Support Vector Machines

A portion (1/3) of
the slides are taken from
Prof. Andrew Moore's
SVM tutorial at
http://www.cs.cmu.edu/~awm/tutorials
And
Mingyue Tan of UBC

Rao Vemuri
rvemuri@gmail.com

Overview

- Module 1: Why SVM?
- Module 2. Introduction to Vectors
- Module 3: Review of Linear Algebra
- Mod 4: Classifiers & Classifier Margin
- Mod 5: Linear SVMs: Optimization Problem
- Mod 6: Hard Vs Soft Margin Classification
- Mod 7: Nonlinear SVMs
- Mod 8. Applications
    - Gene Data Classification/Text Categorization

Module 1: Why SVM?

- Competitive with other classification methods
- Relatively easy to learn
- Kernel methods give an opportunity to extend the idea to
  - Regression
  - Density estimation
  - Kernel PCA
  - Etc.

Advantages of SVMs - 1

- A principled approach to classification, regression and novelty detection
- Good generalization capabilities
- Hypothesis has an explicit dependence on data, via support vectors – hence, can readily interpret model

Advantages of SVMs - 2

- Learning involves optimization of a convex function (no local minima as in neural nets)
- Only a few parameters are required to tune the learning machine (unlike lots of weights and learning parameters, hidden layers, hidden units, etc as in neural nets)

Module 2: Vectors & Vector Algebra

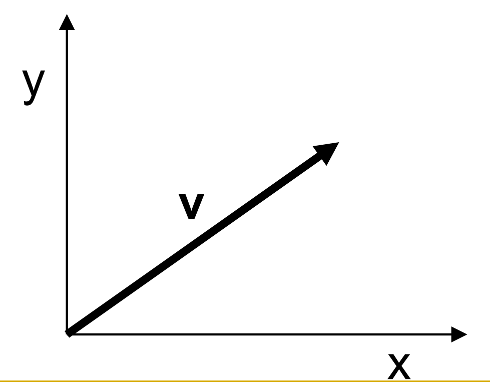
- Vectors, matrices, dot products
- Eqn of a straight line in vector notation
- Familiarity with
  - Perceptron is useful
  - Mathematical programming will be useful
  - Vector spaces will be an added benefit
- The more comfortable you are with Linear Algebra, the easier this material will be

What is a Vector ?

- Think of a vector as a *directed line segment in N-dimensions*! (has "length" and "direction")

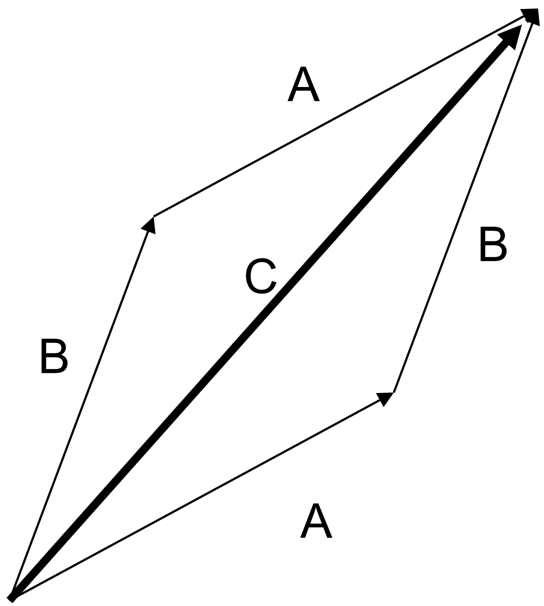
$$\vec{v} = \begin{bmatrix} a \\ b \\ c \end{bmatrix}$$

- Basic idea: convert geometry in higher dimensions into algebra!
  - Once you define a "nice" *basis* along each dimension: x-, y-, z-axis …
    - Vector becomes a 1 x N matrix!
    - **v** = [a b c]^T
    - Geometry starts to become linear algebra on vectors like **v**!



Vector Addition: A+B

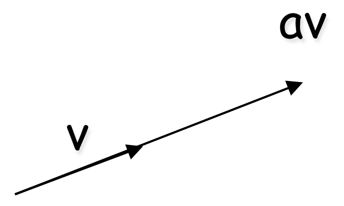
$$\mathbf{A+B} = (x_1, x_2) + (y_1, y_2) = (x_1 + y_1, x_2 + y_2)$$



A+B = C
(use the head-to-tail method to combine vectors)

Scalar Product: *a*v

$$a\mathbf{v} = a(x_1, x_2) = (ax_1, ax_2)$$



Change only the length (“scaling”), but keep *direction fixed*.

Sneak peek: matrix operation (**Av**) can change *length, direction and also dimensionality*!

Vectors: Magnitude (Length) and Phase (direction)

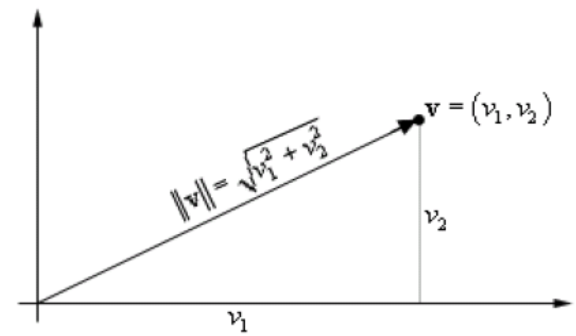
$$v = (x_1, x_2, \ldots, x_n)^{\mathrm{T}}$$

$$\|v\| = \sqrt{\sum_{i=1}^{n} x_i^2}$$ (**Magnitude or "2-norm"**)

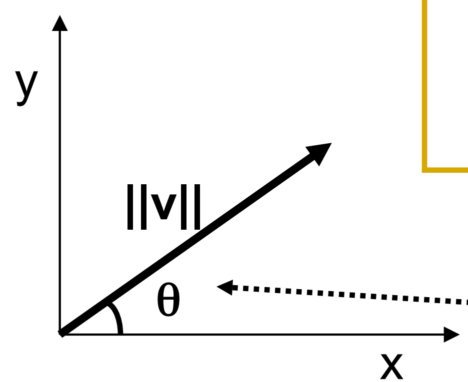
If $\|v\| = 1$, v is a unit vector

$\mathbf{u} = \frac{1}{\|\mathbf{v}\|}\mathbf{v}$, then $\mathbf{u}$ is a unit vector.

(unit vector => **pure direction**)

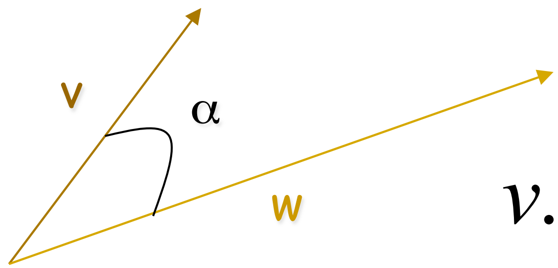


Alternate representations:
Polar coords: (**||v||, θ**)
Complex numbers: **$\|\mathbf{v}\|e^{j\theta}$**



"phase"

