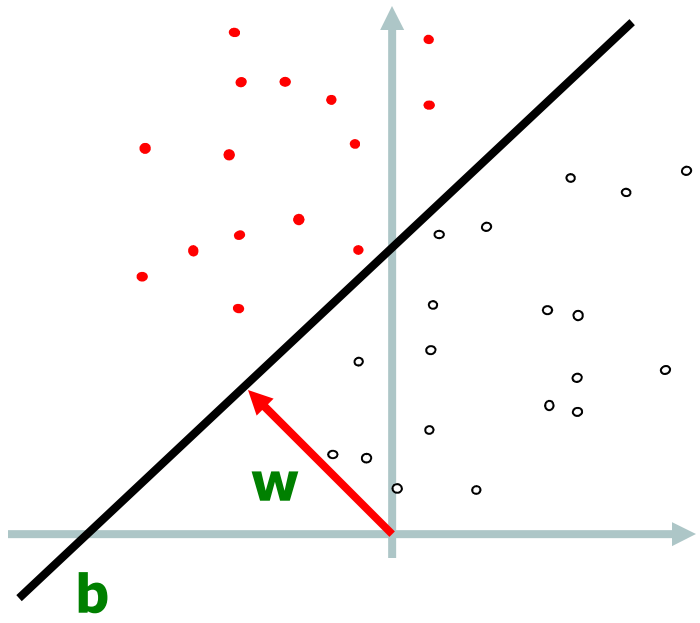
- Foundations
  - ~~Linear Programming;Primal/Dual~~
  - Constrained Optimization
    - Lagrange Multipliers
    - KKT
  - Perceptron Factoids
    - Dual Representation
- "Best" Linear Separator: Max Margin!
- Coping with Non-Linearly Separated Data
  - Slack Variables
- Kernel Trick
- Regression

Linear Classifiers



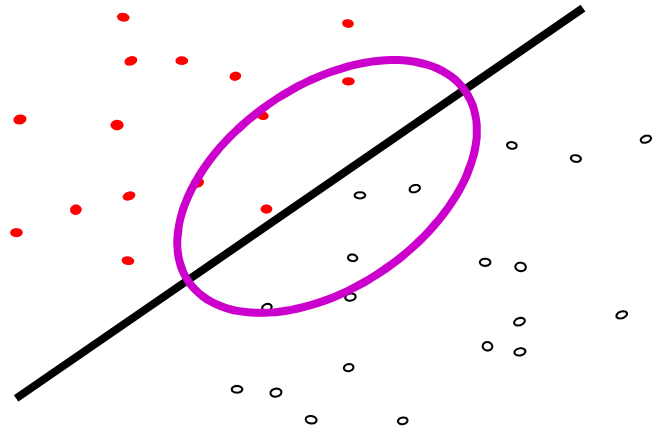
$f(\mathbf{x}, \mathbf{w}, b) = sign(\mathbf{w}^T\mathbf{x} + b)$

Input space	$\mathbf{x} \in \mathbf{X}$
Output space	$y \in Y = \begin{Bmatrix} +1 \\ -1 \end{Bmatrix}$
Real-valued fn:	$f: \mathbf{X} \rightarrow Y$
Training Set	$S = \begin{Bmatrix} [\mathbf{x}_1, y_1] \\ \vdots \\ [\mathbf{x}_m, y_m] \end{Bmatrix}$



Perceptron Training Rule

Initialize $\mathbf{w} = 0$
Do until done
  Predict "+" iff $\mathbf{w}^T \mathbf{x} > 0$
    else "–"
  Mistake on $y = +1$: $\mathbf{w} \leftarrow \mathbf{w} + \mathbf{x}$
  Mistake on $y = -1$: $\mathbf{w} \leftarrow \mathbf{w} - \mathbf{x}$



$\mathbf{w} \leftarrow \mathbf{w} + y\,\mathbf{x}$

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

- … only uses Informative Points (mistake driven)
- Coefficient α_i of point $\mathbf{x}_i$ reflects its 'difficulty'

Mistake Bound Theorem

From earlier...

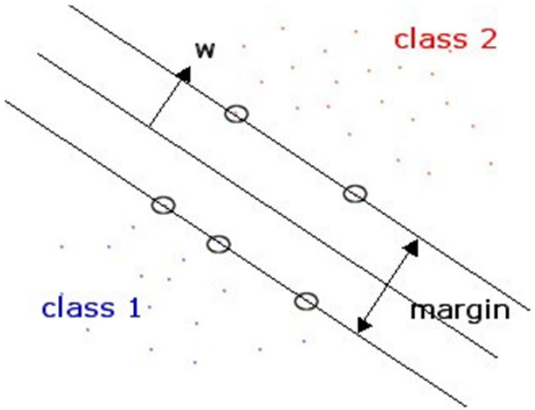
Theorem: [Rosenblatt 1960]

If data is consistent w/some linear threshold **w**,

then number of mistakes is $\leq (1/\Delta)^2$,

where $\Delta = \min_x \frac{|\mathbf{w} \cdot x|}{|\mathbf{w}| \times |x|}$ → ≈ margin

- Δ measures "wiggle room" available:
  If |x| = 1, then Δ is max, over all consistent planes, of minimum distance of example to that plane
- w is $\perp$ to separator, as w · x = 0 at boundary
- So |**w** · **x**| is projection of **x** onto plane, PERPENDICULAR to boundary line
  … ie, is distance from **x** to that line (once normalized)



Dual Representation

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

$\Rightarrow$ can re-write decision function

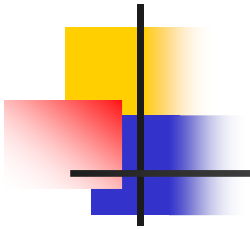
$$f(\mathbf{x}) = \langle \mathbf{w}, \mathbf{x} \rangle + b = \sum_i \alpha_i y_i \langle \mathbf{x}_i, \mathbf{x} \rangle + b$$

$\Rightarrow$ can re-write update rule:

If $y_j \left[\sum_i \alpha_i y_i \langle \mathbf{x}_i, \mathbf{x}_j \rangle + b \right] \leq 0$

Then $\alpha_j \leftarrow \alpha_j + \eta$

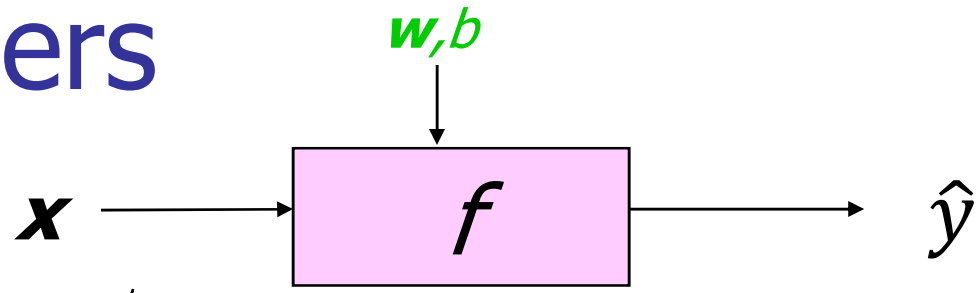
- In dual representation, datapoints $\mathbf{x}_j$ appears only inside dot products



Outline

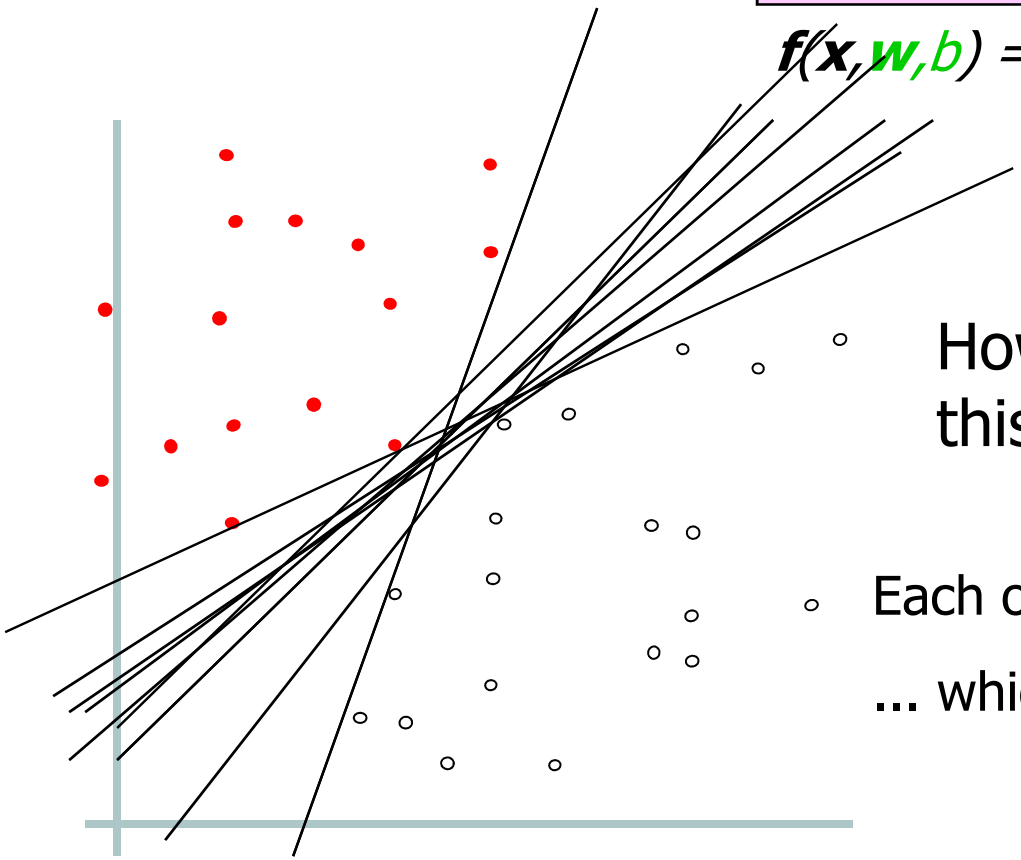
- Foundations
  - ~~Linear Programming;Primal/Dual~~
  - Constrained Optimization
    - Lagrange Multipliers
    - KKT
  - Perceptron Factoids
    - Dual Representation
- "Best" Linear Separator: Max Margin!
- Coping with Non-Linearly Separated Data
  - Slack Variables
- Kernel Trick
- Regression

Linear Classifiers



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

- denotes +1
- denotes –1

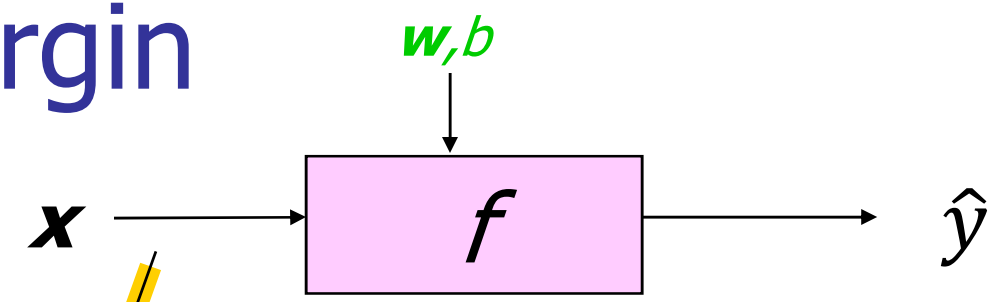


How to classify this data?

