# Support Vector Machines

आई आई टी हैदराबाद
IIT Hyderabad

# Classification Methods

- k-Nearest Neighbors
- Decision Trees
- Naïve Bayes
- Ensemble Methods (Boosting, Random Forests)
- Logistic Regression
- Support Vector Machines
- Neural Networks

# SVM: Overview and History

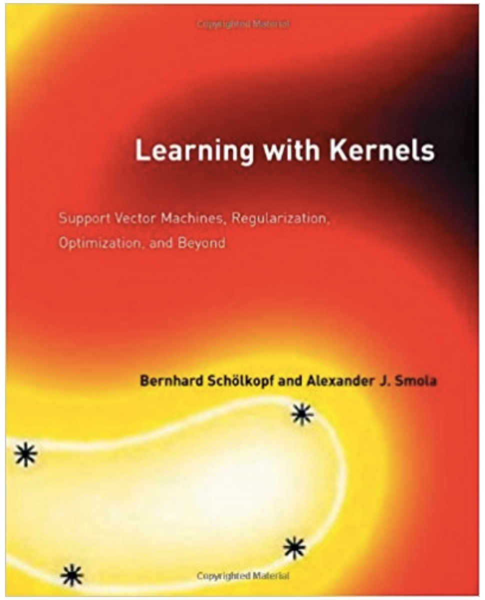
- A discriminative classifier
  - Non-parametric, Inductive
- SVM was developed in 1992 by Vapnik, Guyon and Boser
- SVM became popular because of its success in handwritten digit recognition
- Has been one of the go-to methods in machine learning since the mid-1990s (only recently displaced by deep learning)

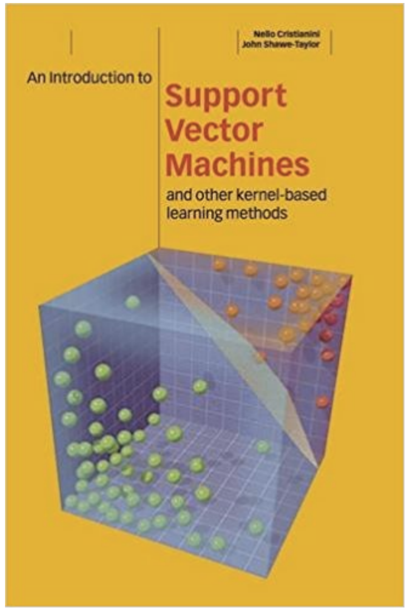
Papers that introduced SVM in its current form

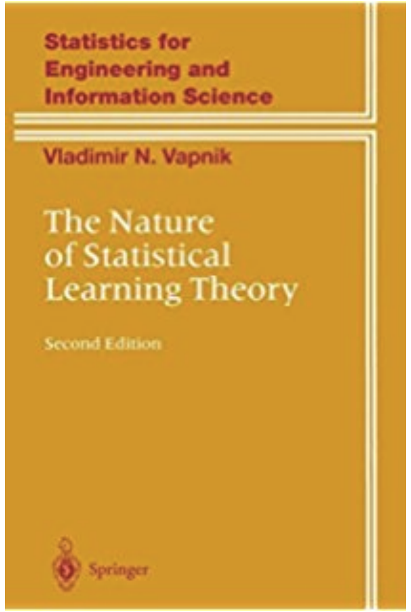

- Boser, B. E.; Guyon, I. M.; Vapnik, V. N. (1992). "A training algorithm for optimal margin classifiers". Proceedings of the fifth annual workshop on Computational learning theory – COLT '92.
- Cortes, C.; Vapnik, V. (1995). "Support-vector networks". Machine Learning. 20 (3): 273–297.

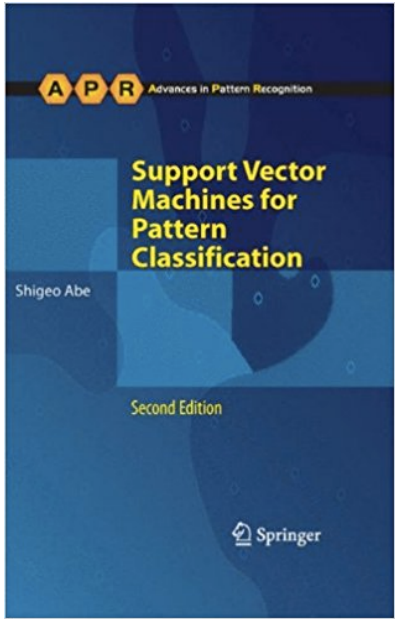
# SVM: Overview and History

- Associated key words
  - Large-margin classifier, Max-margin classifier, Kernel methods

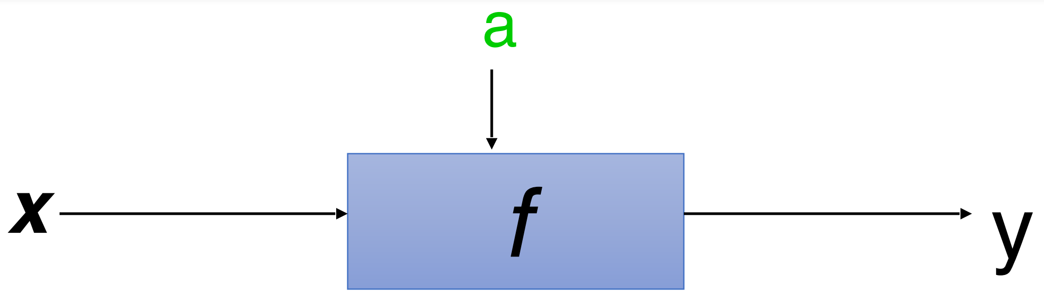


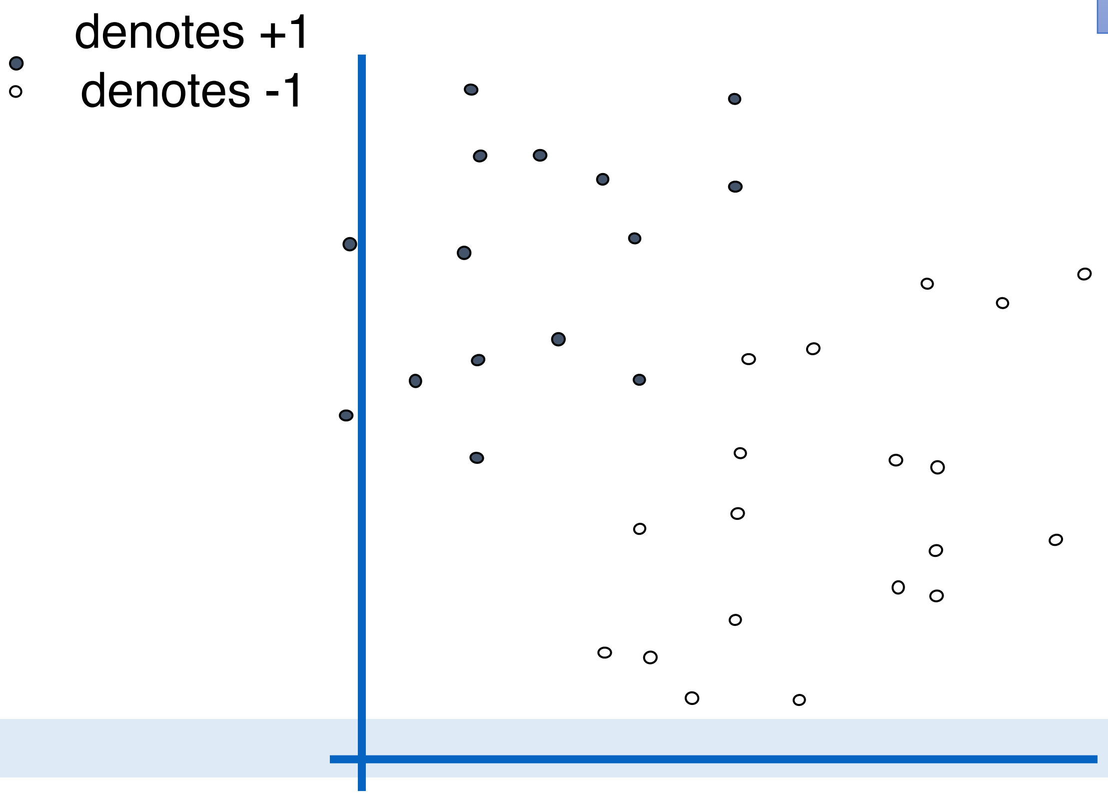



# Linear Classifiers

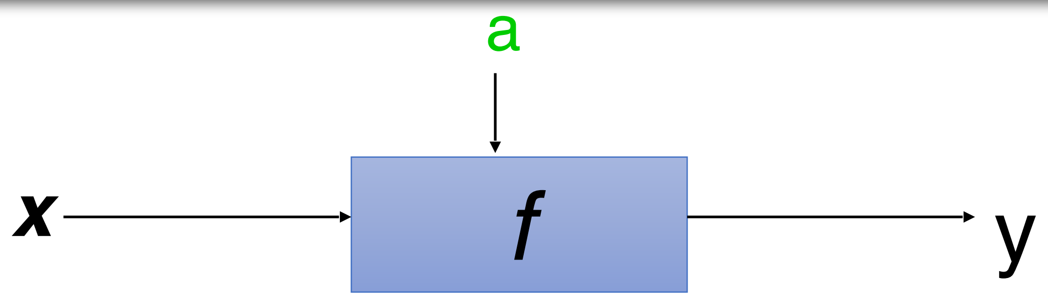



$f(\mathbf{x},\mathbf{w},b) = sign(\mathbf{w}.\ \mathbf{x} + b)$

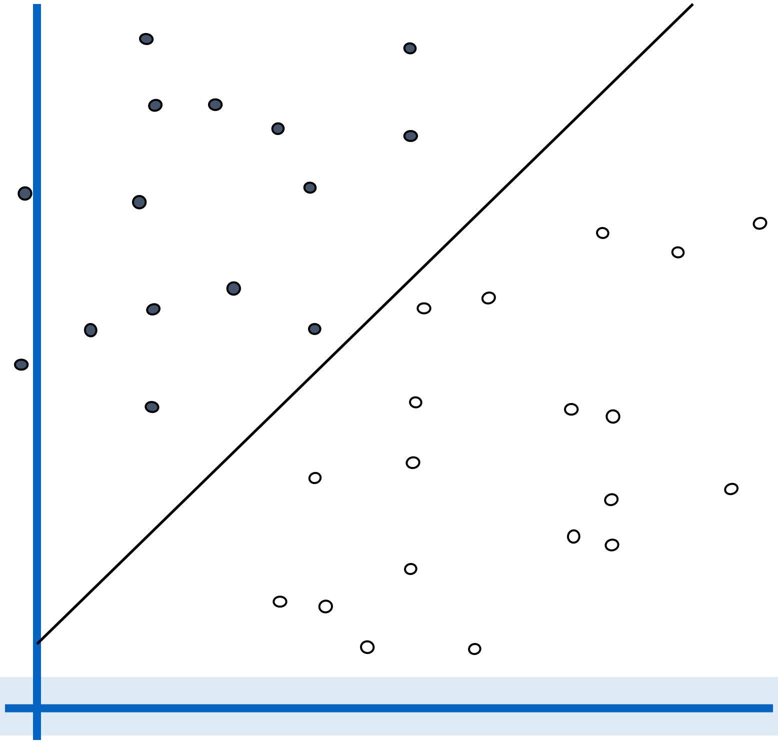



How would you classify this data?