Inner (dot) Product: v.w or $w^T v$



$$v.w = (x_1, x_2).(y_1, y_2) = x_1 y_1 + x_2 . y_2$$

The inner product is a **SCALAR!**

$$v.w = (x_1, x_2).(y_1, y_2) = \| v \| \cdot \| w \| \cos \alpha$$

$$v.w = 0 \Leftrightarrow v \perp w$$

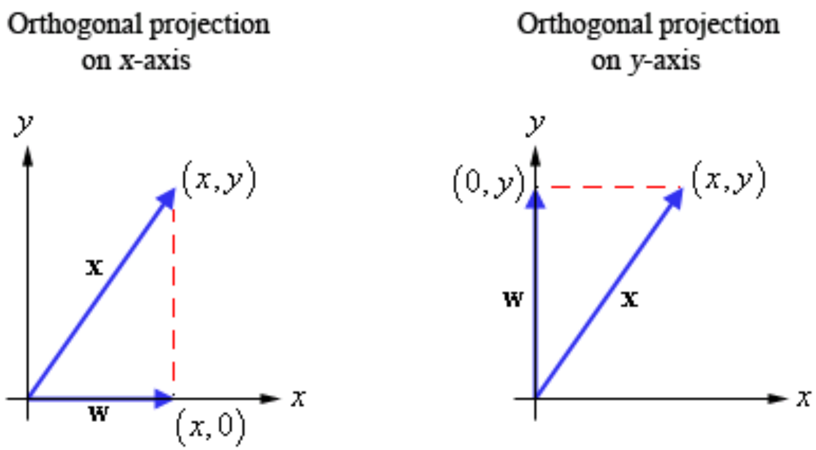
If vectors **v, w** are “columns”, then dot product is $\mathbf{w}^T\mathbf{v}$

Module 3

- Review of Linear Algebra

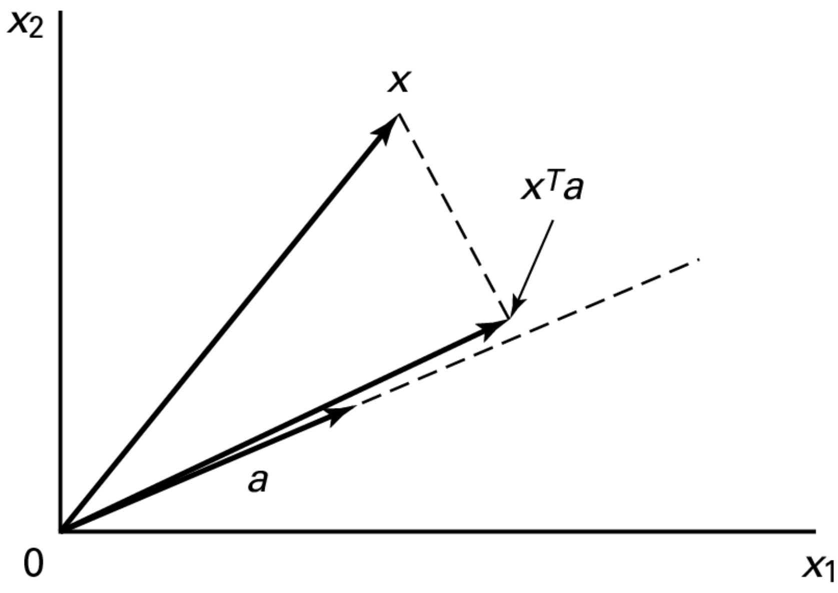
Projections w/ Orthogonal Basis

- Get the component of the vector on each axis:
  - dot-product with unit vector on each axis!



Aside: this is what Fourier transform does!
Projects a function onto a **infinite** number of orthonormal basis **functions: ($e^{j\omega}$ or $e^{j2\pi n\theta}$)**, and adds the results up (to get an equivalent “representation” in the “frequency” domain).

Projection: Using Inner Products -1



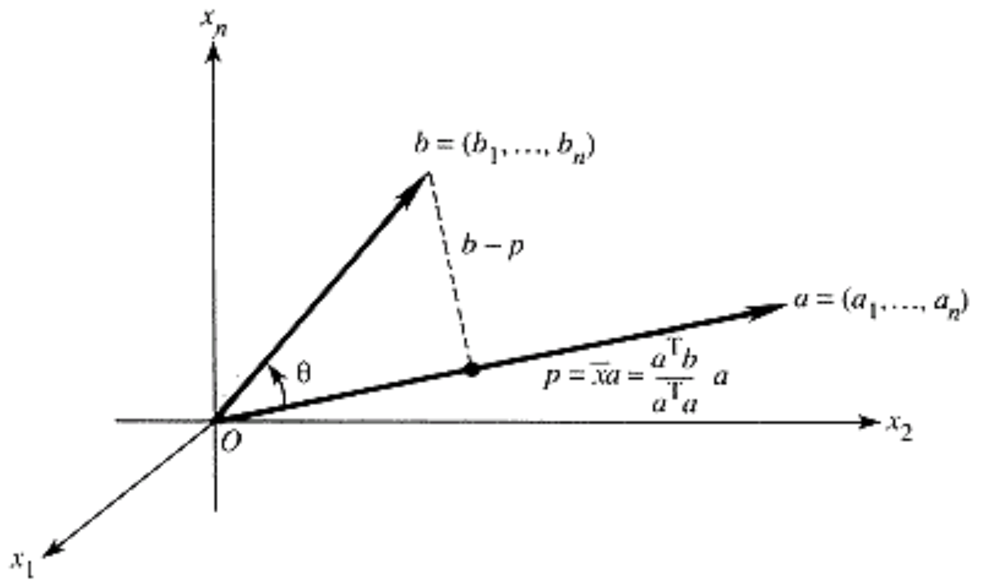
Projection of x along the direction **a** ($\|\mathbf{a}\| = 1$).

p = **a** ($a^T x$)
$\|a\| = a^T a = 1$

Projection: Using Inner Products -2

$\mathbf{p} = \mathbf{a}\ (a^T b) / (a^T a)$

Note: the "error vector" $\mathbf{e} = \mathbf{b} - \mathbf{p}$
is orthogonal (perpendicular) to **p**.
i.e. Inner product: $(\mathbf{b}-\mathbf{p})^T \mathbf{p} = 0$



Review of Linear Algebra - 1

- Consider

$w_1x_1 + w_2x_2 + b = 0 = w^Tx + b = w.x + b$

- In the x_1x_2-coordinate system, this is the equation of a straight line

Proof: Rewrite this as

$x_2 = (w_1/w_2)x_1 + (1/w_2)\, b = 0$

Compare with $y = m\, x + c$

This is the equation of a straight line with slope $m = (w_1/w_2)$ and intercept $c = (1/w_2)$

Other Forms of Eqn of St Line

$w_1x_1 + w_2x_2 + b = 0$

$w^Tx + b = 0$

$w.x + b = 0;$

$<w,x> + b = 0$

Review of Liner Algebra - 2

1. w.x = 0 is the eqn of a st line thru origin
2. w. x + b = 0 is the eqn of any straight line
3. w. x + b = +1 is the eqn of a straight line parallel to (2) on the positive side of Eqn (1) at a distance 1
4. w. x + b = -1 is the eqn of a straight line parallel to (2) on the negative side of Eqn (1) at a distance 1

Module 4

- Classifiers & Classifier Margin

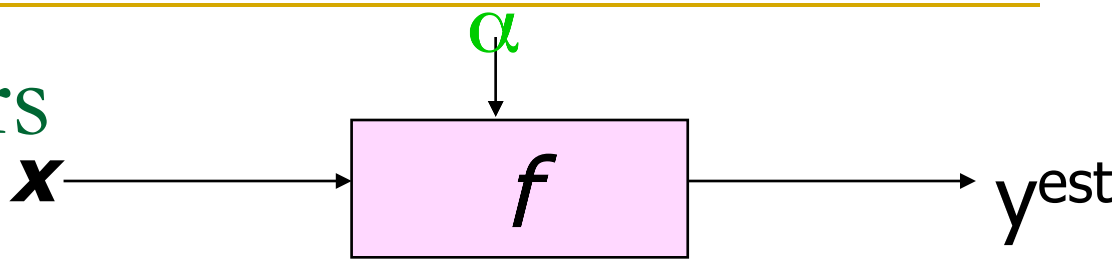
Define a Binary Classifier

- Define f as a classifier
- $f = f\,(w, x, b) = \text{sign}\,(w.x + b)$
- If $f = +1$, x belongs to Class 1
- If $f = -1$, x belongs to Class 2
- We call f a linear classifier because

  $w.x + b = 0$ is a straight line.