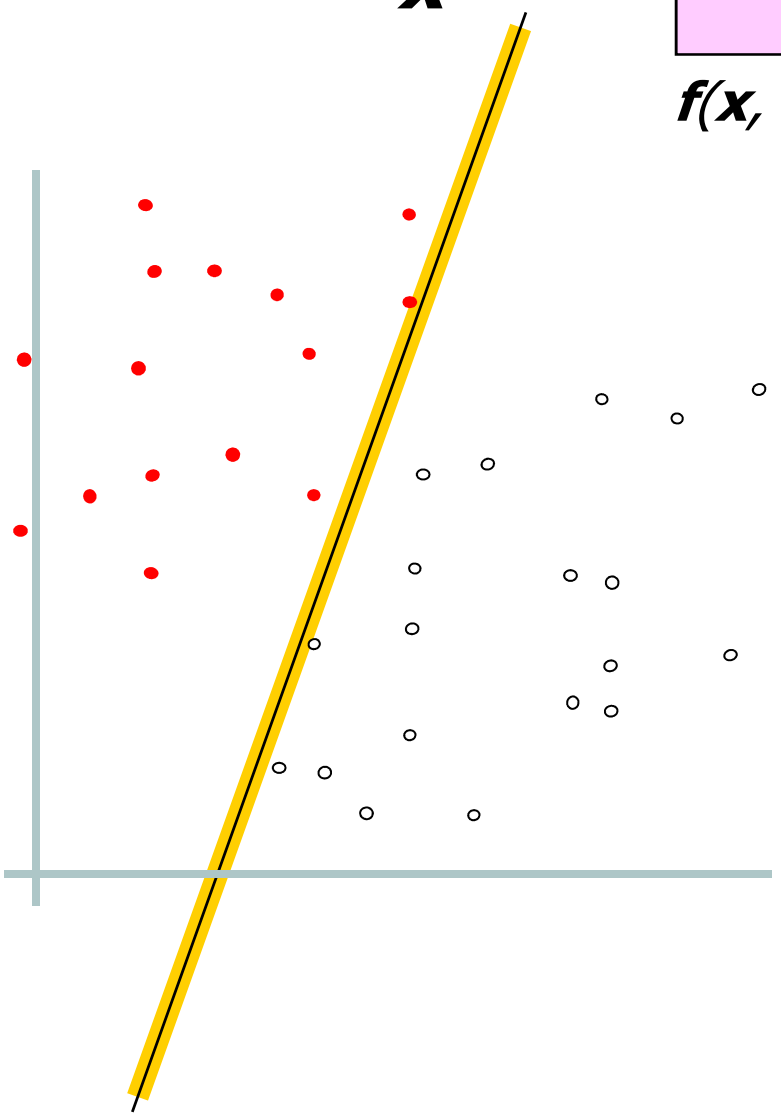
Each of these seems fine..

... which is best?

Classifier Margin

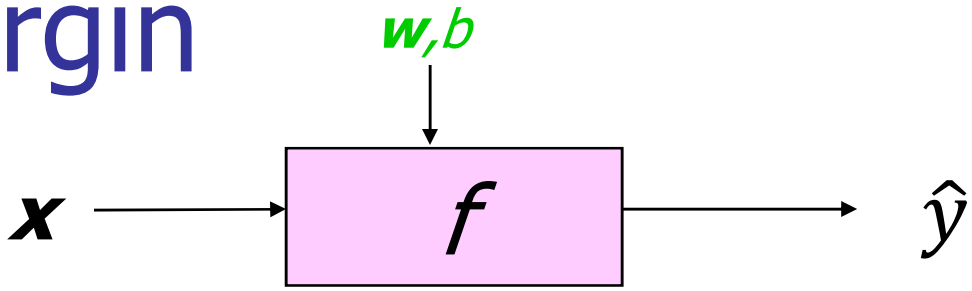


- denotes +1
- denotes –1



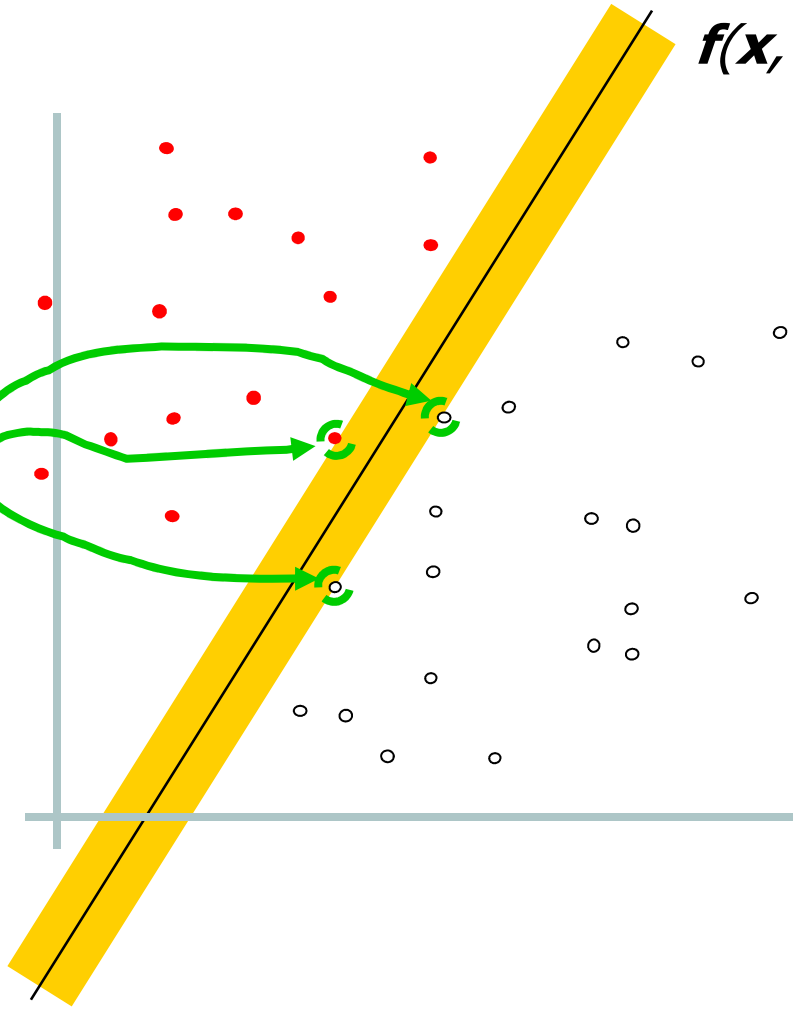
The margin of a linear classifier =
the width that the boundary could be increased by, before hitting a datapoint

Maximum Margin



$f(\mathbf{x}, \mathbf{w}, b) = sign(\mathbf{w}^T\mathbf{x} + b)$

- denotes +1
- denotes –1



Support Vectors are datapoints that "touch" the margin

Maximum margin linear classifier

=

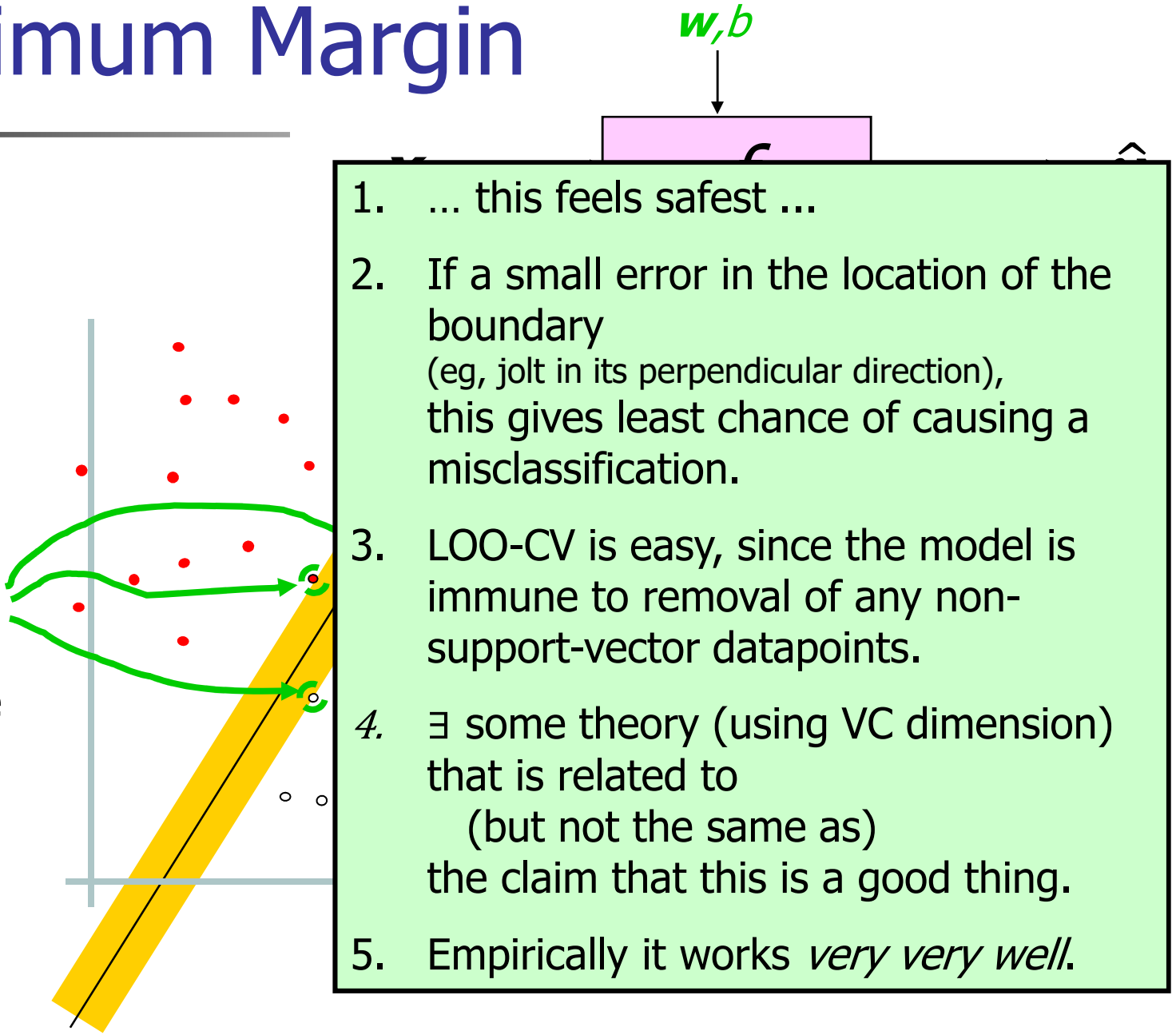
linear classifier with maximum margin

- Linear SVM

=

simplest kind of SVM

Maximum Margin



1. … this feels safest ...
2. If a small error in the location of the boundary (eg, jolt in its perpendicular direction), this gives least chance of causing a misclassification.
3. LOO-CV is easy, since the model is immune to removal of any non-support-vector datapoints.
4. ∃ some theory (using VC dimension) that is related to (but not the same as) the claim that this is a good thing.
5. Empirically it works *very very well*.