# Linear Classifiers



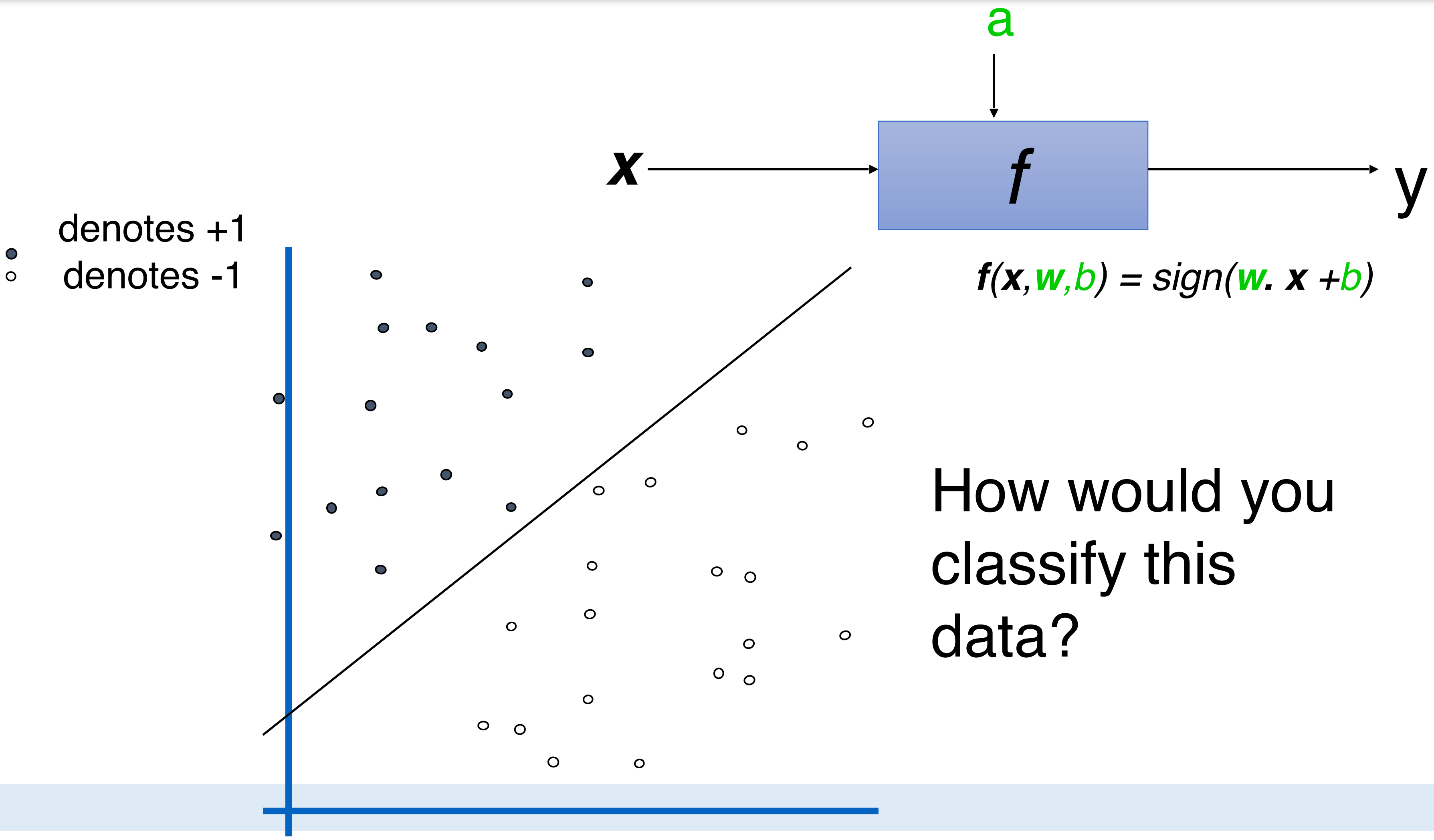

denotes +1
denotes -1



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

How would you classify this data?

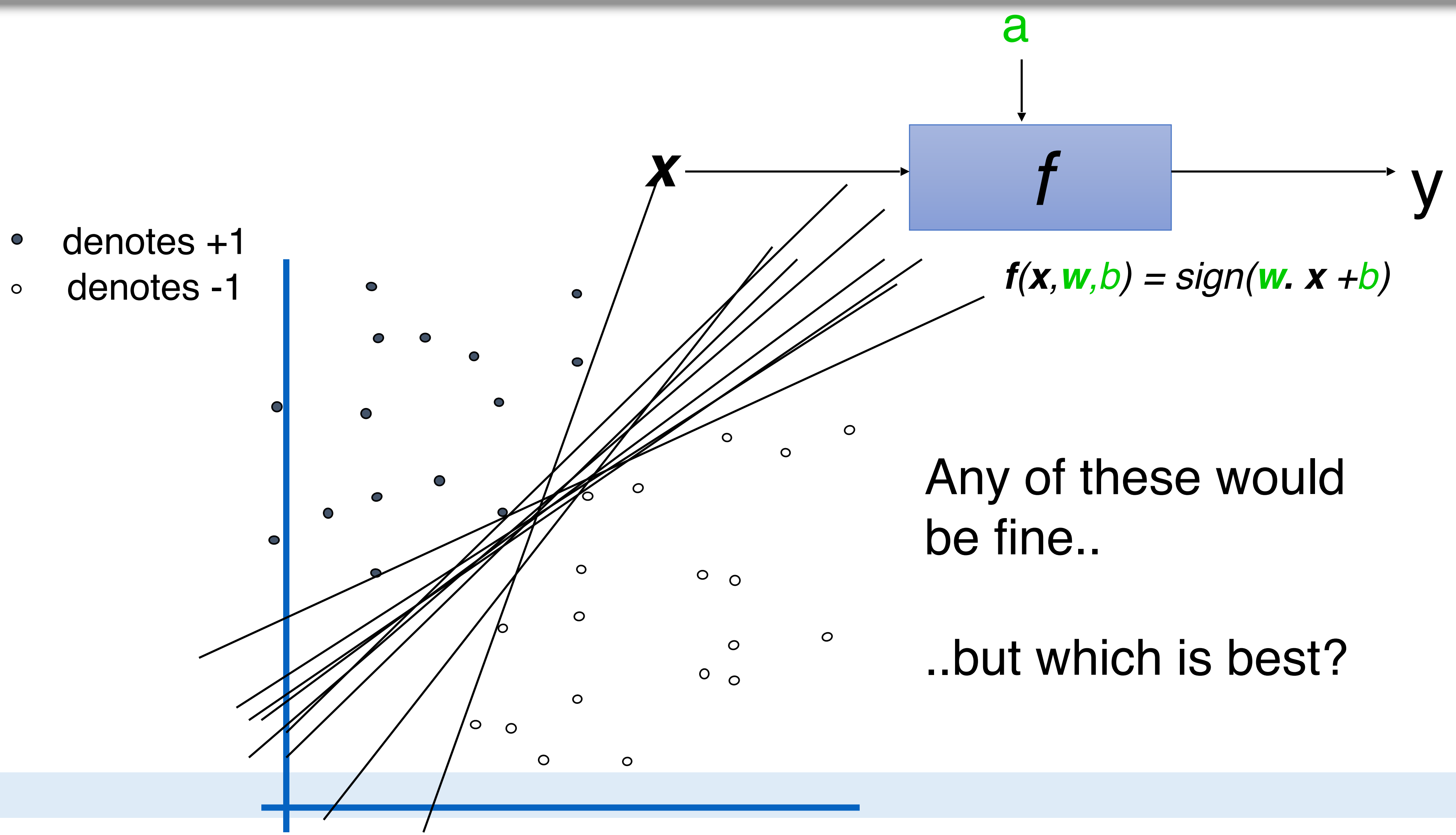
# Linear Classifiers




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

How would you classify this data?