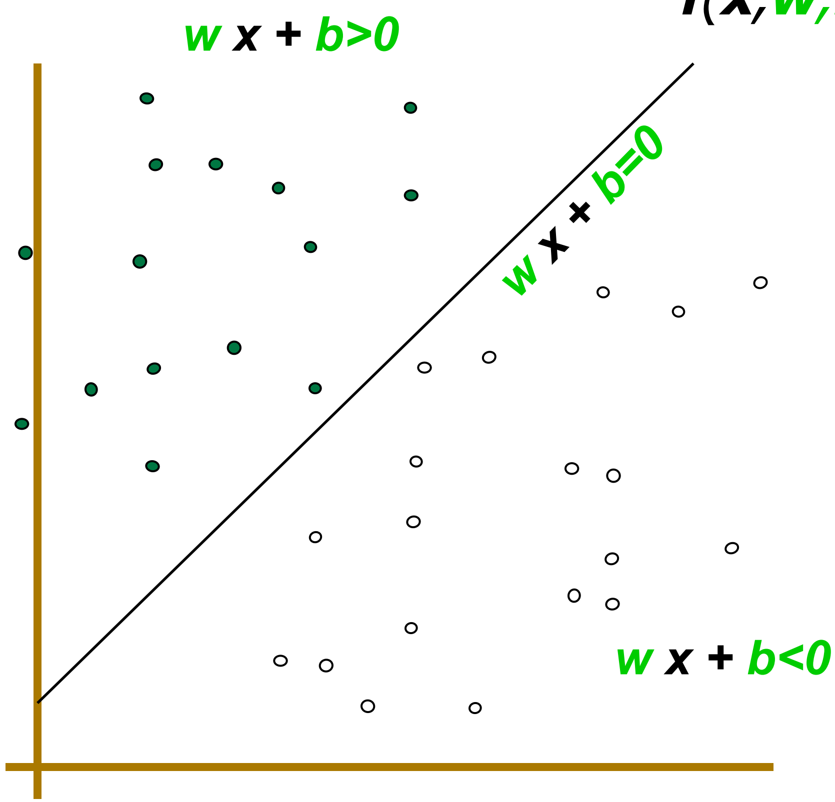
This line is called the class boundary

Linear Classifiers



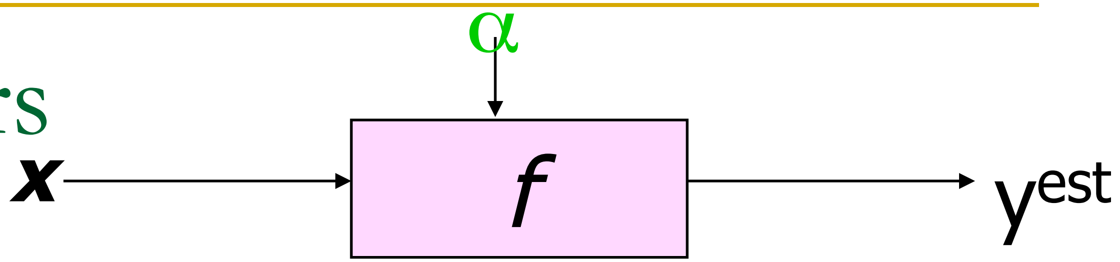
$f(\mathbf{x},\mathbf{w},b) = sign(\mathbf{w}\ \mathbf{x} + b)$

- ● denotes +1
- ○ denotes -1

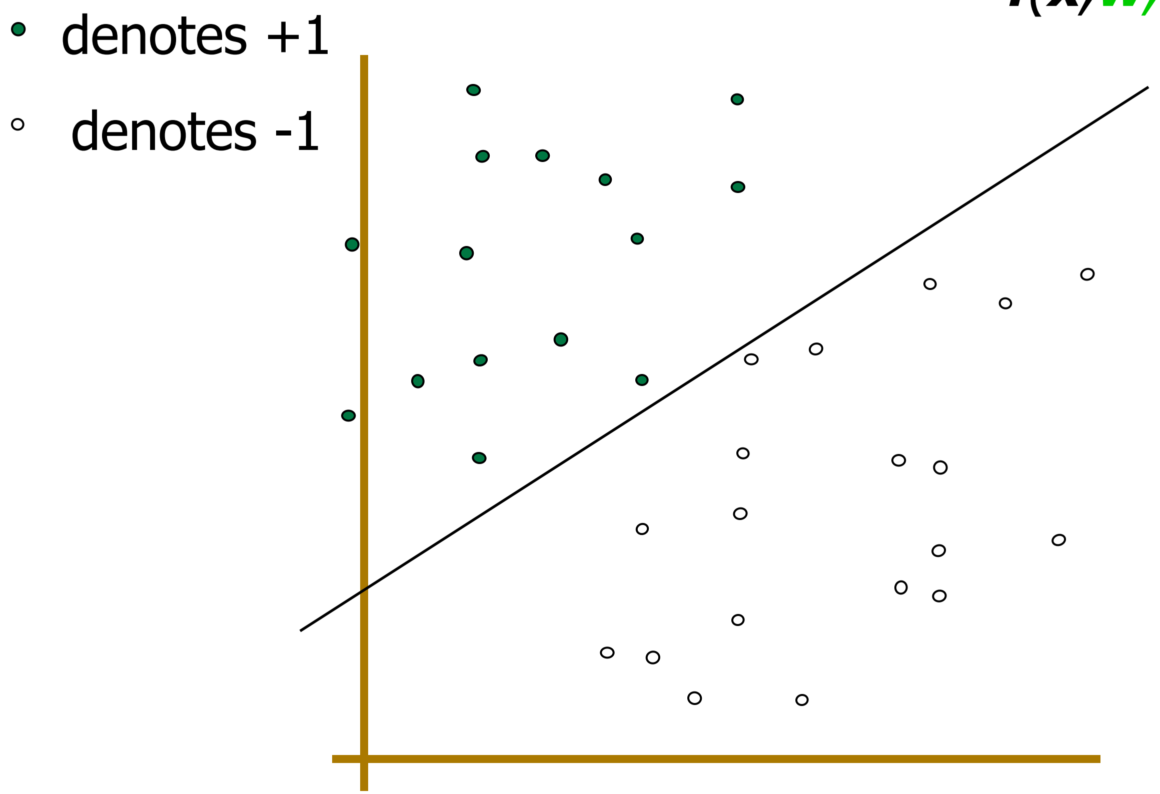


How would you classify this data?