Goal of Max Margin Separator

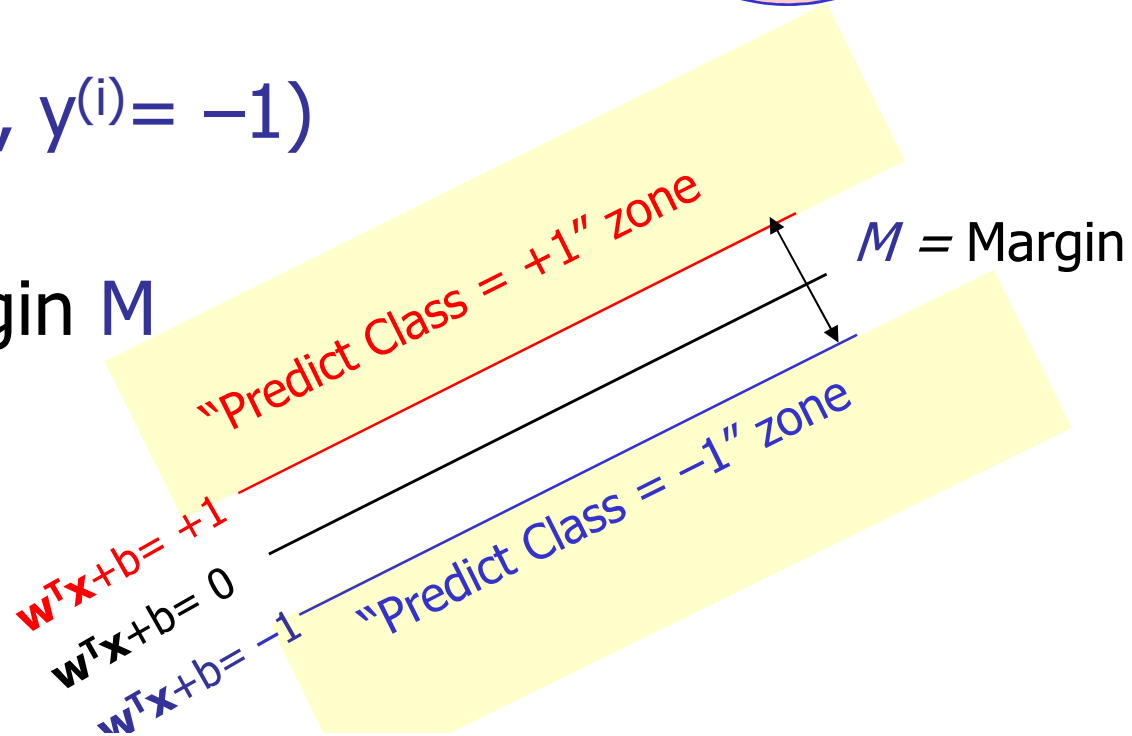
Want a linear separator

$\mathbf{w}$, b for $y = \mathbf{w}^T \boldsymbol{x} + b$

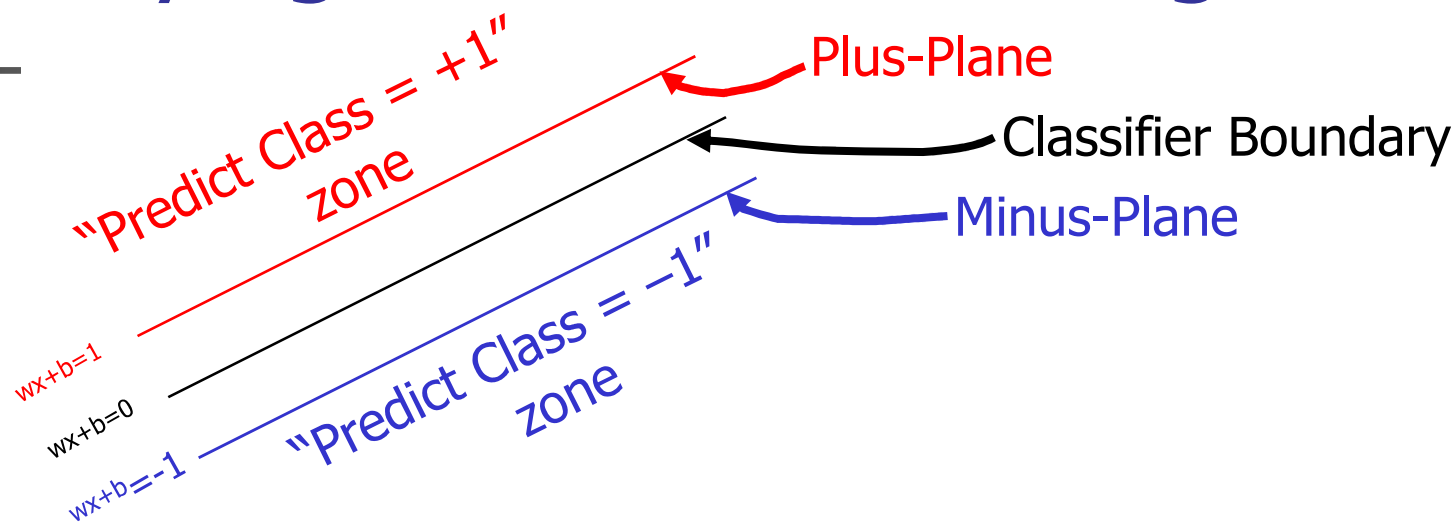
s.t.

- For all +points $(\mathbf{x}^{(i)}, y^{(i)}=+1)$
  $\mathbf{w}^T \mathbf{x}^{(i)} + b \geq +1$
- For all –points $(\mathbf{x}^{(i)}, y^{(i)}= -1)$
  $\mathbf{w}^T \mathbf{x}^{(i)} + b \leq -1$
- Maximizes the margin M

Why 1?
Any >0 constant works, as scales.
1 is convenient…



Specifying a Line and a Margin



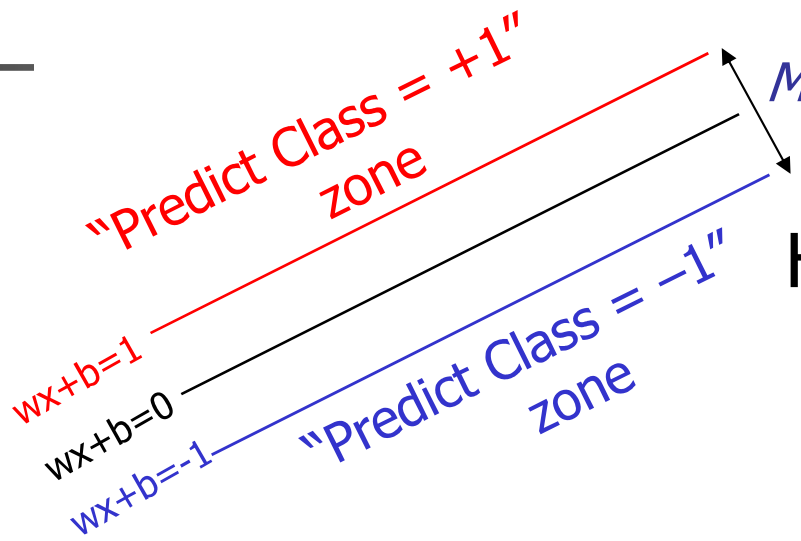
- Plus-plane = $\{ \mathbf{x} : \mathbf{w}^T\mathbf{x} + b = +1 \}$
- Minus-plane = $\{ \mathbf{x} : \mathbf{w}^T\mathbf{x} + b = -1 \}$

Classify as..

+1	if	$\boldsymbol{w}^T\boldsymbol{x} + b \geq 1$
−1	if	$\boldsymbol{w}^T\boldsymbol{x} + b \leq -1$
(whatever)	if	$-1 < \boldsymbol{w}^T\boldsymbol{x} + b < 1$

Never happens

Computing the Margin Width



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

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

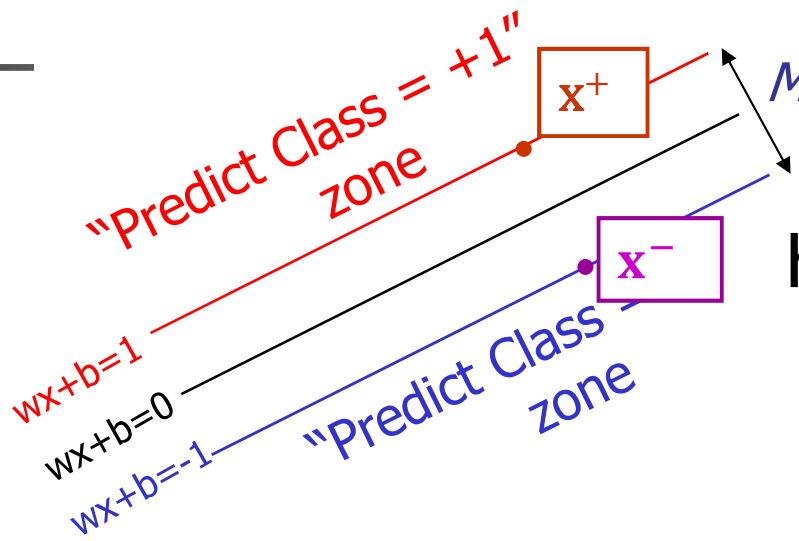
Claim: Plus-plane is || to Minus-Plane

The vector **w** is perpendicular to the Plus Plane. Why?

- Definitions: "vector" ≡ "point"
- **a** perpendicular to **b** iff $\mathbf{a}^T\mathbf{b} = 0$

Let **u** and **v** be two vectors on the Plus Plane. What is $\boldsymbol{w}^T (\boldsymbol{u} - \boldsymbol{v})$?