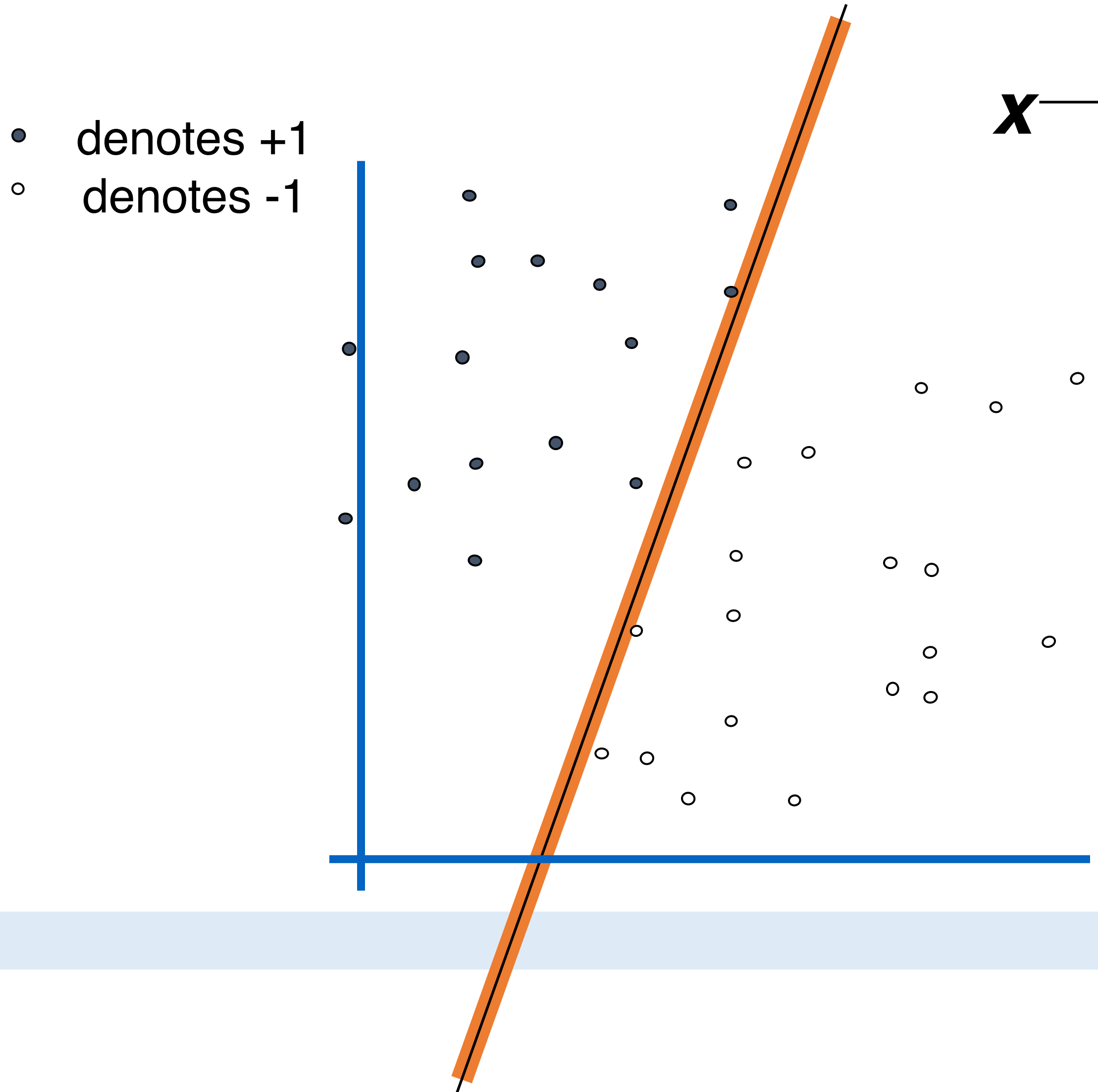
# Linear Classifiers

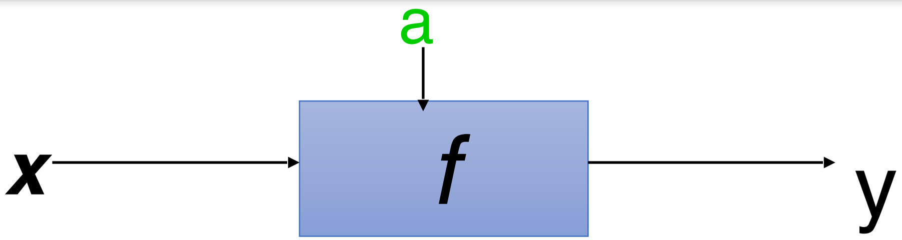



Any of these would be fine..

..but which is best?

# Linear Classifiers

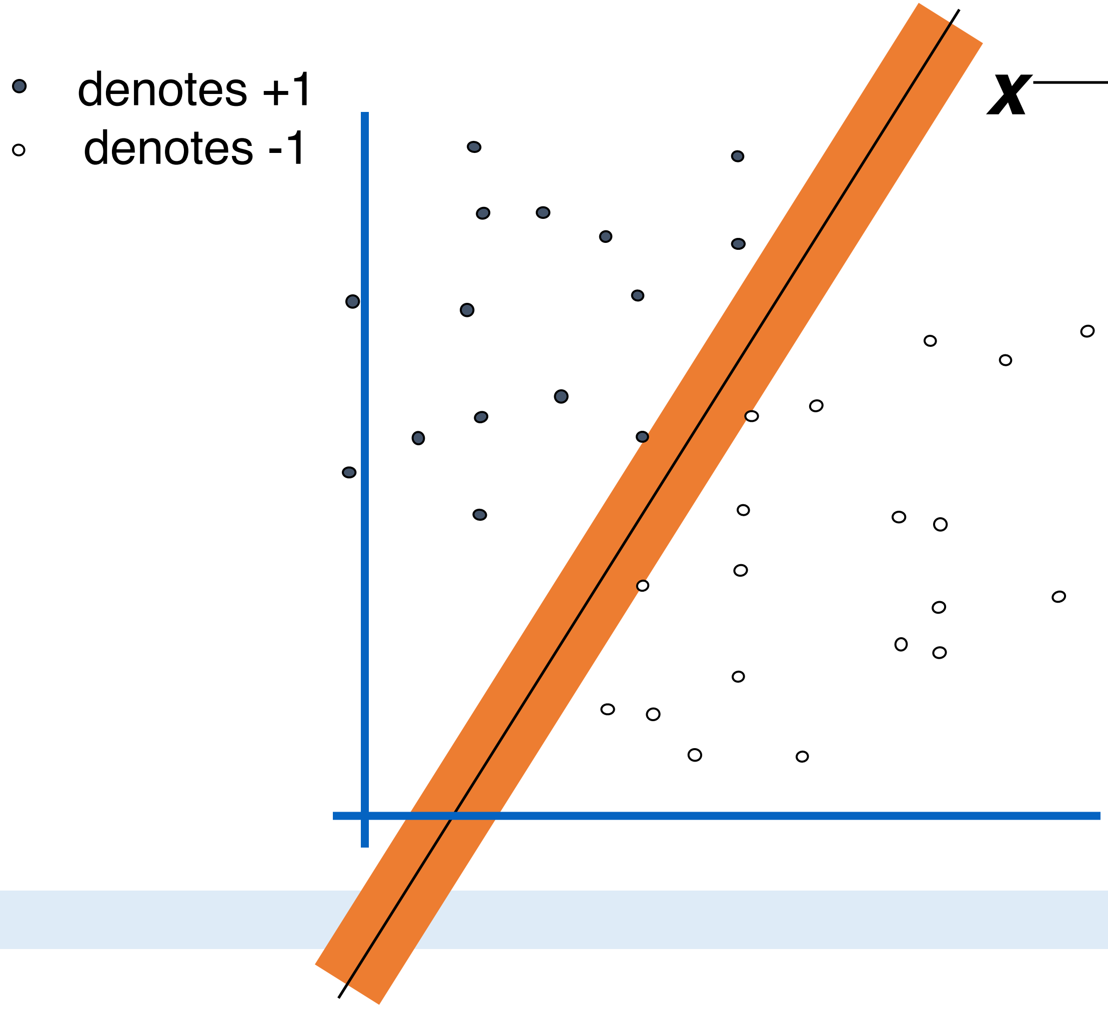


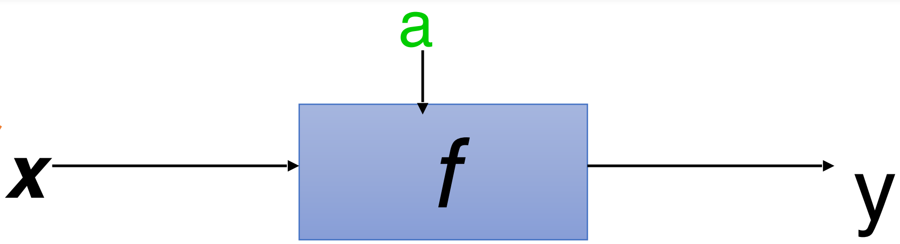




$f(\mathbf{x},w,b) = sign(\mathbf{w}.\ \mathbf{x} + b)$

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

# Linear Classifiers



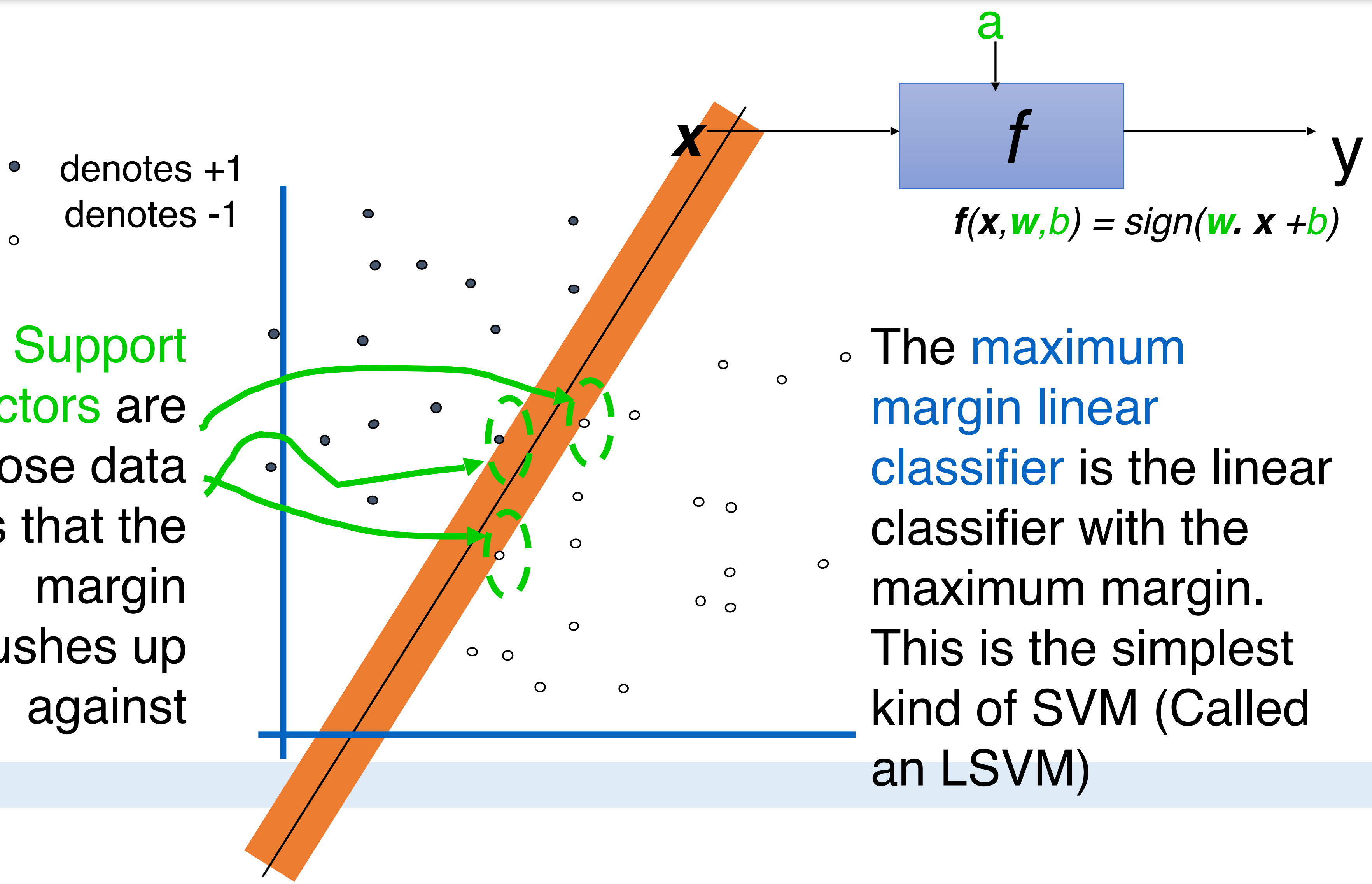




$f(\mathbf{x},\mathbf{w},b) = sign(\mathbf{w}.\ \mathbf{x} + b)$

The maximum margin linear classifier is the linear classifier with the maximum margin.