Linear Classifiers

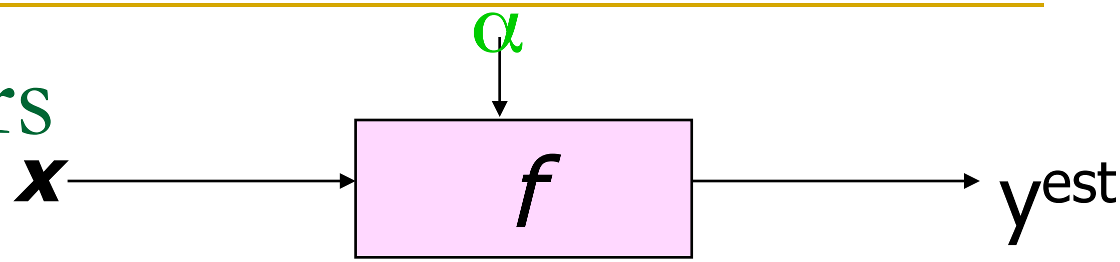


$f(\mathbf{x},\mathbf{w},b) = sign(\mathbf{w}\ \mathbf{x} + b)$



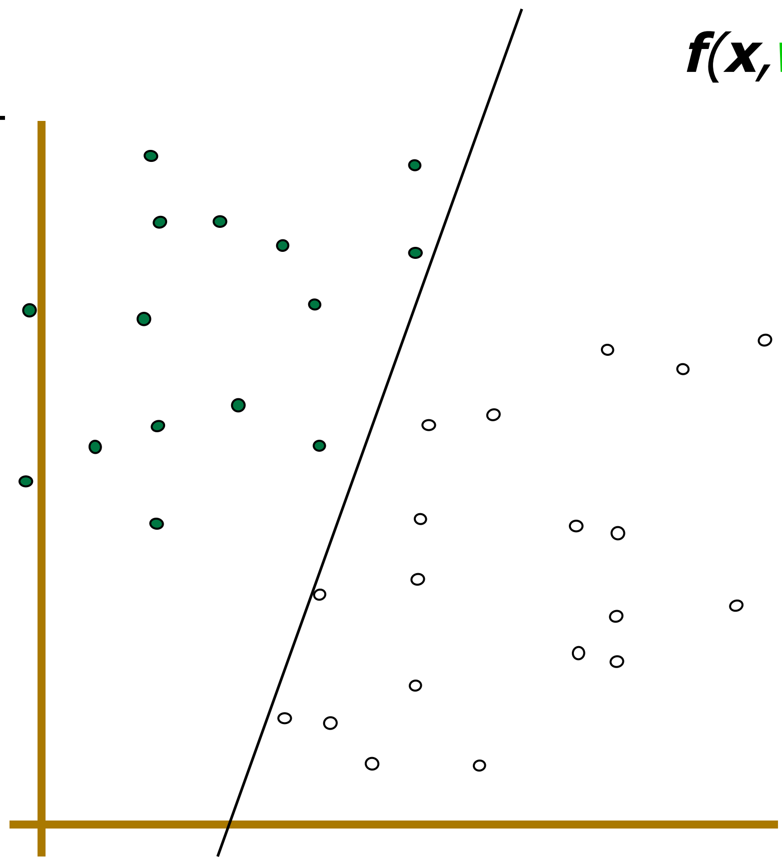
How would you classify this data?

Linear Classifiers



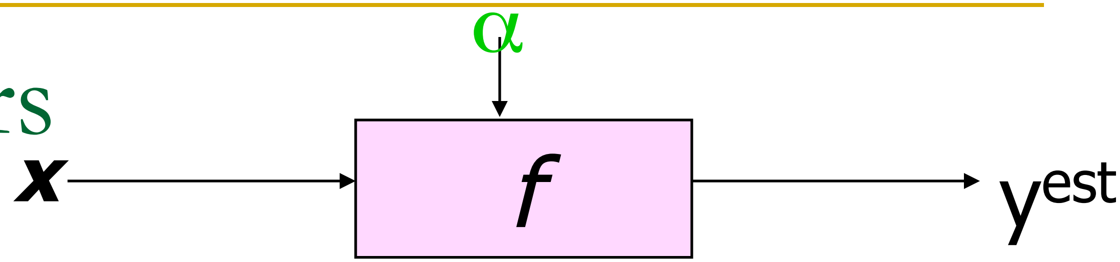
$f(\mathbf{x},\mathbf{w},b) = sign(\mathbf{w}\ \mathbf{x} + b)$

- denotes +1
- denotes -1



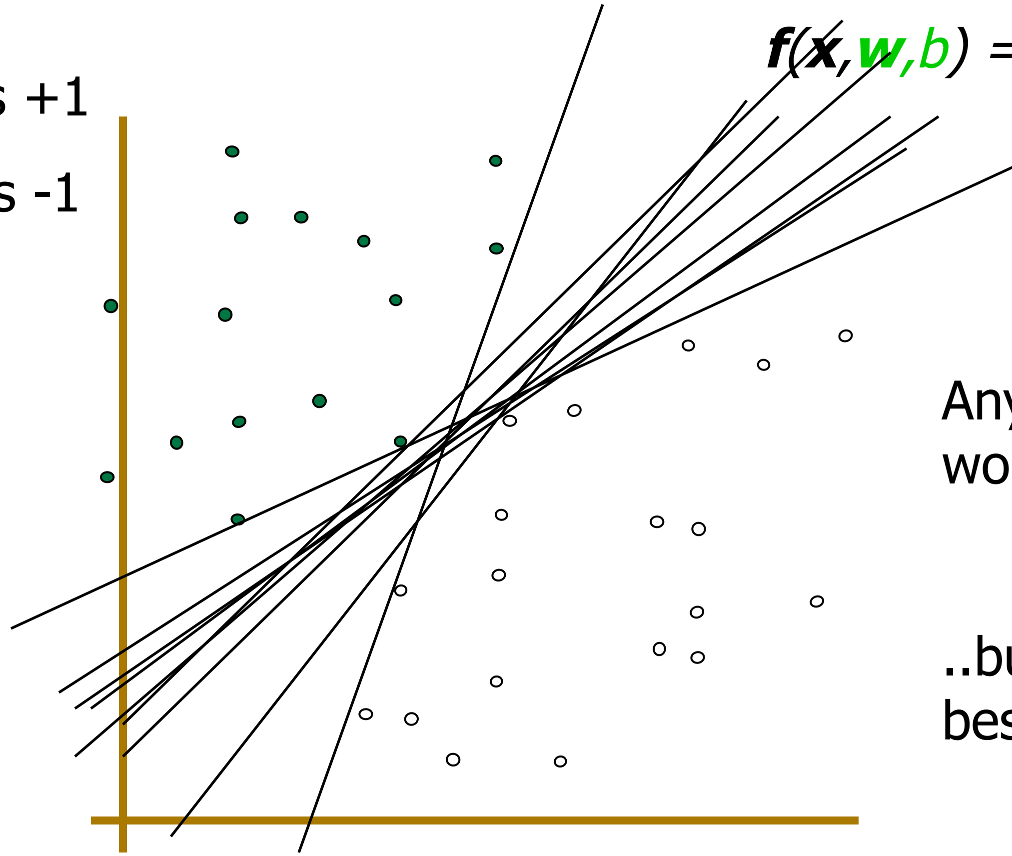
How would you classify this data?