Computing the Margin Width

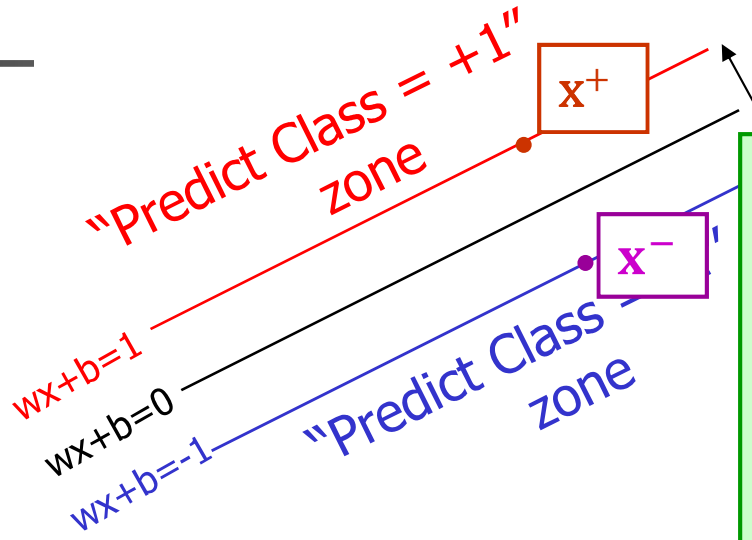


How to compute *M* in terms of ***w*** and *b* ?

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

Any location in $\Re^m$: not necessarily a datapoint

Computing the Margin Width



Line from $\mathbf{x}^-$ to $\mathbf{x}^+$ is ⊥ to the planes.

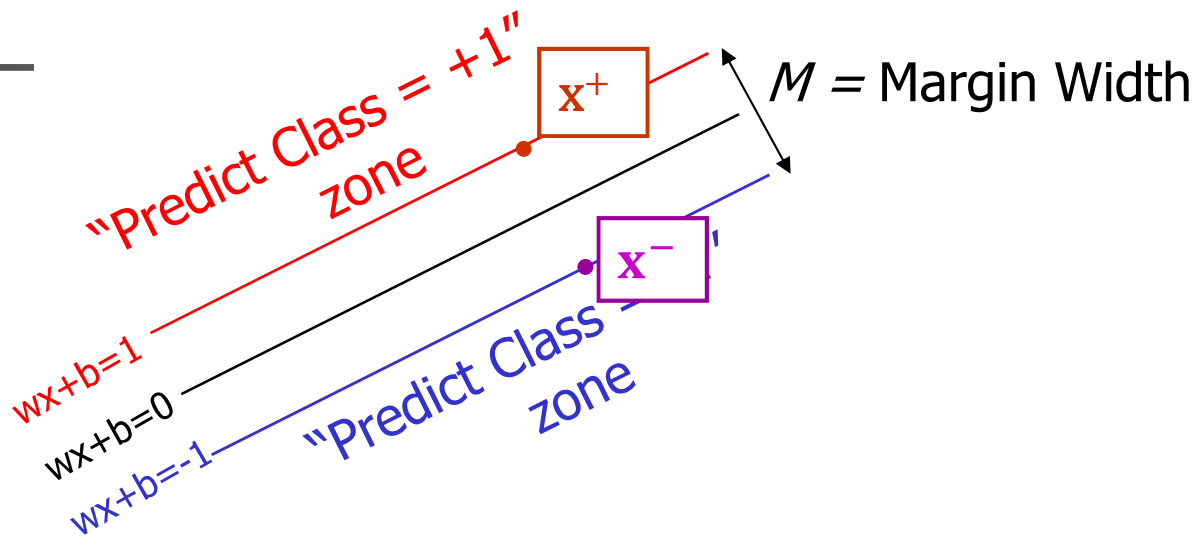
So to get from $\mathbf{x}^-$ to $\mathbf{x}^+$, travel some distance in the direction of $\boldsymbol{w}$

- Plus-plane = $\{ \boldsymbol{x} : \boldsymbol{w}^T \boldsymbol{x} + b =$
- Minus-plane = $\{ \boldsymbol{x} : \boldsymbol{w}^T \boldsymbol{x} + b = -1 \}$
- ular to the Plus Plane
- $\mathbf{x}^-$ = any point the minus plane
- $\mathbf{x}^+$ =the point in plus-plane closest to $\mathbf{x}^-$
- Claim: $\mathbf{x}^+ = \mathbf{x}^- + \lambda \boldsymbol{w}$ for some value of $\lambda \in \Re^+$. Why?

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

A ation in $\Re^m$: n ecessarily a atapoint

Computing the Margin Width

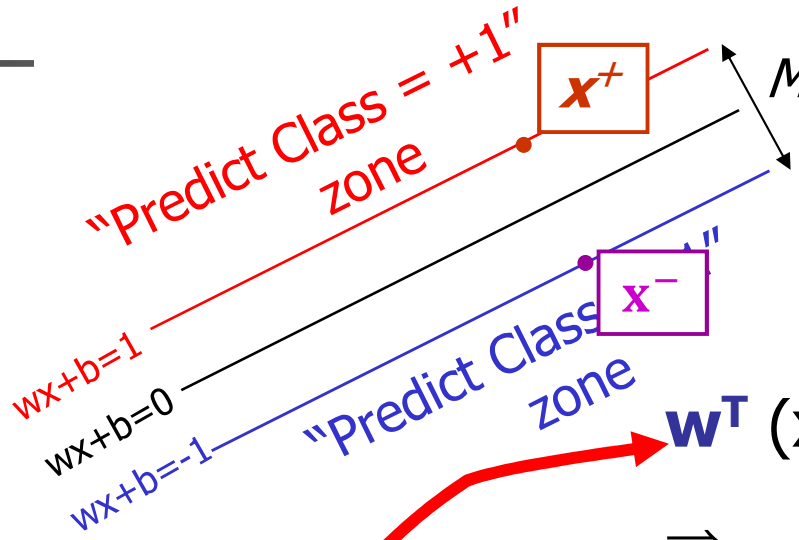


Given…

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

… easy to get M in terms of **w** and b

Computing the Margin Width



Given…

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

… easy to get M in terms of **w** and b

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

$$\Rightarrow$$

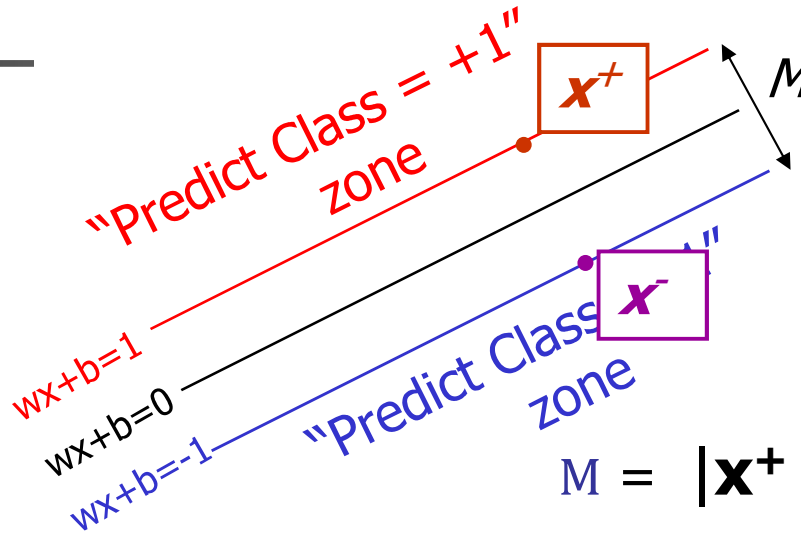
$$(\mathbf{w}^T \mathbf{x}^- + b) + \lambda \mathbf{w}^T \mathbf{w} = 1$$

$$\Rightarrow$$

$$-1 + \lambda \mathbf{w}^T \mathbf{w} = 1$$

$$\Rightarrow \quad \lambda = \frac{2}{\mathbf{w}^T \mathbf{w}}$$

Computing the Margin Width



M = Margin Width = $\frac{2}{\sqrt{\mathbf{w}^T\mathbf{w}}}$

Given ...

- $\mathbf{w}^T \mathbf{x}^+ + b = +1$
- $\mathbf{w}^T \mathbf{x}^- + b = -1$
- $\mathbf{x}^+ - \mathbf{x}^- = \lambda\, \mathbf{w}$
- $|\mathbf{x}^+ - \mathbf{x}^-| = M$
- $\lambda = \frac{2}{\mathbf{w}\cdot\mathbf{w}}$

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

$$= \lambda |\mathbf{w}| = \frac{2}{\mathbf{w}^T\mathbf{w}} \sqrt{\mathbf{w}^T\mathbf{w}}$$

$$= \frac{2}{\sqrt{\mathbf{w}^T\mathbf{w}}}$$

So... to maximize $M = \frac{2}{\sqrt{\mathbf{w}^T\mathbf{w}}}$

just minimize $\mathbf{w}^T\mathbf{w}$

Wait...OMG, I forgot the data!