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

# Maximum Margin Classifier




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

Support Vectors are those data points that the margin pushes up against

The maximum margin linear classifier is the linear classifier with the maximum margin. This is the simplest kind of SVM (Called an LSVM)

# Why Maximum Margin?

a

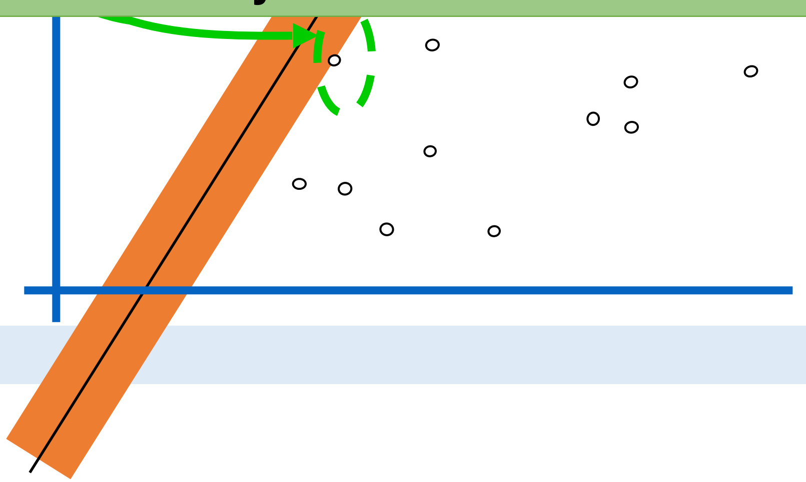
y

b)

- Intuitively this feels safest. If we've made a small error in the location of the boundary this gives us least chance of causing a misclassification.
- The model is immune to removal of any non-support-vector datapoints.
- Empirically it works very well.

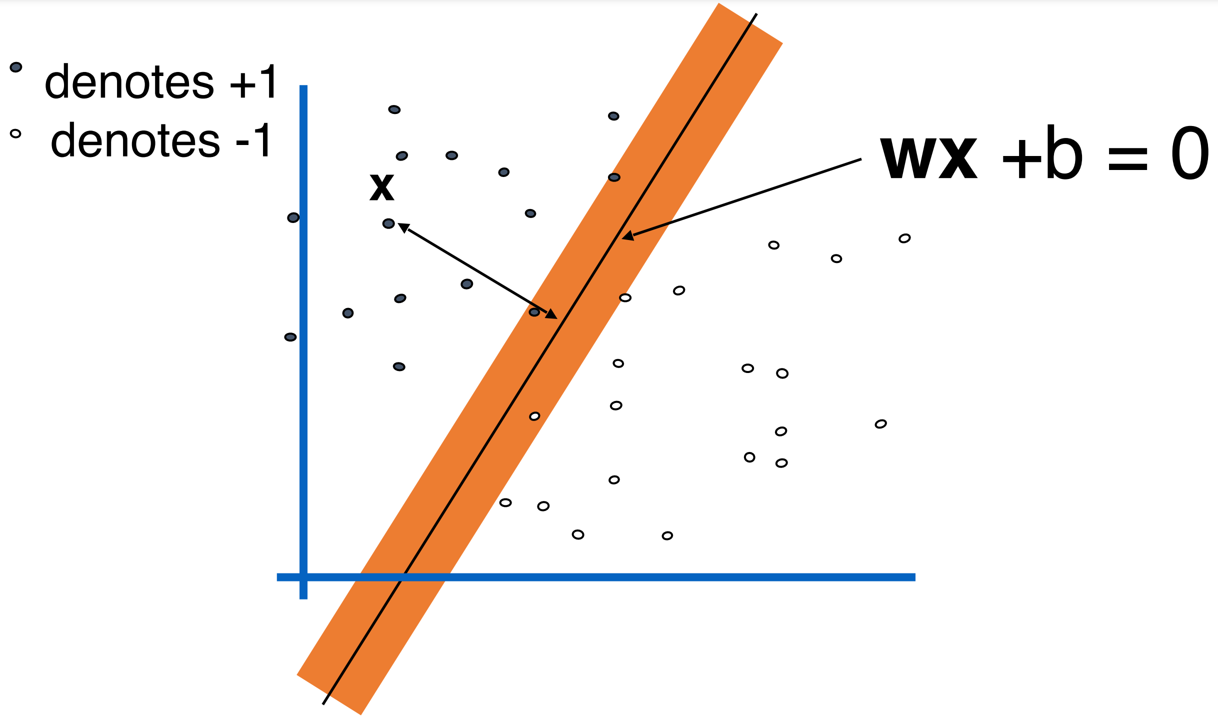
ar

points that the margin pushes up against



classifier with the maximum margin. This is the simplest kind of SVM (Called an LSVM)

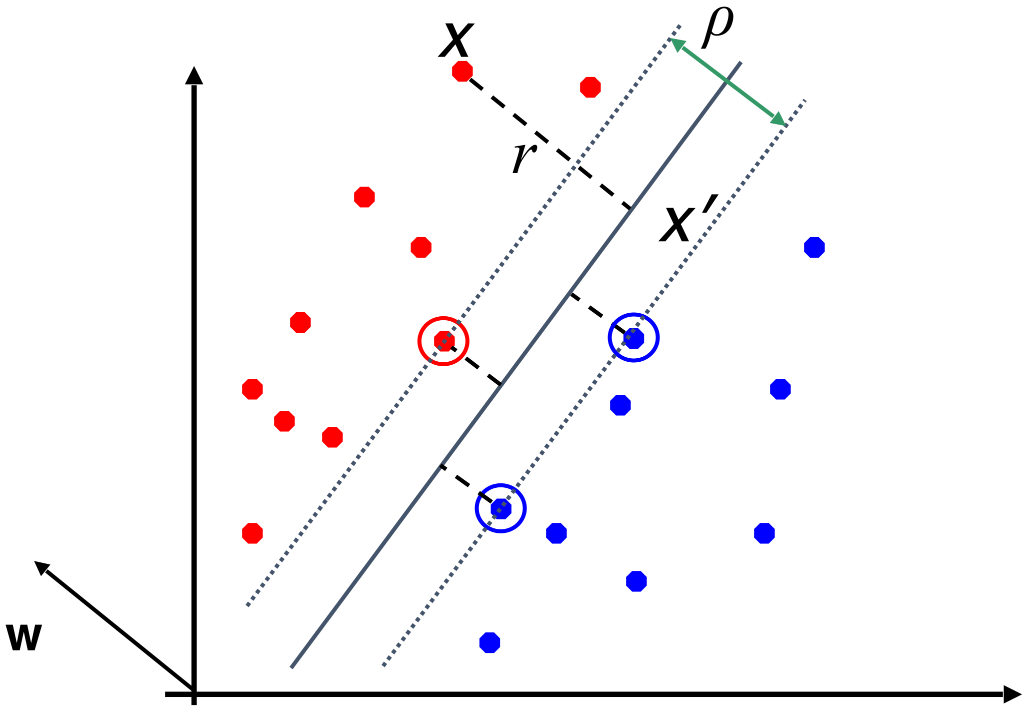
# Estimating the Margin




- What is the distance expression for a point **x** to a line **wx**+b= 0?

# Estimating the Margin

- Distance from example to the separator is $r = y\frac{\mathbf{w}^T\mathbf{x}+b}{\|\mathbf{w}\|}$

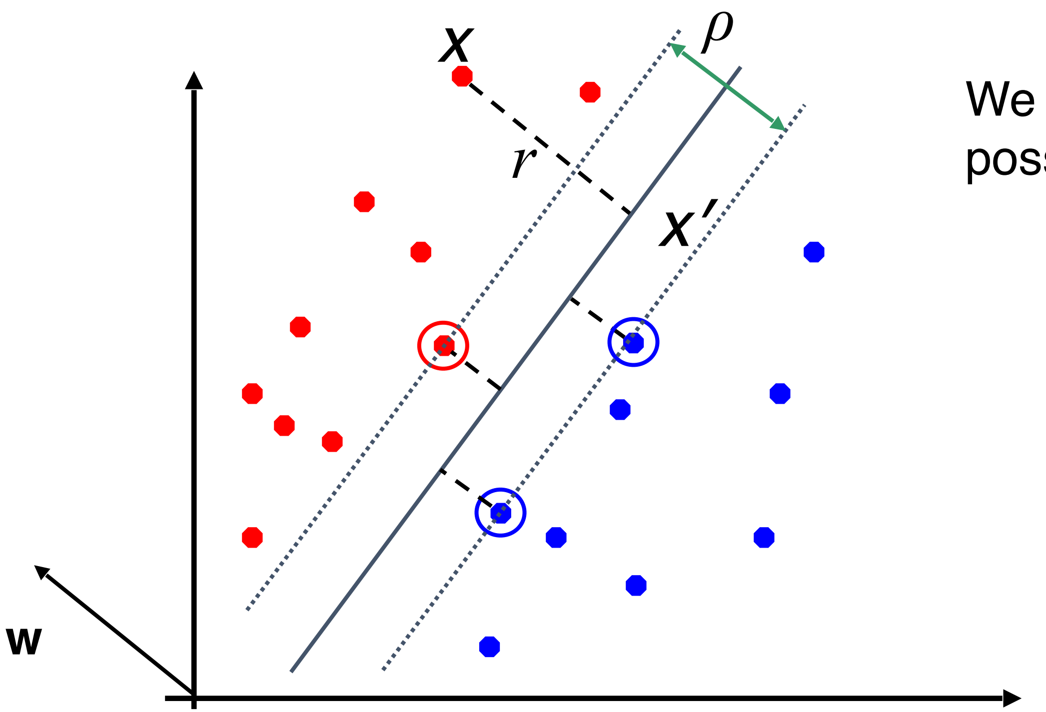



**Derivation of finding *r*:**

- Dotted line **x'**– **x** is perpendicular to decision boundary, so parallel to **w**.
- Unit vector is **w**/ll**w**ll, so line is r**w**/ll**w**ll.
- **x** - **x'** = yr**w**/ll**w**ll.
- **x'** = **x** – yr**w**/ll**w**ll.
- **x'** satisfies $\mathbf{w}^T\mathbf{x}'$+ b = 0.
- So $\mathbf{w}^T$(**x** –yr**w**/ll**w**ll) + b = 0
- Recall that ll**w**ll = sqrt($\mathbf{w}^T\mathbf{w}$).
- So $\mathbf{w}^T\mathbf{x}$ –yrll**w**ll + b = 0
- So, solving for r gives: r = y($\mathbf{w}^T\mathbf{x}$ + b)/ll **w**ll