Linear Classifiers



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

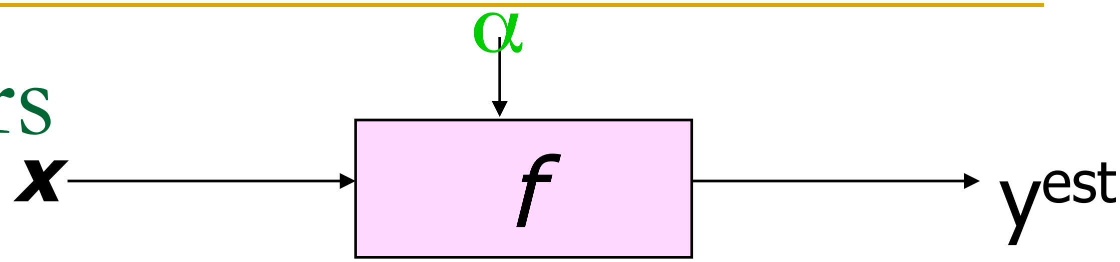
- denotes +1
- denotes -1



Any of these would be fine..

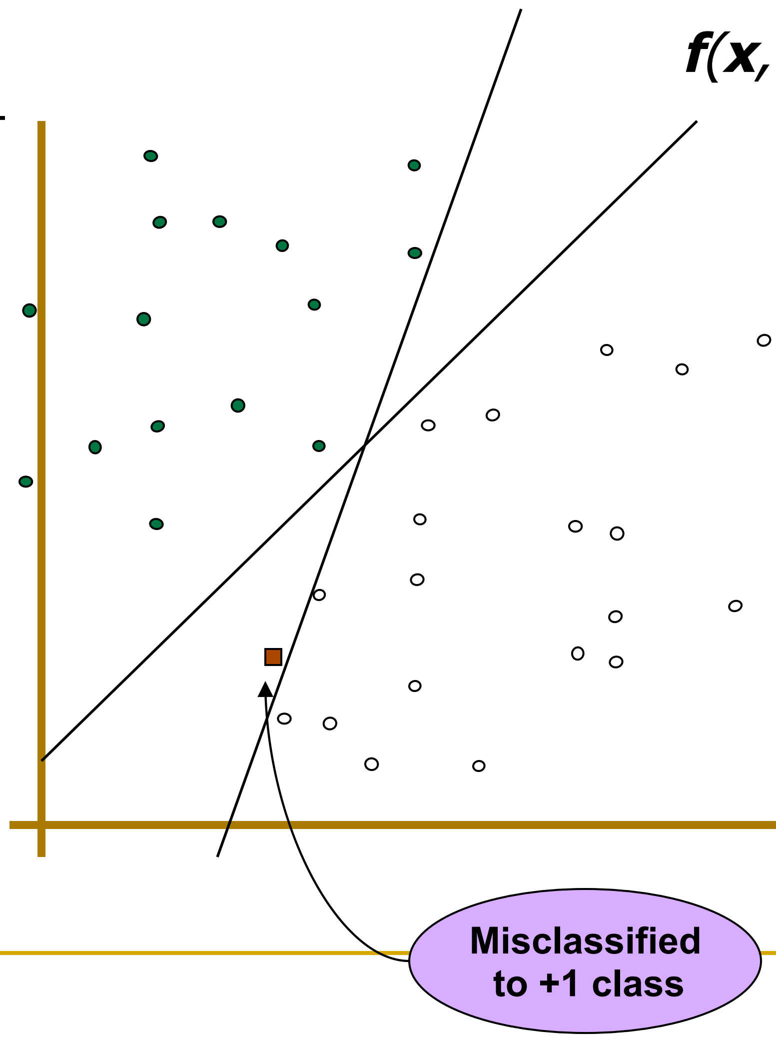
..but which is best?

Linear Classifiers



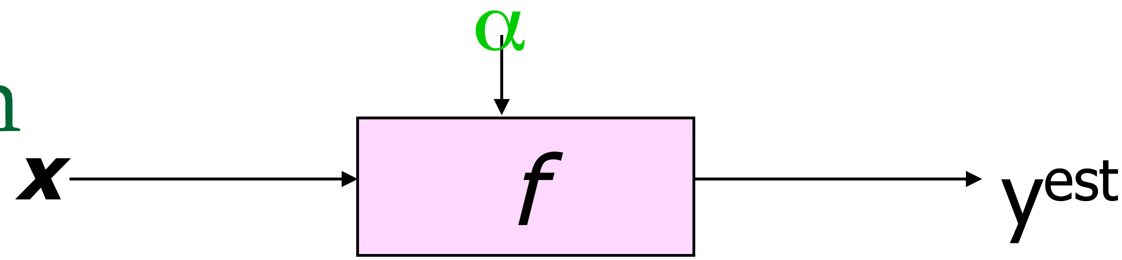
$$f(\mathbf{x},\mathbf{w},b) = sign(\mathbf{w}\ \mathbf{x} + b)$$

- ● denotes +1
- ○ denotes -1



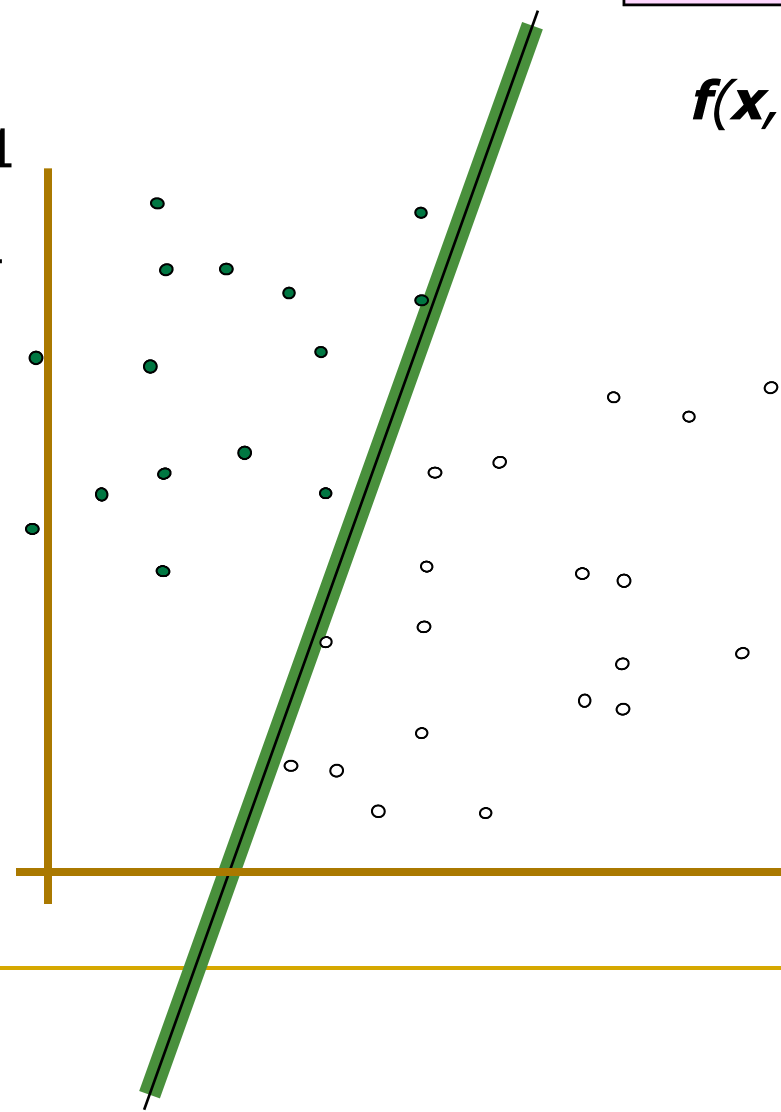
How would you classify this data?