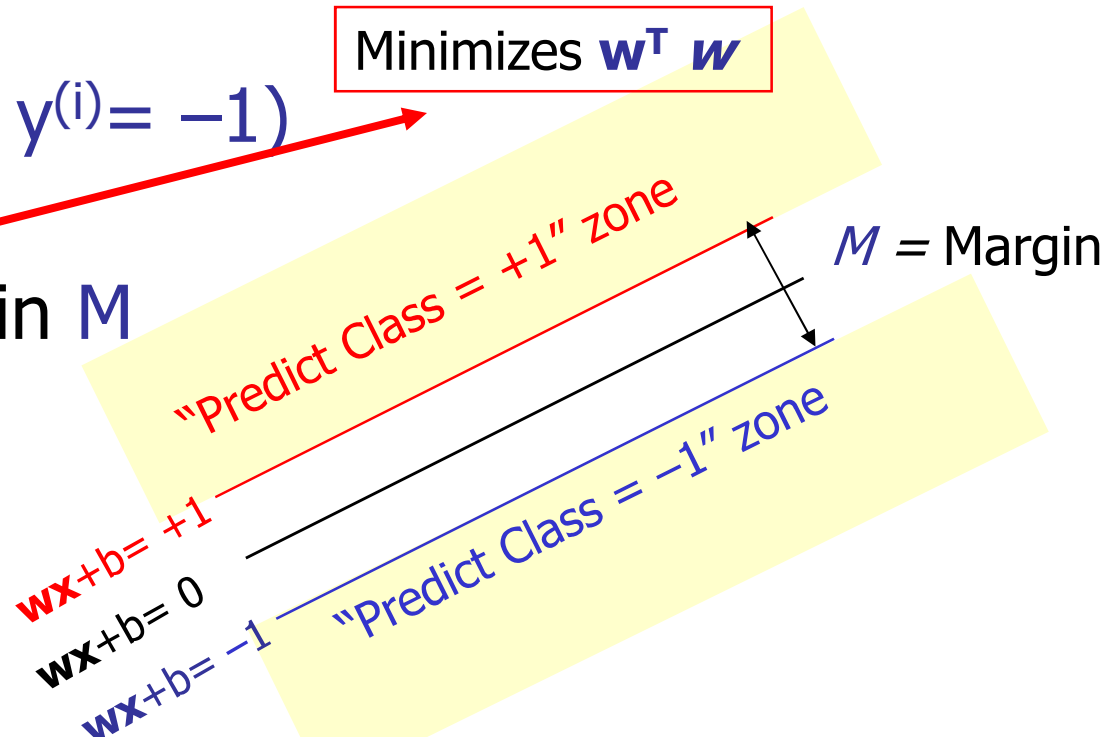
Goal of Max Margin Separator

Want a linear separator

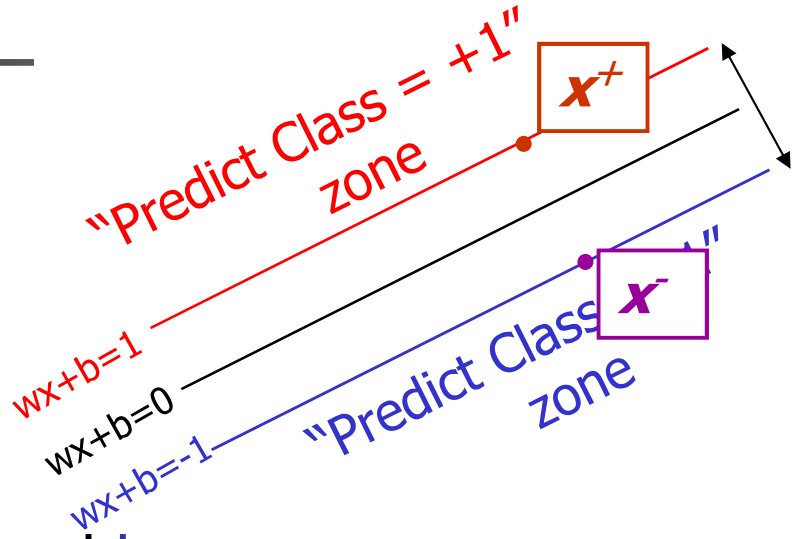
$\mathbf{w}$, b for $y = \mathbf{w}^T \boldsymbol{x} + b$

s.t.

- For all +points $(\mathbf{x}^{(i)}, y^{(i)}=+1)$
  $\mathbf{w}^T \mathbf{x}^{(i)} + b \geq +1$
- For all –points $(\mathbf{x}^{(i)}, y^{(i)}= -1)$
  $\mathbf{w}^T \mathbf{x}^{(i)} + b \leq -1$
- Maximizes the margin M



Learning the Maximum Margin Classifier



$$M = \text{Margin Width} = \frac{2}{\sqrt{\mathbf{w}^T \mathbf{w}}}$$

Given **w** and b we can

- Compute whether each data point is in correct half-plane
- Compute the width of the margin

But… need 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?

Rewrite Problem

Minimize ½ $\boldsymbol{w}^T \boldsymbol{w}$

s.t. $\boldsymbol{w}^T \boldsymbol{x}_k + b \geq 1$ if $y_k = 1$

$\boldsymbol{w}^T \boldsymbol{x}_k + b \leq -1$ if $y_k = -1$

$y_k(\boldsymbol{w} \cdot \boldsymbol{x}_k + b) - 1 \geq 0$

Lagrange Multiplier

Equivalent optimization…

$L(\mathbf{w}, b, \lambda) = \frac{1}{2} |\mathbf{w}|^2 - \sum_k \lambda_k [\, y_k (\mathbf{w}^T \mathbf{x}_k + b) - 1 \,]$

$\min_{\mathbf{w},b} L(\mathbf{w}, b, \lambda)$

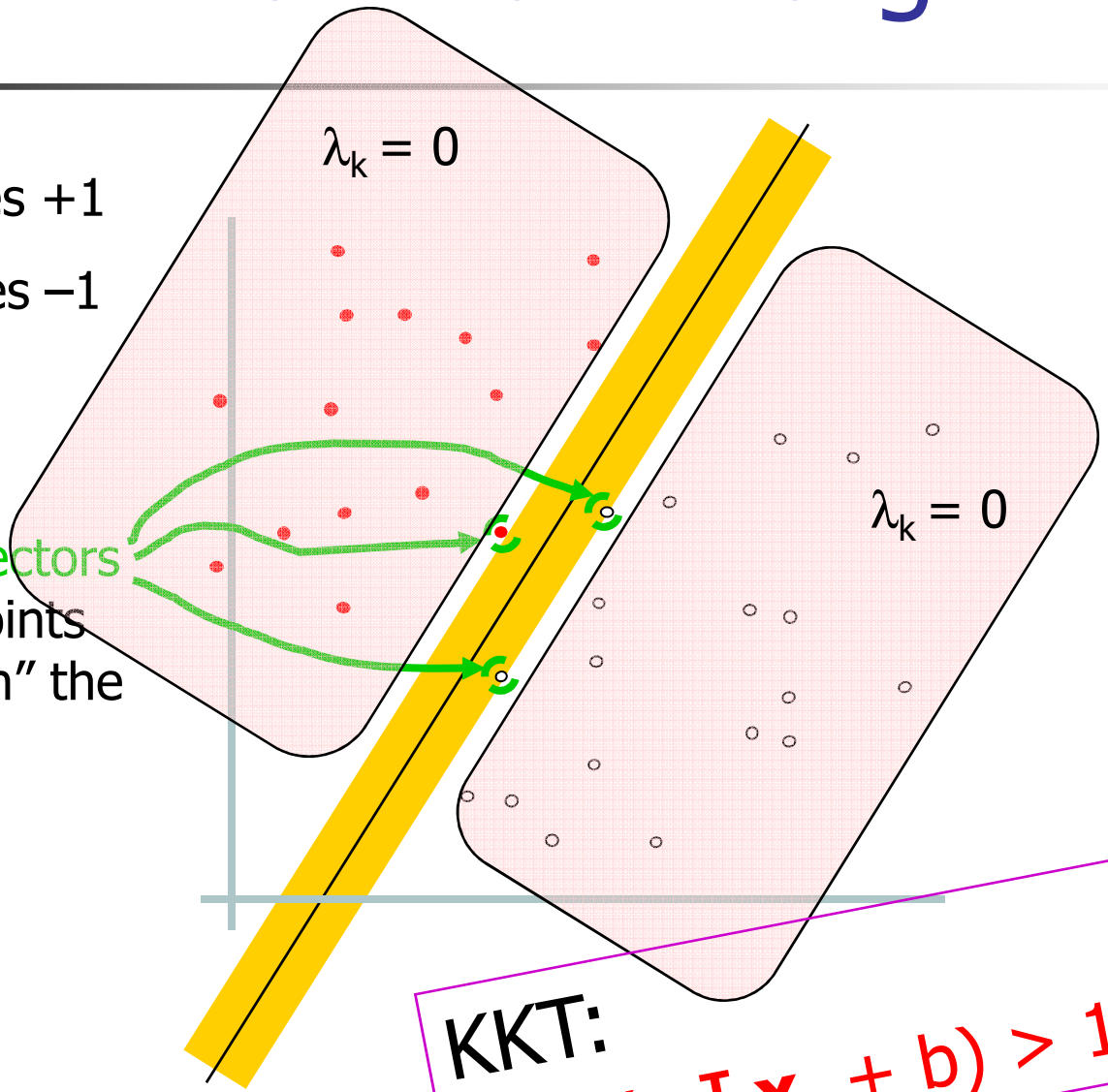
s.t. $\lambda \geq 0$

KKT:

$y_k (\mathbf{w}^T \mathbf{x}_k + b) > 1 \Rightarrow \lambda_k = 0$

Maximum Margin

- denotes +1
- denotes –1



Support Vectors are datapoints that "touch" the margin

KKT:

$$y_k (\mathbf{w}^T \mathbf{x}_k + b) > 1 \Rightarrow \lambda_k = 0$$

Solving Constrained Optimization

$$\min_{\mathbf{w},b} L(\mathbf{w}, b, \boldsymbol{\lambda}) = \tfrac{1}{2} |\mathbf{w}|^2 - \sum_k \lambda_k [\, y_k (\mathbf{w}^T \mathbf{x}_k + b) - 1 \,]$$

s.t. $\boldsymbol{\lambda} \geq \mathbf{0}$

Setting derivatives to 0…

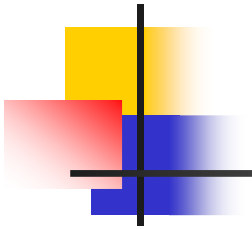
- $\mathbf{w} = \sum_m \lambda_m y_m \mathbf{x}_m$
- $0 = \sum_m \lambda_m y_m$

Substitute back into $L(\cdot,\cdot,\cdot)$:

Find $\boldsymbol{\lambda} \geq \mathbf{0}$ that maximizes

$$L(\boldsymbol{\lambda}) = \sum_k \lambda_k - \tfrac{1}{2} \sum_k \lambda_k y_k \left(\sum_m \lambda_m y_m \mathbf{x}_m\right)^T \mathbf{x}_k)$$

$$= \sum_k \lambda_k - \tfrac{1}{2} \sum_k \sum_m \lambda_k \lambda_m y_k y_m (\mathbf{x}_k^T \mathbf{x}_m)$$



Learning via Quadratic Programming

- QP is a well-studied class of optimization alg's that
  - maximize a **quadratic function** of some real-valued variables
  - subject to **linear constraints**

- Popular ML approach:
  - Describe your learning problem as optimization…
  - …and give it to somebody else to solve!