# Maximum Margin Formulation 1

- Distance from example to the separator is $r = y\frac{\mathbf{w}^T\mathbf{x}+b}{\|\mathbf{w}\|}$



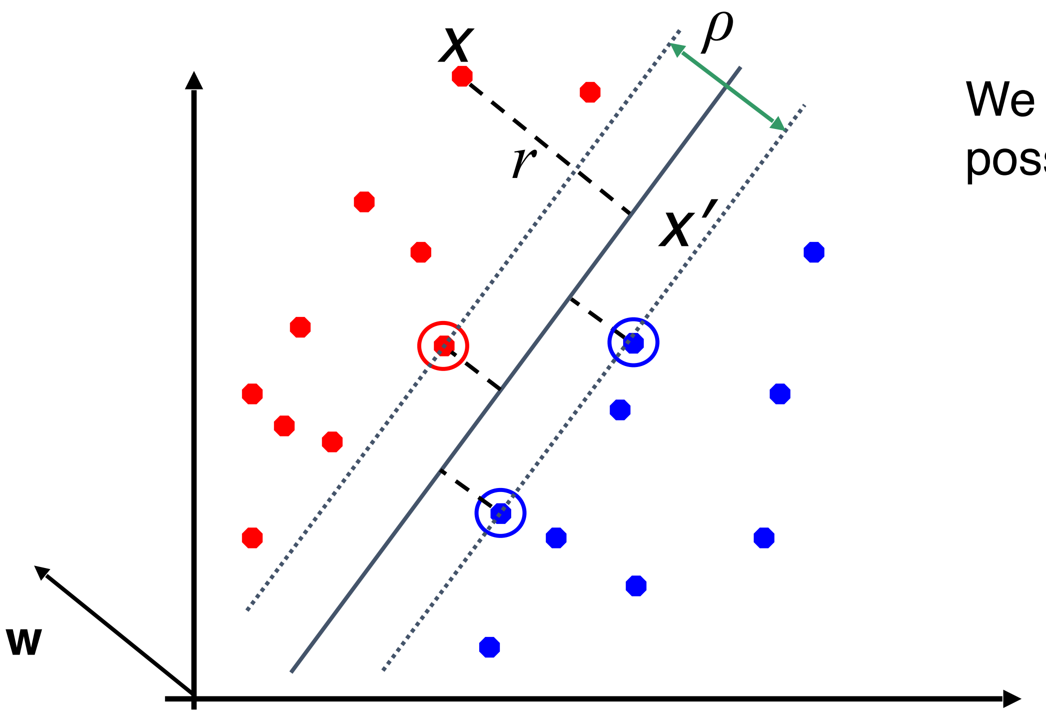

We would like to make the closest distance as large as possible

$$\max_{\mathbf{w},\, b} \min_{i=1}^{N} \frac{y_i(\mathbf{w}^T\mathbf{x}_i + b)}{||\mathbf{w}||}$$

Direct solution of this optimization problem would be very complex

# Maximum Margin Formulation 1

- Distance from example to the separator is $r = y\dfrac{\mathbf{w}^T\mathbf{x}+b}{\|\mathbf{w}\|}$




We would like to make the closest distance as large as possible

$$\max_{\mathbf{w},\, b}\ \min_{i=1}^{N} \frac{y_i(\mathbf{w}^T\mathbf{x}_i + b)}{||\mathbf{w}||}$$

Direct solution of this optimization problem would be very complex

If we make the rescaling $\mathbf{w} \rightarrow \kappa\mathbf{w}$ and $b \rightarrow \kappa b$, then the distance from any point $\mathbf{x}_n$ to the decision surface is unchanged, since the $k$ factor cancels out when we divide by $//\mathbf{w}//$.

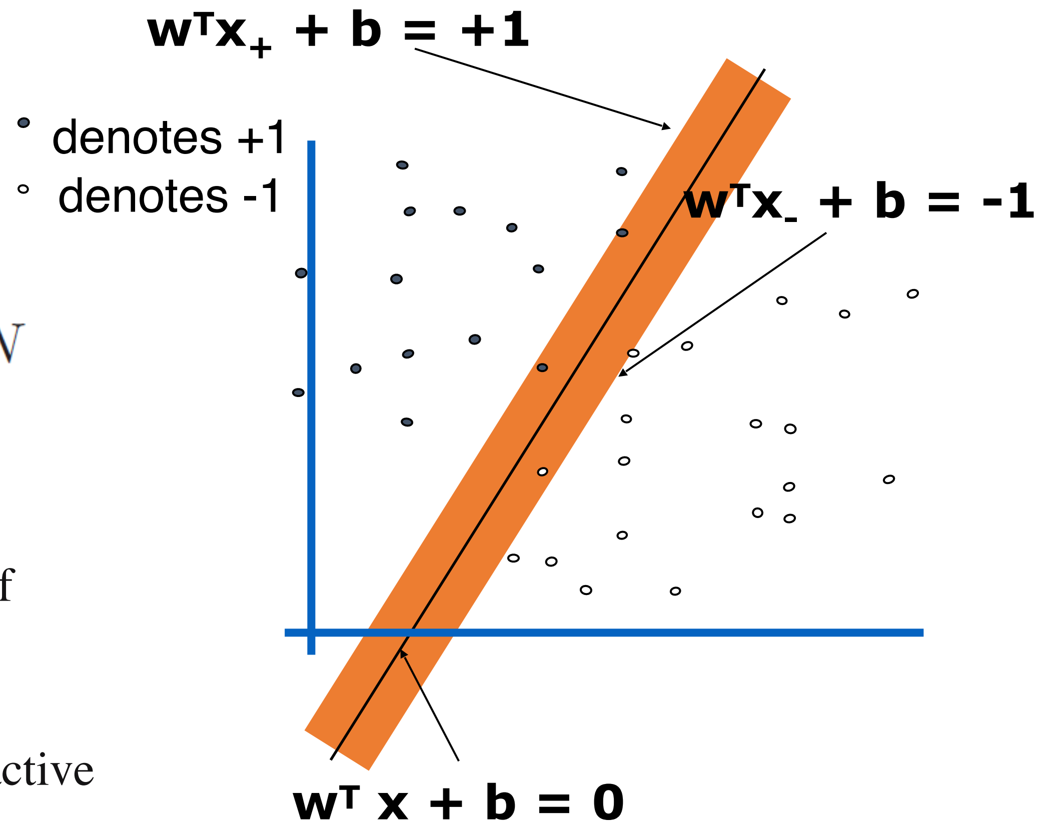
# Maximum Margin Formulation 2

- Let us choose normalization such that $y(\mathbf{w}^T\mathbf{x}_+ + b) = +1$ for the closest point (support vector)

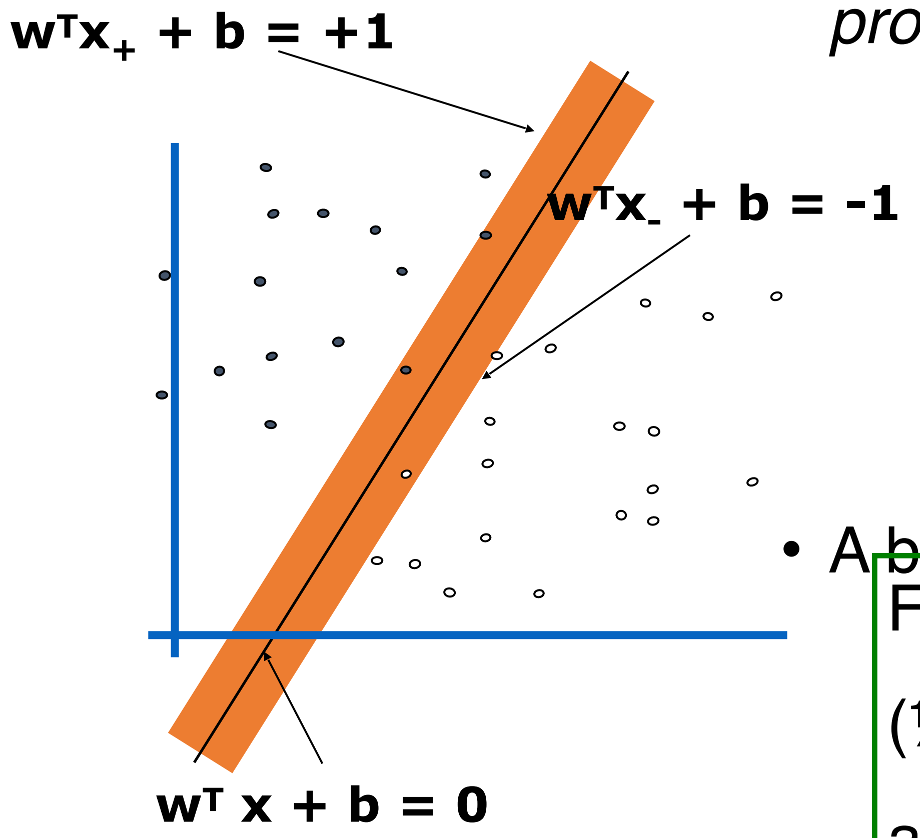
$$\min_{\mathbf{w}, w_0} \frac{1}{2}||\mathbf{w}||^2 \quad \text{s.t.} \quad y_i(\mathbf{w}^T\mathbf{x}_i + b_0) \geq 1, i = 1 : N$$

example of a *quadratic programming* problem in which we are trying to minimize a quadratic function subject to a set of linear inequality constraints.

once the margin has been maximized there will be at least two active constraints



# Maximizing the Margin




- Then we can formulate the *quadratic optimization problem:*

Find $\mathbf{w}$ and $b$ such that

$\rho = \dfrac{2}{\|\mathbf{w}\|}$ is maximized; and for all $\{(\mathbf{x}_i, y_i)\}$

$\mathbf{w}^T\mathbf{x}_i + b \geq 1$ if $y_i = +1$; $\mathbf{w}^T\mathbf{x}_i + b \leq -1$ if $y_i = -1$