Classifier Margin



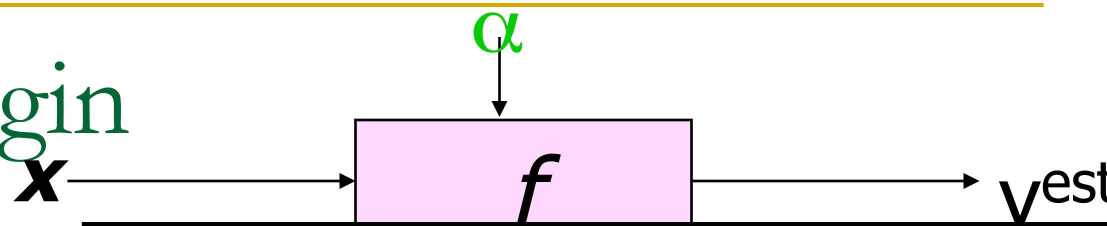
$\boldsymbol{f}(\mathbf{x},\mathbf{w},b) = sign(\mathbf{w}\,\mathbf{x} + b)$

- denotes +1
- denotes -1



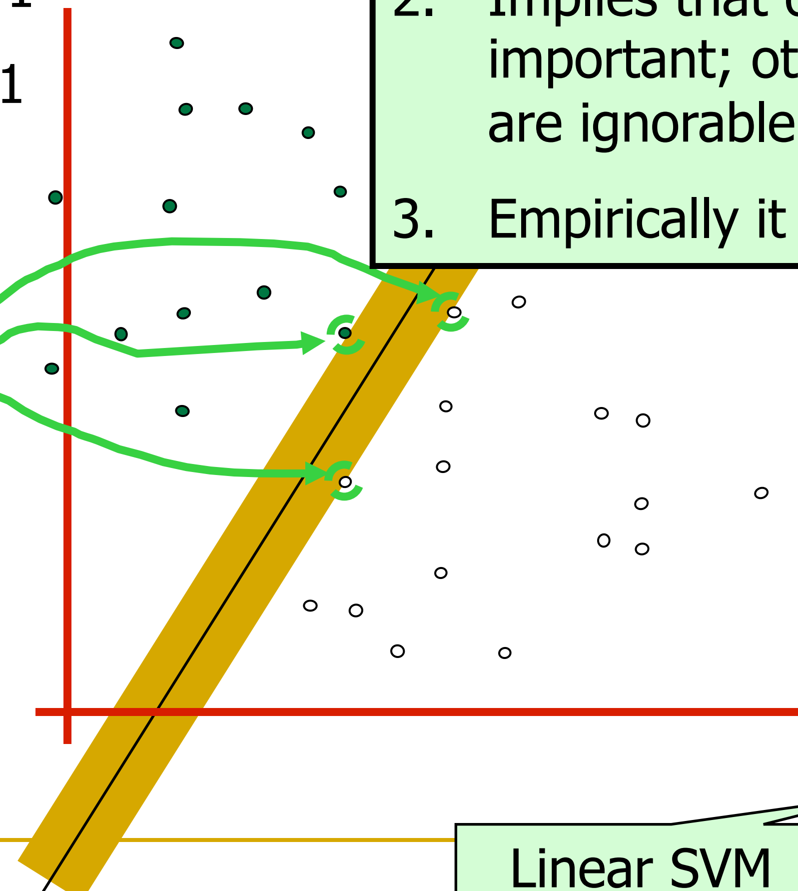
Define the margin of a linear classifier as the width that the boundary could be increased by before hitting a datapoint.

Maximum Margin



1. Maximizing the margin is good according to intuition and PAC theory
2. Implies that only support vectors are important; other training examples are ignorable.
3. Empirically it works very very well.

- denotes +1
- ○ denotes -1



Support Vectors are those datapoints that the margin pushes up against

linear classifier with the, um, maximum margin.

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

Linear SVM

Significance of Maximum Margin - 1

- From the perspective of statistical learning theory, the motivation for considering the Binary Classifier SVM’s comes from theoretical bounds on generalization error
- These bounds have two important features

Significance of Maximum Margin - 2

- The upper bound on the generalization error does not depend upon the dimensionality of the space
- The bound is minimized by maximizing the margin

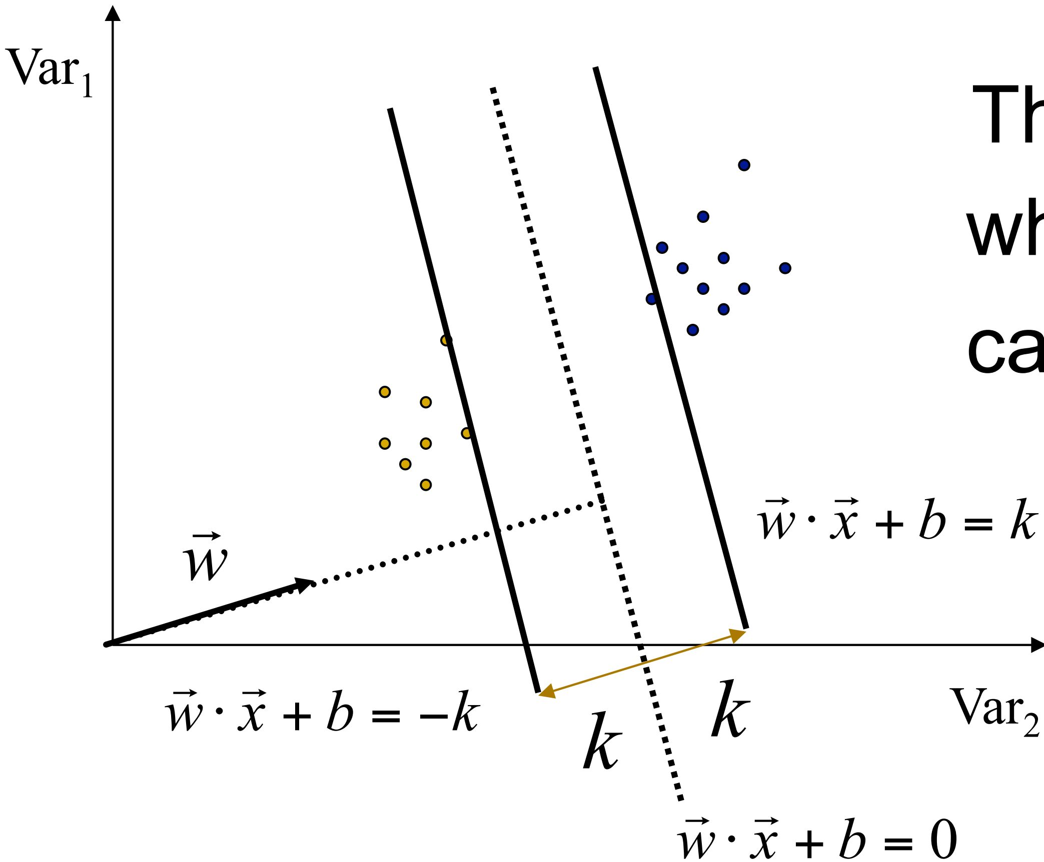
Module 5

Linear SVMs: Optimization Problem

SVMs: Three Main Ideas

1. Define an optimal hyperplane for a linearly separable case:
    1. One that maximizes the margin
    2. Solve the optimization problem
2. Extend the definition to non-linearly separable cases:
    1. Have a penalty term for misclassifications
3. Map data to a high dimensional space where it is easier to classify with linear decision surfaces:
    1. reformulate problem so that data is mapped implicitly to this space