Quadratic Programming – in general

Find $\arg\min_{\mathbf{w}} \; c + \mathbf{d}^T\mathbf{w} + \frac{\mathbf{w}^T\mathbf{K}\mathbf{w}}{2}$ ← Quadratic criterion

Subject to

$$\begin{aligned} a_{11}w_1 + a_{12}w_2 + \ldots + a_{1m}w_m &\le b_1 \\ a_{21}w_1 + a_{22}w_2 + \ldots + a_{2m}w_m &\le b_2 \\ &\vdots \\ a_{n1}w_1 + a_{n2}w_2 + \ldots + a_{nm}w_m &\le b_n \end{aligned}$$

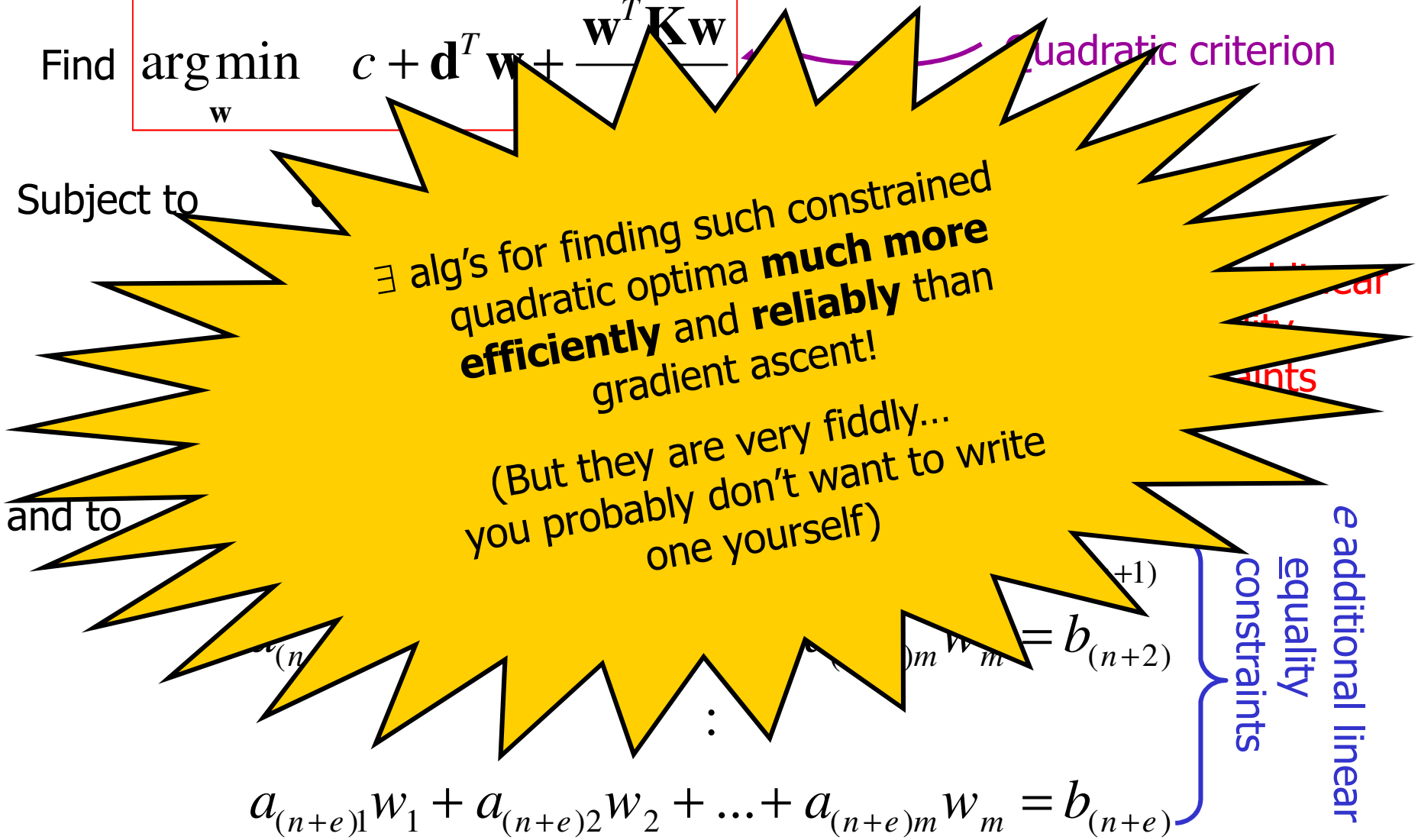
n additional linear inequality constraints

and to

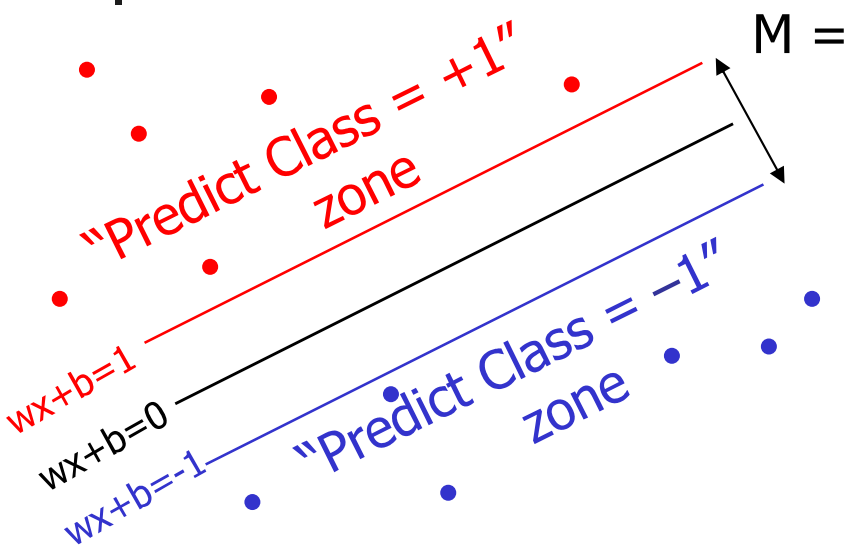
$$\begin{aligned} a_{(n+1)1}w_1 + a_{(n+1)2}w_2 + \ldots + a_{(n+1)m}w_m &= b_{(n+1)} \\ a_{(n+2)1}w_1 + a_{(n+2)2}w_2 + \ldots + a_{(n+2)m}w_m &= b_{(n+2)} \\ &\vdots \\ a_{(n+e)1}w_1 + a_{(n+e)2}w_2 + \ldots + a_{(n+e)m}w_m &= b_{(n+e)} \end{aligned}$$

e additional linear equality constraints

Quadratic Programming – in general



Learning the Maximum Margin Classifier



$$M = \frac{2}{\sqrt{\mathbf{w}^T\mathbf{w}}}$$

Given guess of **w**, b, can…

- Compute whether each data point is in the correct half-plane
- Compute the margin width
- Leading to *R* datapoints, $\{[\mathbf{x}_k, y_k]\}$ where $y_k \in \{+1, -1\}$

How many constraints? *R*

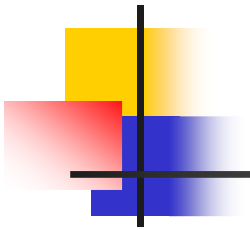
What is quadratic optimization criterion?

Minimize $\mathbf{w}^T\mathbf{w}$

What are they?

$\mathbf{w}^T\mathbf{x}_k + b \geq 1$ if $y_k = 1$

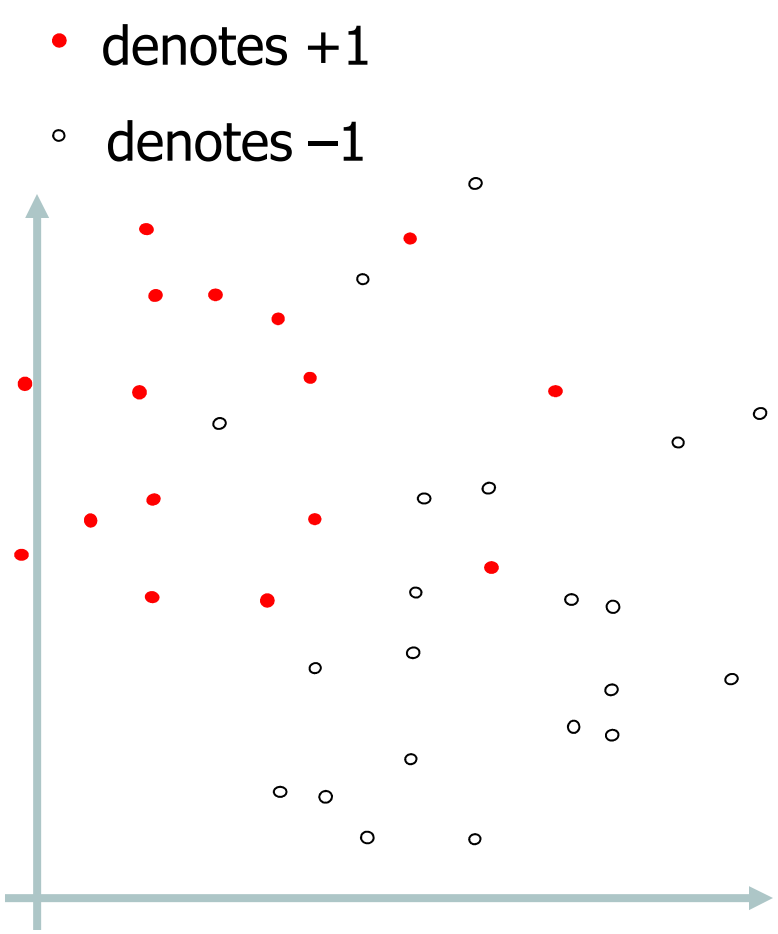
$\mathbf{w}^T\mathbf{x}_k + b \leq -1$ if $y_k = -1$



Outline

- Foundations
  - ~~Linear Programming;Primal/Dual~~
  - Constrained Optimization
    - Lagrange Multipliers
    - KKT
  - Perceptron Factoids
    - Dual Representation
- "Best" Linear Separator: Max Margin!
- Coping with Non-Linearly Separated Data
  - Slack Variables
- Kernel Trick
- Regression

What if not linearly separable?



Earlier analysis does not work!

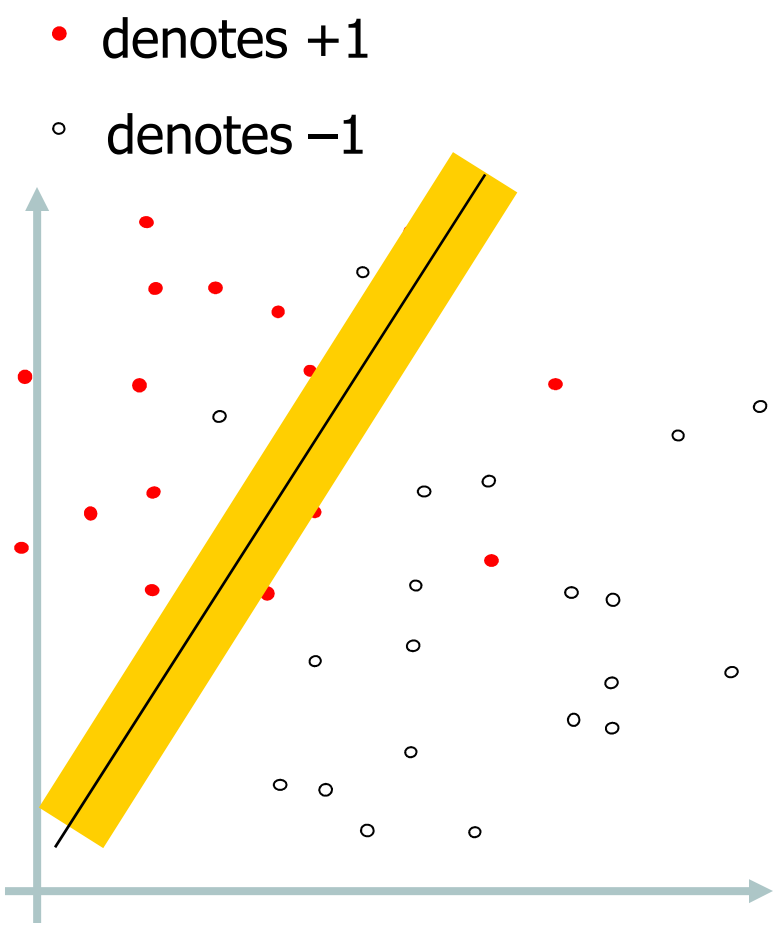
What should we do?