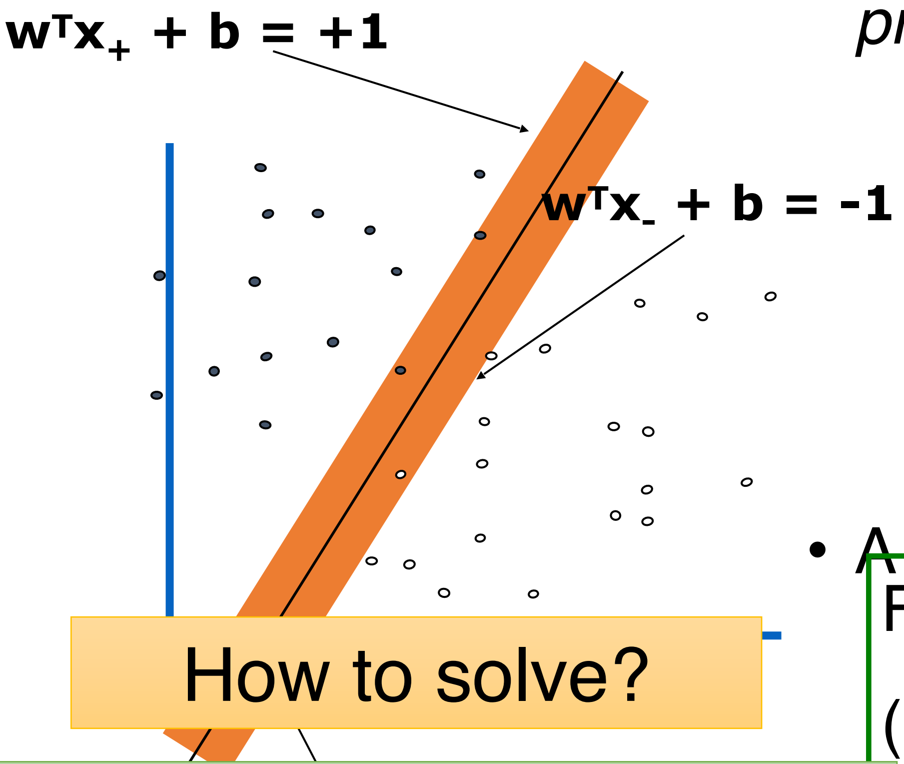
- A better formulation (min $\|\mathbf{w}\|$ = max 1/ $\|\mathbf{w}\|$ ):

Find $\mathbf{w}$ and $b$ such that

$(½\ \mathbf{w}^T\mathbf{w})$ is minimized

and for all $\{(\mathbf{x}_i, y_i)\}$: $y_i(\mathbf{w}^T\mathbf{x}_i + b) \geq 1$

# Hard Margin Linear support vector machines




- Then we can formulate the *quadratic optimization problem:*

Find **w** and *b* such that

$\rho = \dfrac{2}{\|\mathbf{w}\|}$ is maximized; and for all $\{(\mathbf{x_i}, y_i)\}$

$\mathbf{w}^T\mathbf{x_i} + b \geq 1$ if $y_i = +1$; $\mathbf{w}^T\mathbf{x_i} + b \leq -1$ if $y_i = -1$

- A better formulation (min **||w||** = max 1/ **||w||** ):

Find **w** and *b* such that

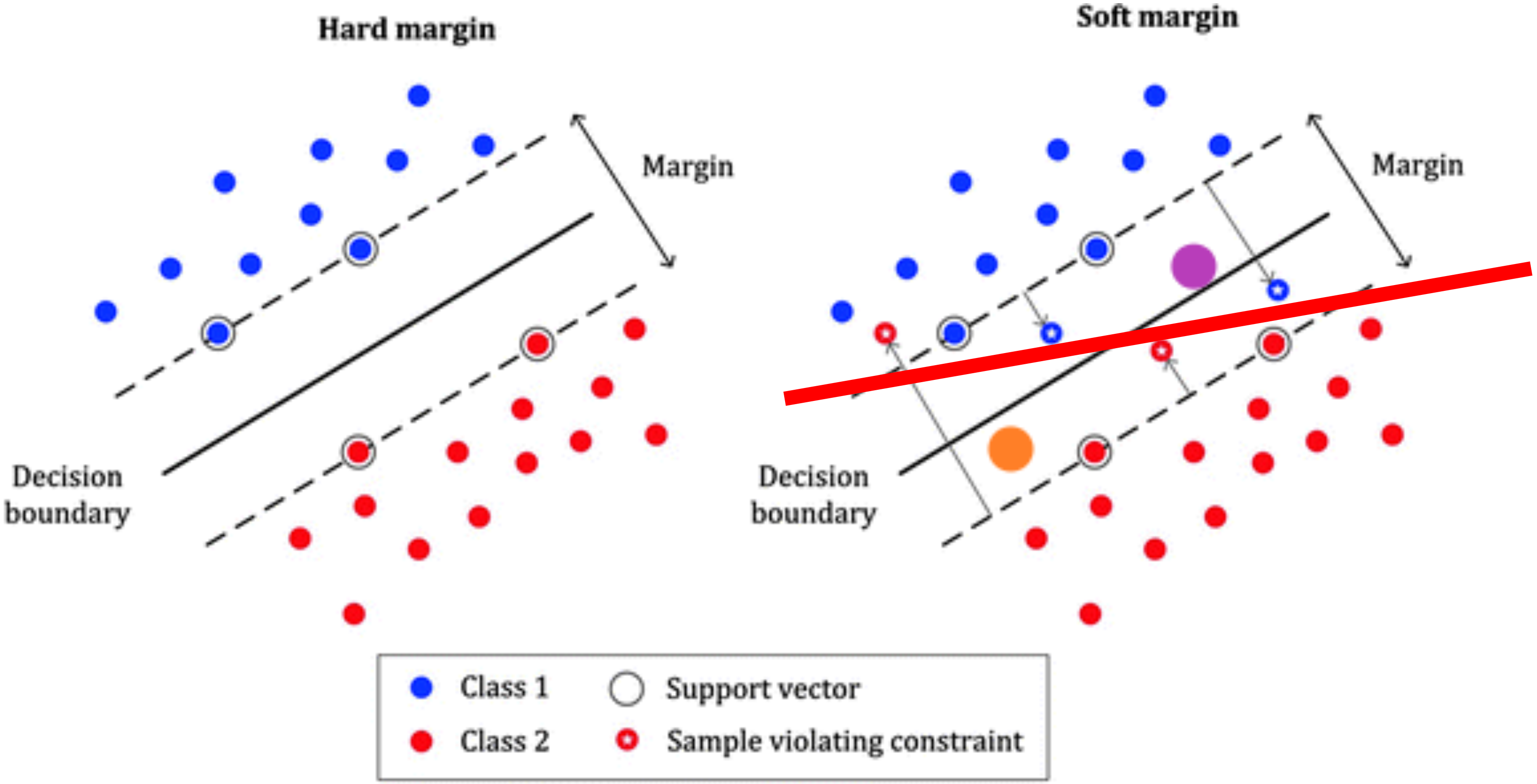
$(\frac{1}{2}\, \mathbf{w}^T\mathbf{w})$ is minimized

and for all $\{(\mathbf{x_i}, y_i)\}$: $y_i\, (\mathbf{w}^T\mathbf{x_i} + b) \geq 1$

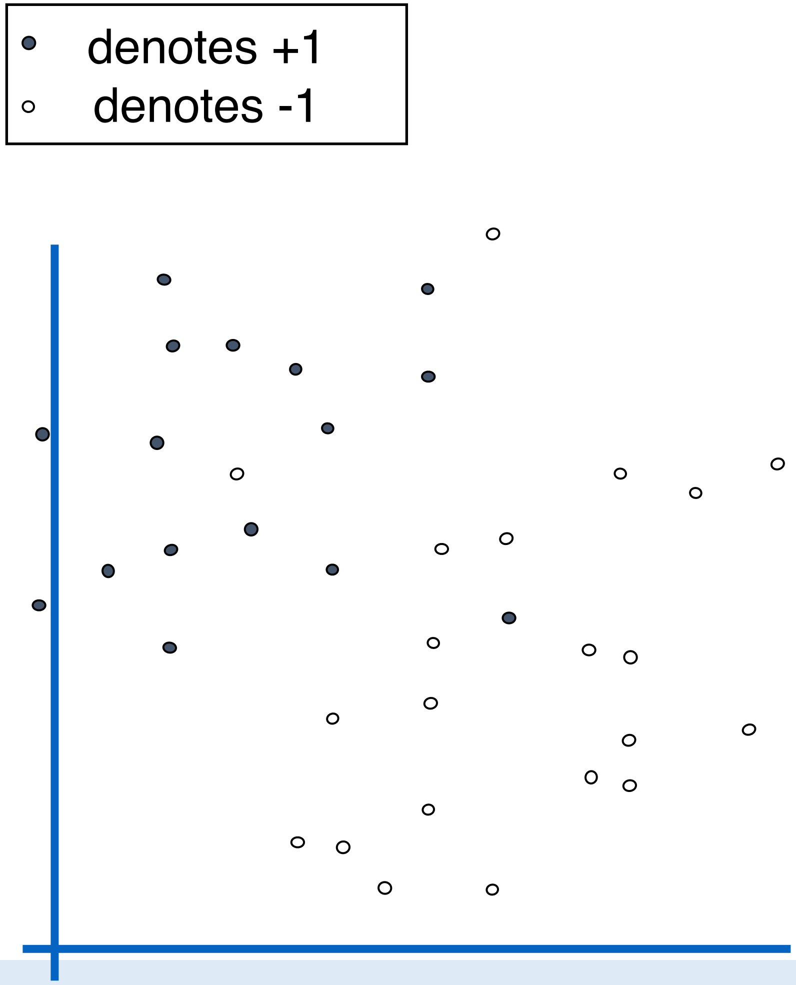
How to solve?

Quadratic Programming

# Hard Margin Linear support vector machines



# Non-separable Data




This is going to be a problem!
What should we do?