Setting Up the Optimization Problem



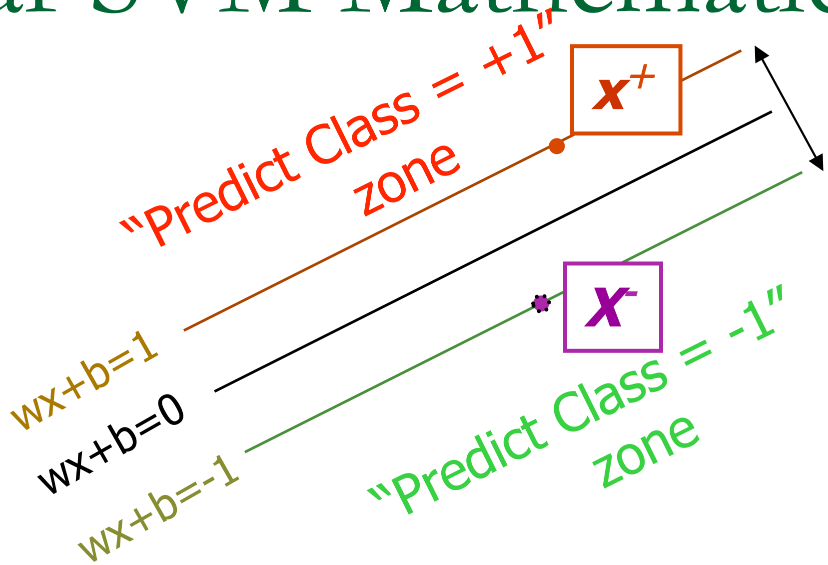
- The hyperplanes when k =1 are canonical planes

An Observation

- The vector w is perpendicular to the Plus plane. Why?
- Why choose wx+b = +1 and wx+b = -1 as the planes defining margin boundaries?
- Because—
- Let u and v be two vectors in the Plus plane. Then what is

  w.(u-v)?
- because sign (wx+b) has TWO degrees of freedom and what matters in their ratio

Linear SVM Mathematically



M=Margin Width

What we know:

- $\boldsymbol{w} \,.\, \boldsymbol{x}^+ + b = +1$
- $\boldsymbol{w} \,.\, \boldsymbol{x}^- + b = -1$
- $\boldsymbol{w} \,.\, (\boldsymbol{x}^+ - \boldsymbol{x}^-) = 2$

Also

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

Define

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

Calculation of Lambda

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

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

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

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

$$-1+\lambda(w.w)=+1 \Rightarrow \lambda=\frac{2}{w.w}$$

Calculation of Margin, M

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

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

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

Learning the Maximum Margin Classifier

- Given a guess of **w** and b we can compute
  - whether all data points are in the correct half-planes
  - the width of the margin
- Now we just need to write a program to search the space of **w**'s and b's to find the widest margin that matches all the data points. How?
- Gradient descent? Matrix Inversion? EM? Newton's Method?

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.

Linear SVM Mathematically

- Goal: **1) Correctly classify all training data**

$$wx_i + b \geq 1 \quad \text{if } y_i = +1$$
$$wx_i + b \leq -1 \quad \text{if } y_i = -1$$
$$y_i(wx_i + b) \geq 1 \quad \text{for all i}$$

2) Maximize the Margin $M = \dfrac{2}{\|w\|}$

same as minimize $\dfrac{1}{2}w^T w$

- **We can formulate a Quadratic Optimization Problem and solve for w and b**

- Minimize $\Phi(w) = \dfrac{1}{2}w^T w$

  subject to $y_i(wx_i + b) \geq 1 \quad \forall i$

Solving the Constrained Minimization

- Classical method is to minimize the associated un-constrained problem using Lagrange multipliers. That is minimize

$$L(\vec{w},b)=\frac{1}{2}\vec{w}.\vec{w}-\sum_{i=1}^{N}\alpha_i\left[y_i((\vec{w}.\vec{x}_i)+b)-1\right]$$

- This is done by finding the saddle points:

$$\partial L / \partial b = 0 \; gives \sum \alpha_i y_i = 0$$

..contd

$$\partial L / \partial w = 0 \; gives \; w = \sum \alpha_i y_i x_i$$

- Which when substituted back in L tells us that we should maximize the functional:

$$W(\alpha) = \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{N} \alpha_i \alpha_j y_i y_j < x_i, x_j >$$

- Subject to alphas greater than or equal to 0

...contd

- Subject to the constraints

$$\alpha_i \geq 0$$

- and

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

Decision Surface

- The decision surface then is defined by

$$D(z) = sign\left(\sum_{i}^{N} \alpha_i y_i (x_i - z) + b\right)$$

- Where z is a test vector

Solving the Optimization Problem

Find **w** and b such that
$\mathbf{\Phi}(\mathbf{w}) = \frac{1}{2}\, \mathbf{w}^T\mathbf{w}$ is minimized;
and for all $\{(\mathbf{x_i}, y_i)\}$: $y_i(\mathbf{w}^\mathbf{T}\mathbf{x_i} + b) \geq 1$

- **Need to optimize a *quadratic* function subject to *linear* constraints.**
- **Quadratic optimization problems are a well-known class of mathematical programming problems, and many (rather intricate) algorithms exist for solving them.**
- **The solution involves constructing a *dual problem* where a *Lagrange multiplier* α_i is associated with every constraint in the primary problem:**

Find $\alpha_1 \ldots \alpha_N$ such that
$\mathbf{Q}(\boldsymbol{\alpha}) = \Sigma\alpha_i - \frac{1}{2}\Sigma\Sigma\alpha_i\alpha_j y_i y_j \mathbf{x_i}^\mathbf{T}\mathbf{x_j}$ is maximized and
(1) $\Sigma\alpha_i y_i = 0$
(2) $\alpha_i \geq 0$ for all α_i

The Optimization Problem Solution

- The solution has the form:

$$\boxed{\mathbf{w} = \Sigma\alpha_i y_i \mathbf{x_i} \qquad b = y_k - \mathbf{w^T x_k} \text{ for any } \mathbf{x_k} \text{ such that } \alpha_k \neq 0}$$

- Each non-zero α_i indicates that corresponding $\mathbf{x_i}$ is a support vector.
- Then the classifying function will have the form:

$$\boxed{f(\mathbf{x}) = \Sigma\alpha_i y_i \mathbf{x_i}^\mathbf{T}\mathbf{x} + b}$$

- Notice that it relies on an *inner product* between the test point $\mathbf{x}$ and the support vectors $\mathbf{x_i}$ – we will return to this later.
- Also keep in mind that solving the optimization problem involved computing the inner products $\mathbf{x_i}^\mathbf{T}\mathbf{x_j}$ between all pairs of training points.