Idea 1: Find **w** s.t.

- minimum $\mathbf{w}^T\mathbf{w}$
- minimum # of training set errors.

Problem: Minimizing *TWO* objectives is ill-defined !

What if not linearly separable?



#train error = 2 + 4

Earlier analysis does not work!

What should we do?

Idea 1.1: Find **w** that

Minimizes

$\mathbf{w}^T\mathbf{w} + C$ (#train errors)

Tradeoff parameter

But… a ***serious*** practical problem dooms this approach

What if not linearly separable?

Earlier analysis does not work!

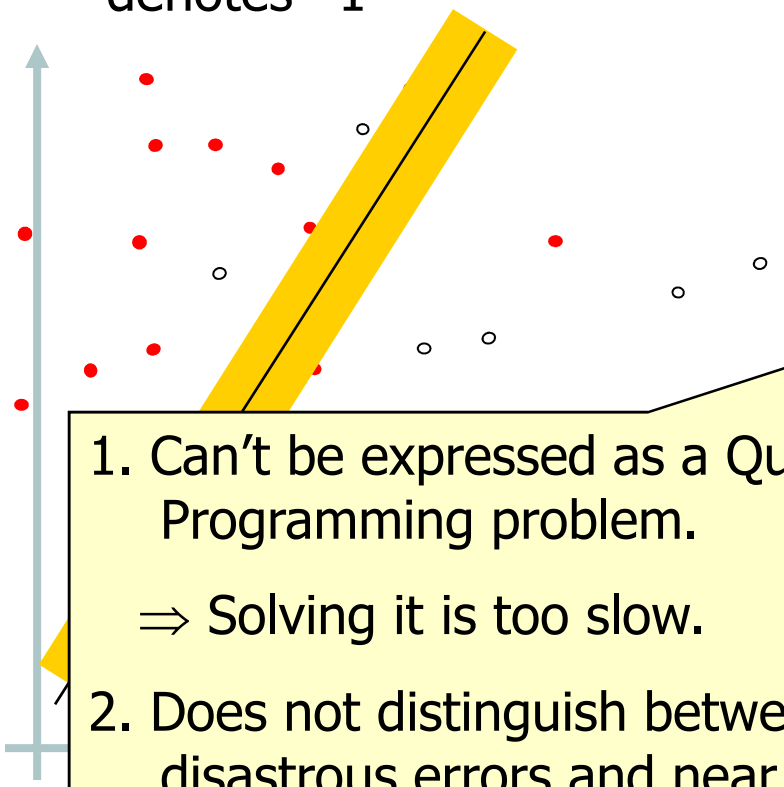
What should we do?

Idea 1.1: Find **w** that

Minimizes

$\mathbf{w}^T\mathbf{w} + C$ (#train errors)

Tradeoff parameter



1. Can't be expressed as a Quadratic Programming problem.

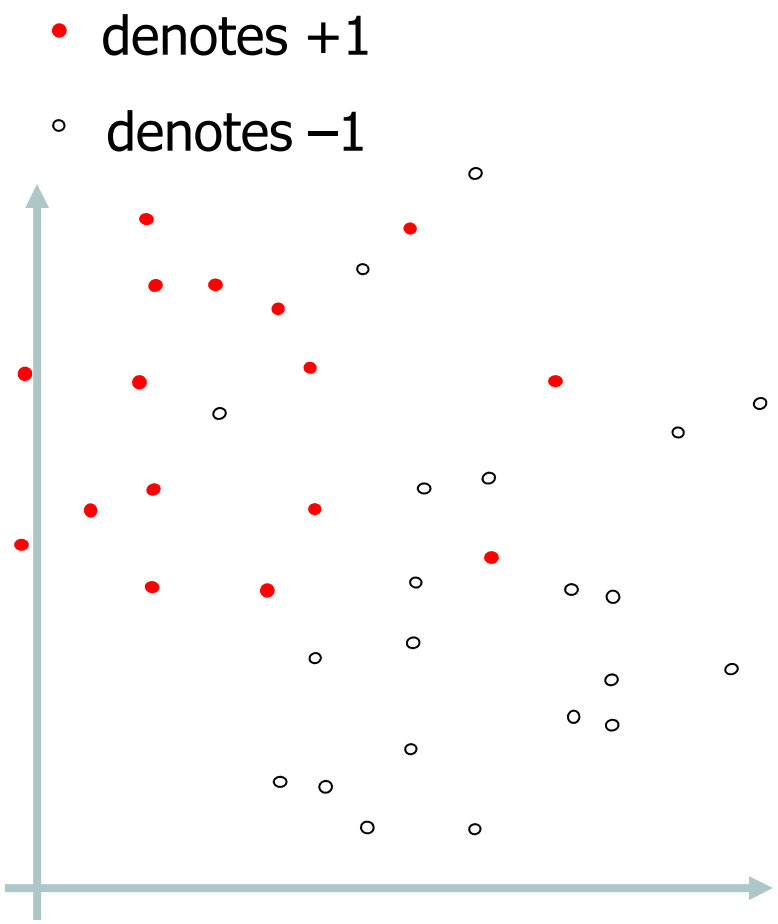
   ⇒ Solving it is too slow.

2. Does not distinguish between disastrous errors and near misses

#train error = 2 + 4

?? any other ideas ??

What if not linearly separable?



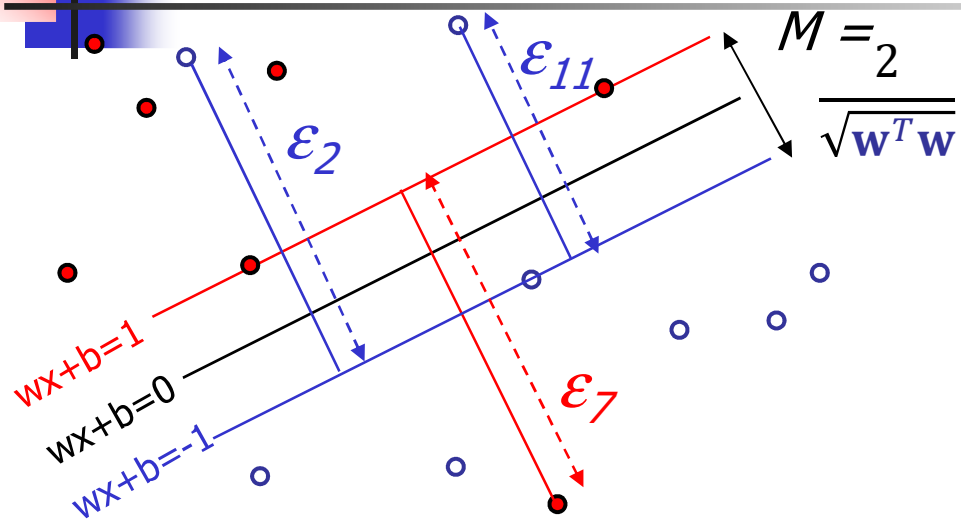
Earlier analysis does not work!

What should we do?

Idea 2: Find **w** that

Minimizes
$\mathbf{w}^T\mathbf{w}$ + *C (distance from incorrectly labeled points to their correct place)*

Learning Maximum Margin with Noise



Given guess of ***w***, *b*, can…

- Compute whether each data point is in the correct half-plane
- Compute the margin width

R datapoints, $\{[\mathbf{x}_k, y_k]\}$ where $y_k \in \{+1, -1\}$

What is quadratic optimization criterion?

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

How many constraints? *R*

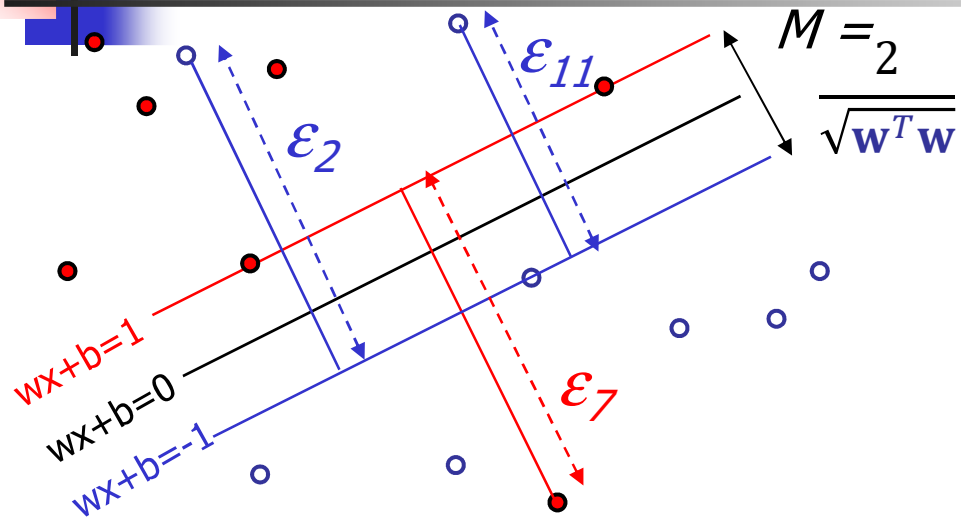
What should they be?

$\mathbf{w}^T\mathbf{x}_k + b \geq (1 - \varepsilon_k)$ if $y_k = 1$

$\mathbf{w}^T\mathbf{x}_k + b \leq (-1 + \varepsilon_k)$ if $y_k = -1$

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

Learning Maximum Margin with Noise



Given guess of ***w***, *b*, can…

- Compute whether each data point is in the correct half-plane
- Compute the margin width

R datapoints, $\{[\mathbf{x}_k, y_k]\}$ where $y_k \in \{+1, -1\}$

What is quadratic optimization criterion?