Previous formulation implicitly used an error function (through constraints) that gave infinite error if a data point was misclassified and zero error if it was classified correctly

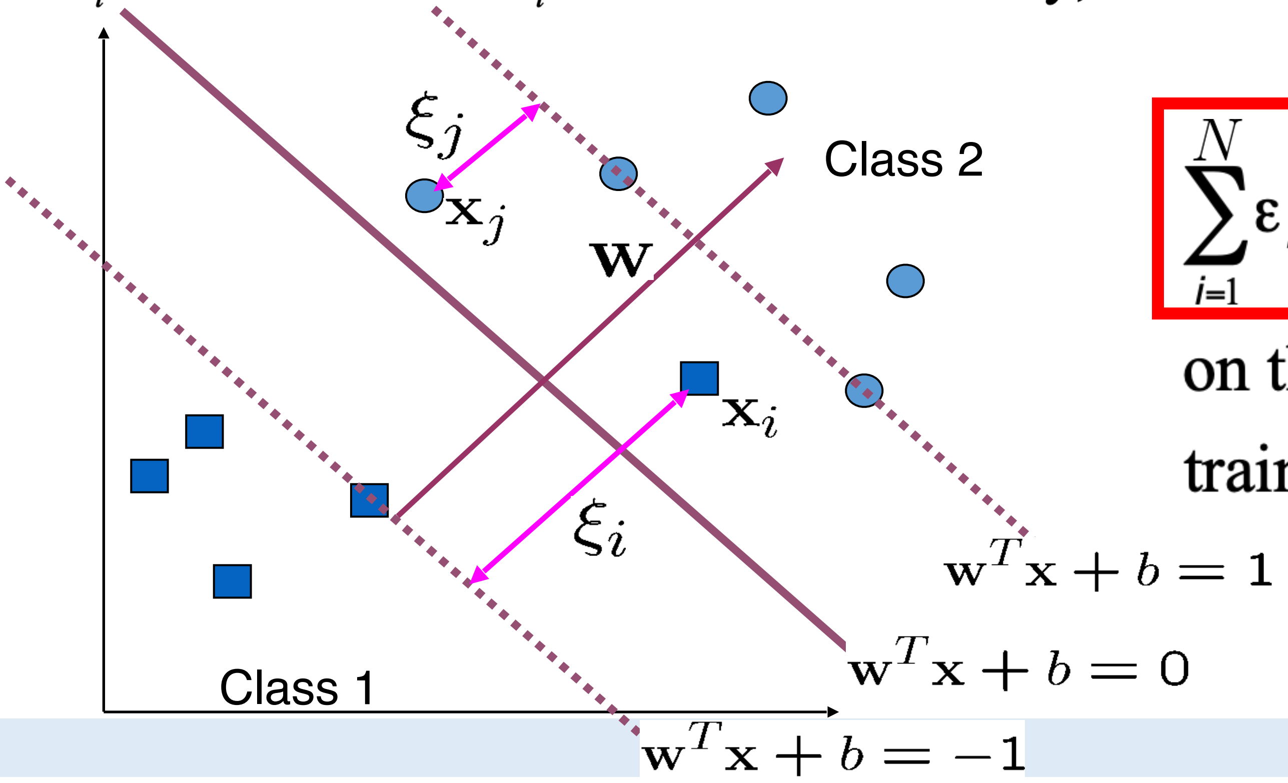
data points are allowed to be on the 'wrong side' of the margin boundary, but with a penalty that increases with the distance from that boundary

# SVM for Noisy Data

$\varepsilon_i \ge 1 \quad \Leftrightarrow \quad y_i(wx_i + b) < 0,$ i.e., misclassification

slack variable $0 \le \epsilon_i \le 1 \quad \Leftrightarrow \quad x_i$ is correctly classified, but lies inside the margin

$\varepsilon_i = 0 \quad \Leftrightarrow \quad x_i$ is classified correctly, and lies outside the margin




$\boxed{\sum_{i=1}^{N} \varepsilon_i}$ is an upper bound on the number of training errors.

# Soft-Margin SVM : SVM for Noisy Data

Balance the trade off between margin and classification errors

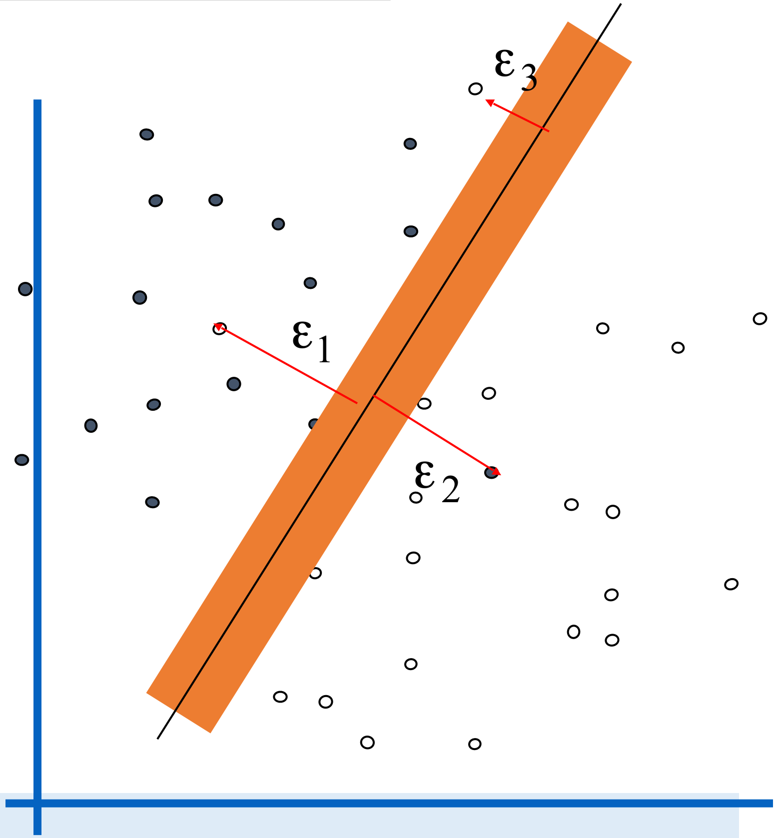
$$\min_{w,b,\xi} \quad \frac{1}{2}||w||^2 + C\sum_n \xi_n$$

$$\text{subj. to} \quad y_n\,(w \cdot x_n + b) \geq 1 - \xi_n \qquad (\forall n)$$

$$\xi_n \geq 0 \qquad (\forall n)$$

Parameter $C > 0$ controls the trade-off between the slack variable penalty and the margin.

● denotes +1
○ denotes -1



# SVM for Noisy Data

- Use the Lagrangian formulation for the optimization problem.
- Introduce a positive Lagrangian multiplier for each inequality constraint.

$$y_i(x_i \bullet w + b) - 1 + \varepsilon_i \geq 0, \text{ for all i.}$$

$$\alpha_i$$

$$\varepsilon_i \geq 0, \text{ for all i.}$$

Lagrangian multipliers

$$\beta_i$$

Get the following Lagrangian: $L_p = \|w\|^2 + c\sum_i \varepsilon_i - \sum_i \alpha_i \{y_i(x_i \bullet w + b) - 1 + \varepsilon_i\} - \sum_i \beta_i \varepsilon_i$

# SVM for Noisy Data

$$L_p = \|w\|^2 + c\sum_i \varepsilon_i - \sum_i \alpha_i \{y_i(x_i \bullet w + b) - 1 + \varepsilon_i\} - \sum_i \beta_i \varepsilon_i$$

$$\frac{\partial L_p}{\partial w} = 2w - \sum_i \alpha_i y_i x_i = 0 \quad \Rightarrow \quad w = \frac{1}{2}\sum_i \alpha_i y_i x_i$$

$$\frac{\partial L_p}{\partial b} = -\frac{1}{2}\sum_i \alpha_i y_i = 0 \quad \Rightarrow \quad \sum_i \alpha_i y_i = 0$$

$$\frac{\partial L_p}{\partial \varepsilon_i} = c - \beta_i - \alpha_i = 0 \quad \Rightarrow \quad c = \beta_i + \alpha_i$$

Take the derivatives of $L_p$ with respect to w, b, and $\varepsilon_i$.

Karush-Kuhn-Tucker Conditions

$$0 \le \alpha_i \le c \quad \forall i$$

$$L_D = \sum_i \alpha_i - \frac{1}{2}\sum_{i,j} \alpha_i \alpha_j y_i y_j (x_i \bullet x_j)$$

Both $\varepsilon_i$ and its multiplier $\beta_i$ are not involved in the function.

# SVM Lagrangian Dual

Maximize $\sum_{k=1}^{R} \alpha_k - \frac{1}{2} \sum_{k=1}^{R} \sum_{l=1}^{R} \alpha_k \alpha_l Q_{kl}$ where $Q_{kl} = y_k y_l (\mathbf{x}_k \cdot \mathbf{x}_l)$

subject to constraints: $0 \le \alpha_k \le c \quad \forall k$ $\qquad \sum_{k=1}^{R} \alpha_k y_k = 0$

Once solved, we obtain w and b using:

$$\mathbf{w} = \frac{1}{2} \sum_{k=1}^{R} \alpha_k y_k \mathbf{x}_k$$

$$y_i(x_i \bullet w + b) - 1 = 0$$

$$b = -y_i(y_i(x_i \bullet w) - 1)$$

Then classify with:

***f*(*x*,*w*,*b*) = *sign*(*w*. *x* + *b*)**

आई आई टी हैदराबाद
IIT Hyderabad

# Back to SVM

**SVM standard (primal) form:**

Minimize: $\frac{1}{2}\|\vec{w}\|^2$
*(w,b)*

Such that: $y_i(\vec{w}\cdot\vec{x}_i - b) \geq 1$
*(for all i)*

*Maximize $\gamma = 2/\|w\|$*

**SVM standard (dual) form:**

Maximize: $\sum_{i=1}^{n}\alpha_i - \frac{1}{2}\sum_{i,j}\alpha_i\alpha_j y_i y_j (x_i \cdot x_j)$
*($\alpha_i$)*

Such that: $\sum_{i=1}^{n}\alpha_i y_i = 0 \qquad \alpha_i \geq 0$
*(for all i)*

Both yield the same solution

*Only a function of "support vectors"*

आई आई टी हैदराबाद
IIT Hyderabad

# SVM Lagrangian Dual

Maximiz
e $\sum_{k=1}^{R} \alpha_k - \frac{1}{2} \sum_{k=1}^{R} \sum_{l=1}^{R} \alpha_k \alpha_l Q_{kl}$ where $Q_{kl} = y_k y_l (\mathbf{x}_k \cdot \mathbf{x}_l)$

subject to constraints: $0 \le \alpha_k \le c \quad \forall k \qquad \sum_{k=1}^{R} \alpha_k y_k = 0$

Datapoints with $\alpha_k > 0$ will be the support vectors

Once solved, we obtain w and b

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

$$\frac{1}{2} \sum_{k=1}^{R} \alpha_k y_k \mathbf{x}_k$$

$$y_i (x_i \bullet w + b) - 1 = 0$$

$$b = -y_i (y_i (x_i \bullet w) - 1)$$

If $a_n < C$, $\xi n = 0$ and hence such points lie on the margin

Then classify with:

***f*(*x*,*w*,*b*) = *sign*(*w*. *x* + *b*)**

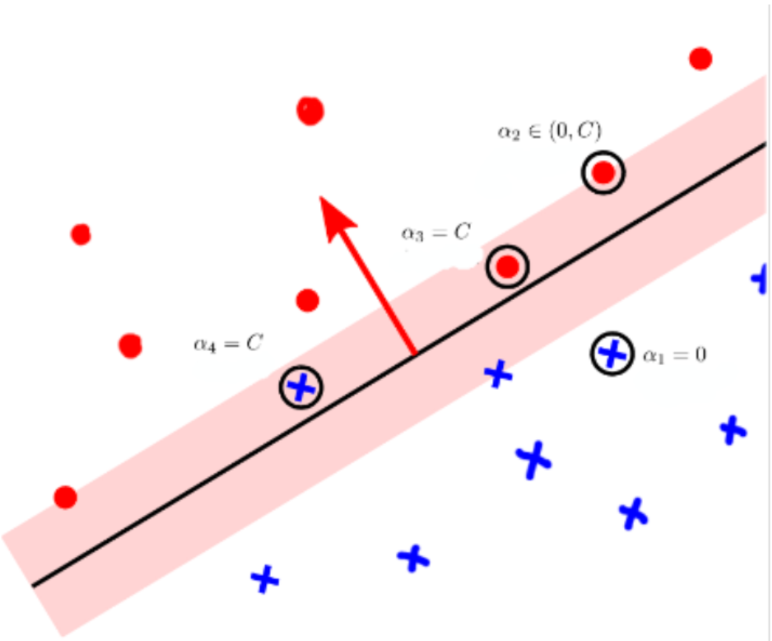
# SVMs and Dual variables

There are 3 kinds of data vectors $\mathbf{x}_n$.