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

Called "slack variables"

How many constraints? *2R*

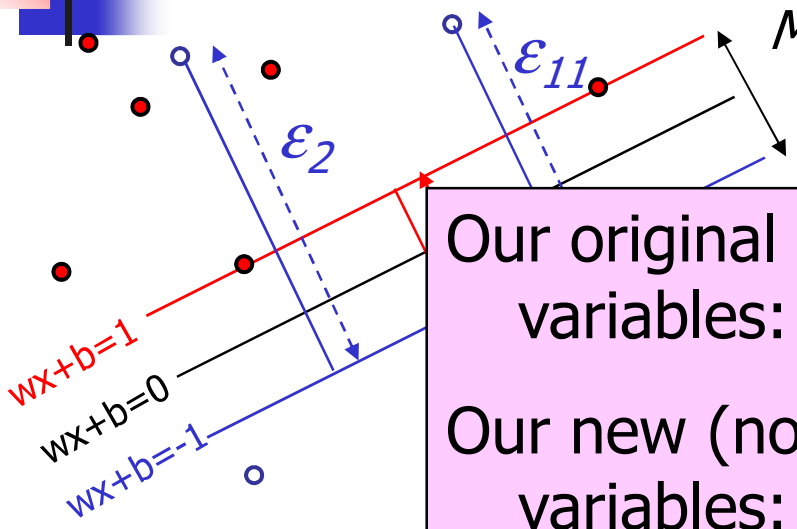
What should they be?

$\mathbf{w}^T\mathbf{x}_k + b \geq (1-\varepsilon_k)$ *if* $y_k = 1$

$\mathbf{w}^T\mathbf{x}_k + b \leq (-1+\varepsilon_k)$ *if* $y_k = -1$

$\varepsilon_k \geq 0$ for all *k*

Learning Maximum Margin with Noise



$M = \frac{2}{\sqrt{\mathbf{w}^T \mathbf{w}}}$

Give

- Compute wheth data

ct

idth

y_k)

m = # input dimensions

Our original (noiseless data) QP had $m+1$ variables: $w_1, w_2, \ldots w_m$ and b.

Our new (noisy data) QP has $m+1+R$ variables: $w_1, w_2, \ldots w_m, b, \varepsilon_k, \varepsilon_1, \ldots \varepsilon_R$

R = # instances

What is quadratic optimization criterion?

Minimize $\frac{\boldsymbol{w}^T \boldsymbol{w}}{2} + C \sum_{k=1}^{R} \varepsilon_k$

How many constrain

What should they be?

$\boldsymbol{w}^T \boldsymbol{x}_k + b \geq (1 - \varepsilon_k)$ if $y_k = 1$

$\boldsymbol{w}^T \boldsymbol{x}_k + b \leq (-1 + \varepsilon_k)$ if $y_k = -1$

$\varepsilon_k \geq 0$ for all k

Managing trade-off "C"

- Minimize $\frac{\mathbf{w}^{\mathrm{T}}\mathbf{w}}{2} + C\sum_{k=1}^{R}\varepsilon_k$
- Big ***C*** $\Rightarrow$ "Fit the training data as much as possible!"
  (at the expense of maximizing margin)
- Small ***C*** $\Rightarrow$"Maximize the margin as much as possible!"
  (at the expense of fitting the training data)
- Best value: Internal cross-validation

Solving Constrained Optimization

$$\min_{\mathbf{w},b} L(\mathbf{w}, b, \boldsymbol{\lambda}, \boldsymbol{\varepsilon}) = \tfrac{1}{2} |\mathbf{w}|^2 + C \sum_k \varepsilon_k - \sum_k \lambda_k [\, y_k(\mathbf{w}^T \mathbf{x}_k + b) - (1-\varepsilon_k) \,]$$

s.t. $\boldsymbol{\lambda}, \boldsymbol{\varepsilon} \geq 0$

Setting derivatives to 0…

- $\mathbf{w} = \sum_m \lambda_m y_m \mathbf{x}_m$
- $0 = \sum_m \lambda_m y_m$

To incorporate slack variables ε_k

Just add constraints:

$$0 \leq \lambda_k \leq C$$

Actually:

$L(\boldsymbol{\lambda}) = \ldots + \sum_k (C - \lambda_k)\varepsilon_k$

But $\varepsilon_k > 0 \Rightarrow \lambda_k = C$ …

Substitute back into $L(..)$:

Find $\boldsymbol{\lambda} \geq \mathbf{0}$ that minimizes

$$L(\boldsymbol{\lambda}) = \sum_k \lambda_k - \tfrac{1}{2} \sum_k \sum_m \lambda_k \lambda_m y_k y_m (\mathbf{x}_k^T \mathbf{x}_m)$$

An Equivalent QP

$$\max_{\lambda} \sum_{k=1}^{R} \lambda_k - \frac{1}{2} \sum_{k=1}^{R} \sum_{r=1}^{R} \lambda_k \lambda_r Q_{kr} \quad \text{where} \quad Q_{kr} = \mathrm{y_k\, y_r\, (\mathbf{x}_k{}^T \mathbf{x}_r)}$$

Subject to the constraints:

$$0 \le \lambda_k \le C \quad \forall k \qquad \sum_{k=1}^{R} \lambda_k y_k = 0$$

Then define:

$$\mathbf{w} = \sum_{\text{k s.t. } \lambda_k > 0} \lambda_k \mathrm{y_k} \mathbf{x_k}$$

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

where $K = \arg\max_k \lambda_k$

Then classify with:

f(**x**,**w**,b) = sign(**w**^{T} **x** + b)

An Equivalent QP

$$\max_{\lambda} \sum_{k=1}^{R} \lambda_k - \frac{1}{2} \sum_{k=1}^{R} \sum_{r=1}^{R} \lambda_k \lambda_r Q_{kr} \quad \text{where} \quad Q_{kr} = \mathrm{y_k\, y_r\, (x_k{}^T x_r)}$$

Subject to the constraints:

$$0 \le \lambda_k \le C$$

$$\sum_{k=1}^{R}$$

$\mathbf{x}_k$ only appears in dot product!

Then define:

$$\mathbf{w} = \sum_{\text{k s.t. } \lambda_k > 0} \lambda_k \mathrm{y_k} \mathbf{x_k}$$

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

where $K = \arg\max_k \lambda_k$

Datapoints with $\lambda_k > 0$ == support vectors

Then classify with:

f(**x**, **w**, b) = sign($\mathbf{w}^T \mathbf{x}$ + b)

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

(probably << R)

An Equivalent QP

$$\max_{\lambda} \sum_{k=1}^{R} \lambda_k - \frac{1}{2} \sum_{k=1}^{R} \sum_{r=1}^{R} \lambda_k r Q_{kr} \quad \text{where} \quad Q_{kr} = \mathrm{y_k\, y_r\, (x_k{}^T x_r)}$$

Subject to the constra

dot product!

Why use this equivalent QP?

- QP packages can optimize it more quickly
- Stay tuned…

Then de

$$\boldsymbol{w} = \sum$$

k s.t. $\alpha_k > 0$

+ b)

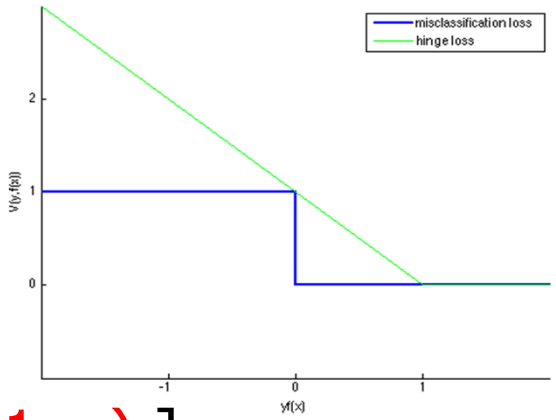
$$b = y_K(1 - \epsilon_K) - \boldsymbol{x}_K \cdot \boldsymbol{w}$$

where $K = \arg\max_k \alpha_k$

not this sum only needs to be over the support vectors.

(probably << R)

Hinge Loss



$\min_{\mathbf{w},b} L(\mathbf{w}, b, \lambda, \varepsilon) = \frac{1}{2} |\mathbf{w}|^2 +$

$C \sum_k \varepsilon_k - \sum_k \lambda_k [y_k(\mathbf{w}^T \mathbf{x}_k + b) - (1-\varepsilon_k)]$

s.t. $\lambda, \varepsilon \geq 0$

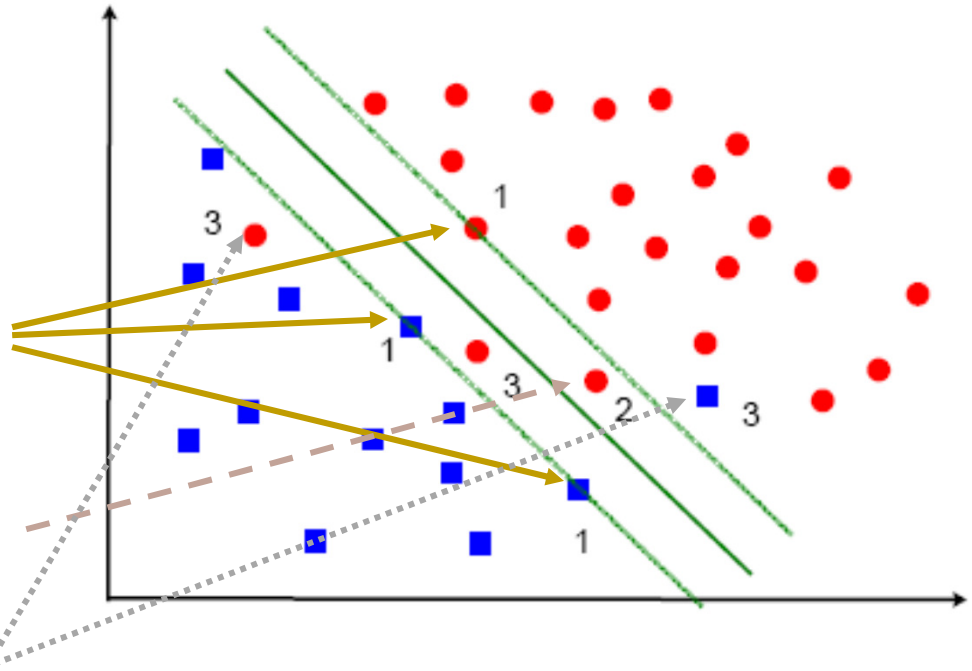
- $|\mathbf{w}|^2$ is regularizer

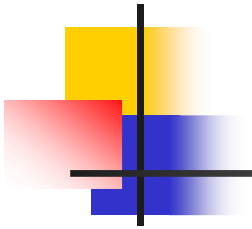
Hidden, as part of HW …

Types of Support Vectors

Support Vectors (s.v.s)

1. Margin s.v. $\xi_i = 0$
   - correct
2. Non-Margin s.v. $\xi_i < 0$
   - Correct (in margin)
3. Non-Margin s.v. $\xi_i > 0$
   - Error





What do we have?

- Method for learning a
  maximum-margin linear classifier
  when the data are …
  - "Linearly separable" –
    ∃ line that gets 0 training error
- But if not linearly separable
  - ie, no such line,

  must trade-off between
  - maximizing margin vs
  - minimizing "stuff-is-on-the-wrong-side-ness"
- … OR DO WE?? Kernels!