1. Non-support vectors. Examples that lie on the correct side outside the margin, so $\alpha_n = 0$.

2. Essential support vectors. Examples that lie just on the margin, therefore $\alpha_n \in (0, C)$.

3. Bound support vectors. Examples that lie strictly inside the margin, or on the wrong side, therefore $\alpha_n = C$

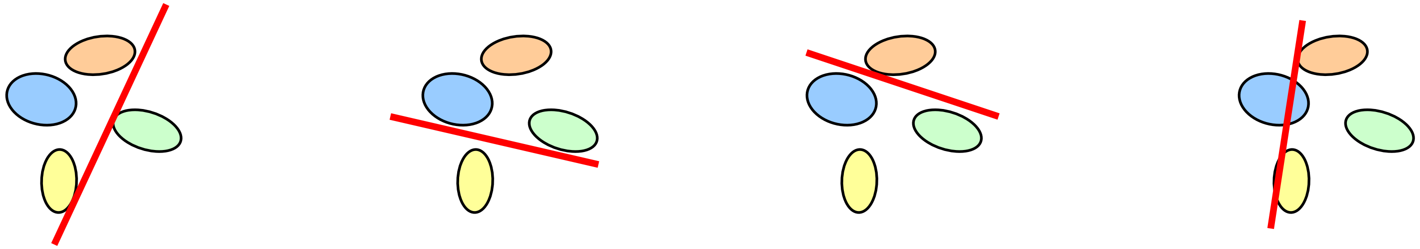


# Multi-class Classification with SVMs

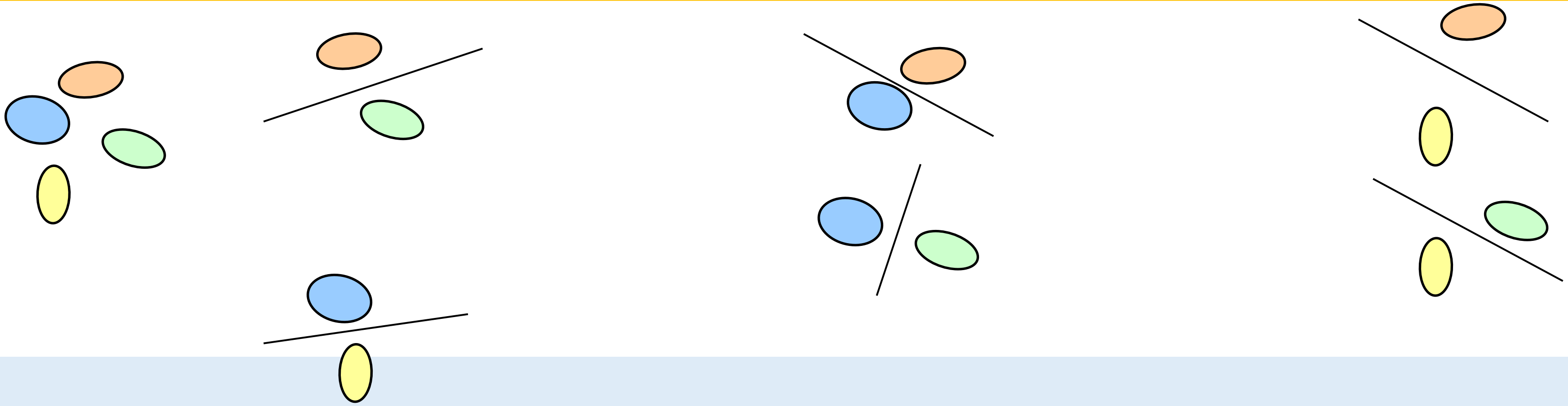
- SVMs can only handle two-class outputs.
- What can be done?
- Answer: with output arity N, learn N SVM's
  - SVM 1 learns "Output==1" vs "Output != 1"
  - SVM 2 learns "Output==2" vs "Output != 2"
  - :
  - SVM N learns "Output==N" vs "Output != N"
- Then to predict the output for a new input, just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

# Multi-class Classification using SVM

## One- versus-all



## One- versus-one



Soft-margin SVM objective:

$$\begin{aligned} \min_{\mathbf{w},b,\xi} \frac{1}{2}\|\mathbf{w}\|_2^2 + \gamma \sum_{i=1}^{N} \xi_i & \\ \text{s.t.} \quad t^{(i)}(\mathbf{w}^\top \mathbf{x}^{(i)} + b) \geq 1 - \xi_i & \qquad i = 1, \ldots, N \\ \xi_i \geq 0 & \qquad i = 1, \ldots, N \end{aligned}$$

$$\xi_i = \max\{0, 1 - t^{(i)}(\mathbf{w}^\top \mathbf{x}^{(i)} + b)\}.$$

$$\sum_{i=1}^{N} \xi_i = \sum_{i=1}^{N} \max\{0, 1 - t^{(i)}(\mathbf{w}^\top \mathbf{x}^{(i)} + b)\}.$$

write $\max\{0, y\} = (y)_+$

# Soft Margin SVMs and Hinge Loss

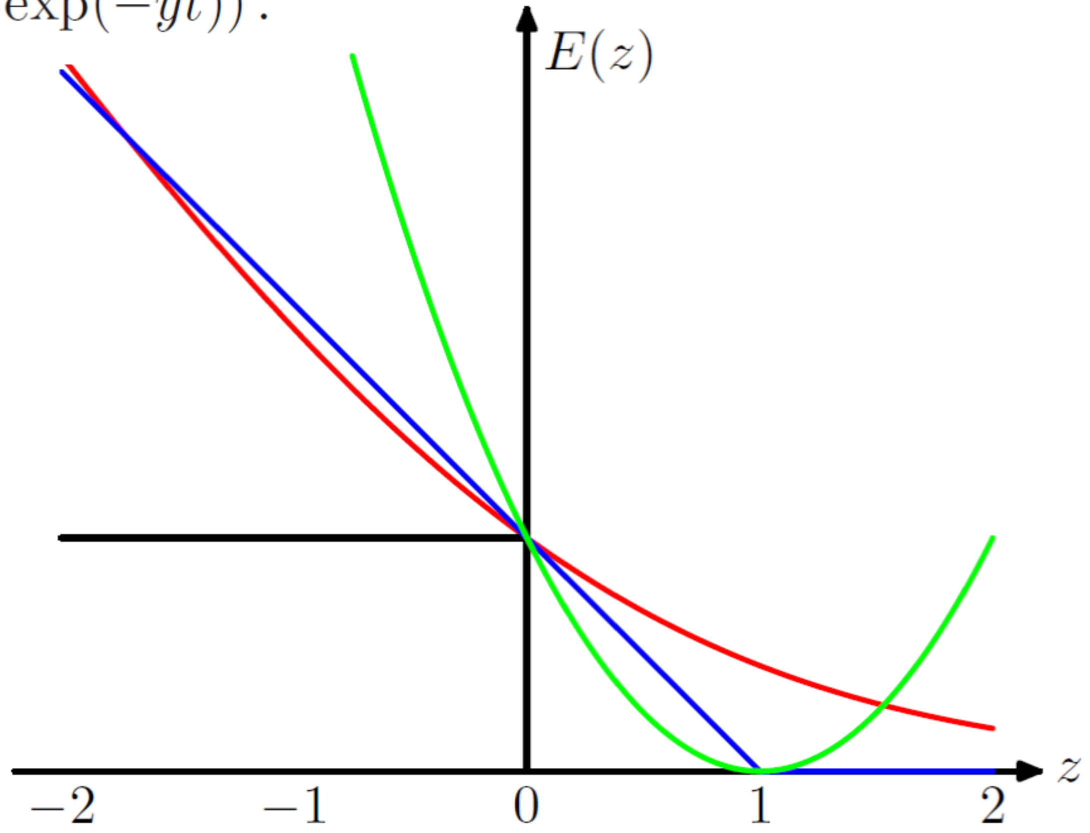
If we write $y^{(i)}(\mathbf{w}, b) = \mathbf{w}^\top \mathbf{x} + b$, then the optimization problem can be written as

$$\min_{\mathbf{w},b,\xi} \sum_{i=1}^{N} \left(1 - t^{(i)} y^{(i)}(\mathbf{w}, b)\right)_+ + \frac{1}{2\gamma} \|\mathbf{w}\|_2^2$$

- The loss function $\mathcal{L}_{\mathrm{H}}(y, t) = (1 - ty)_+$ is called the hinge loss.
- The second term is the $L_2$-norm of the weights.
- Hence, the soft-margin SVM can be seen as a linear classifier with hinge loss and an $L_2$ regularizer.

# Hinge Loss vs other losses

- Blue : hinge loss $E_{\mathrm{SV}}(y_n t_n) = [1 - y_n t_n]_+$
- Red : logistic loss $E_{\mathrm{LR}}(yt) = \ln(1 + \exp(-yt))\,.$
- Green :  squared error
- Black : 0/1 loss



# Readings

- PRML, Bishop, Chapter 7 (7.1-7.3)
- "Introduction to Machine Learning" by Ethem Alpaydin, 2nd edition, Chapters 3 (3.1-3.4), Chapter 13 (13.1-13.9)
- Do read these!
  - https://www.svm-tutorial.com/2017/02/svms-overview-support-vector-machines/
  - https://www.svm-tutorial.com/2016/09/duality-lagrange-multipliers/
  - https://www.svm-tutorial.com/2017/10/support-vector-machines-succinctly-released/