SVM in ScikitLearn

```
from sklearn import svm
SVMclassifier = svm.SVC()
```

SVR in Scikit-Learn

```
from sklearn.Linear_model import LinearRegression
from sklearn import svm
```

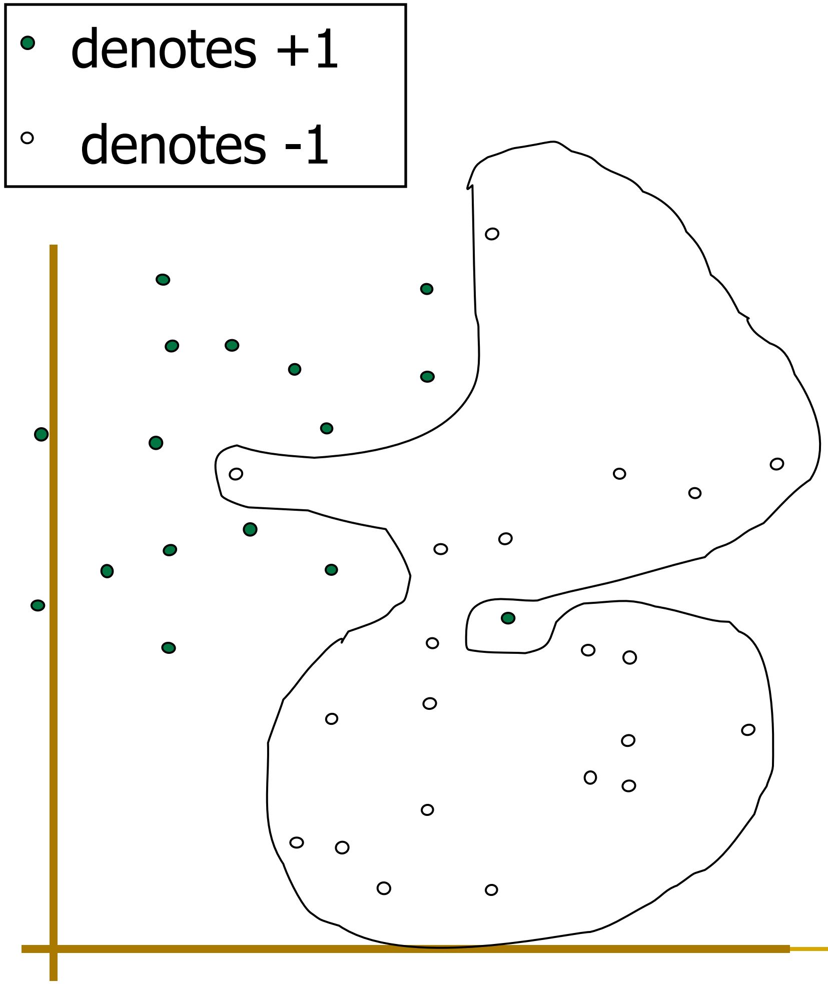
#svm contains methods for support vector regression (eg, SVR and LinearSVR). Let us instantiate a couple of SVR models

```
svm_lm = svm.SVR(kernel='Linear', C = 1e1) # linear kernel
Svm_rbf = svm.SVR(kernel ='rbf', C = 1e1) # Gaussian kernel
```

Module 6

Hard Vs Soft Margin Classification

Dataset with noise



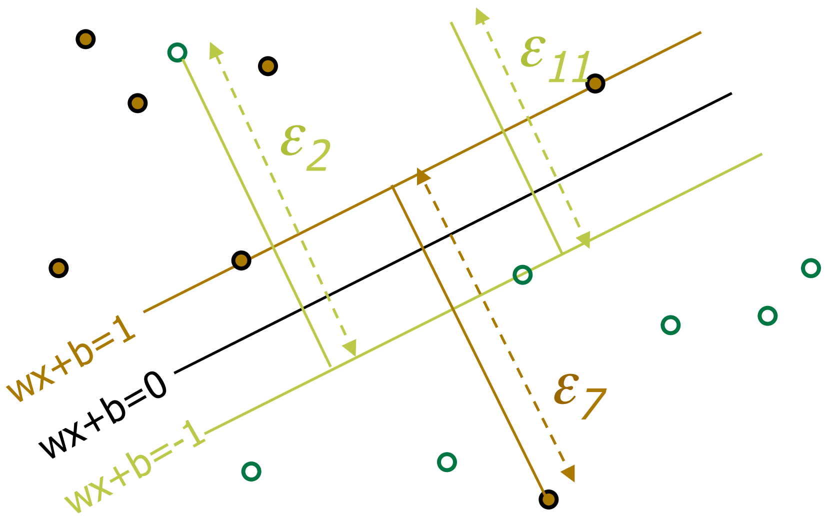
- **Hard Margin:** **So far we required**
  - **all data points be classified correctly**
  - **Allowed NO training errors**
- **What if the training set is noisy?**

- Solution 1: **use very powerful kernels**

OVERFITTING!

Soft Margin Classification

***Slack variables ξi* can be added to allow misclassification of difficult or noisy examples.**



What should our quadratic optimization criterion be?

Minimize

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

Count of Constraints

$$w \,.\, x_k + b \geq 1 - \xi_k \quad if \;\; y_k = 1$$

$$w \,.\, x_k + b \leq -1 + \xi_k \quad if \;\; y_k = -1$$

$$\xi_k \geq 0$$

There are 2R constraints, R= number of instances.

We have w_1,w_2,...,w_m, b weights and E and the above R “epsilons”

Dual Formula for Soft Margin

$$\sum_{k=1}^{R} \alpha_k - \frac{1}{2} \sum_{k=1}^{R} \sum_{l=1}^{R} \alpha_k \alpha_l Q_{kl}, \quad Q_{kl} = y_k y_l (x_k . x_l)$$

$$subject\ to\ the\ constra\,\text{int}\,s$$

$$0 \leq \alpha_k \leq C \quad for\ \forall k \quad \sum_{k=1}^{R} \alpha_k y_k = 0$$

- Then define

$$w = \sum_{k=1}^{R} \alpha_k y_k x_k$$

$$b = y_K (1 - \xi_K) - x_K . w_K$$

$$where$$

$$K = \arg\max_k \alpha_k$$

Now classify, using $f(x,w,b) = sign(w.x + b)$

- Data points with $\alpha_k > 0$ are support vectors
- So the summation for w need to be done only over support vectors

Hard Margin Vs Soft Margin

- **The old formulation:**

Find $\mathbf{w}$ and b such that
$\mathbf{J(w)} = ½\, \mathbf{w}^{\mathrm{T}}\mathbf{w}$ is minimized and for all $\{(\mathbf{x_i}, y_i)\}$
$y_i\,(\mathbf{w^T x_i} + \mathrm{b}) \geq 1$

- **The new formulation incorporating slack variables:**

Find $\mathbf{w}$ and b such that
$\mathbf{J(w)} = ½\, \mathbf{w}^{\mathrm{T}}\mathbf{w} + C\Sigma\xi_i$ is minimized and for all $\{(\mathbf{x_i}, y_i)\}$
$y_i\,(\mathbf{w^T x_i} + b) \geq 1 - \xi_i$ and $\xi_i \geq 0$ for all i

- **Parameter *C* can be viewed as a way to control overfitting.**
- **This is "constrained optimization"**

Linear SVMs: Summary

- **The classifier is a *separating hyperplane.***
- **Most "important" training points are support vectors; they define the hyperplane.**
- **Quadratic optimization algorithms can identify which training points $\mathbf{x}_i$ are support vectors with non-zero Lagrangian multipliers α_i.**
- **Both in the dual formulation of the problem and in the solution training points appear only inside dot products:**

Find $\alpha_1 \ldots \alpha_N$ such that
$Q(\alpha) = \Sigma\alpha_i - \frac{1}{2}\Sigma\Sigma\alpha_i\alpha_j y_i y_j \mathbf{x}_i^T\mathbf{x}_j$ is maximized and
(1) $\Sigma\alpha_i y_i = 0$
(2) $0 \leq \alpha_i \leq C$ for all α_i

$$f(\mathbf{x}) = \Sigma\alpha_i y_i \mathbf{x}_i^T\mathbf{x} + b$$

Comments on Dual Formulation!

- The vector w could be infinite-dimensional and poses problems computationally
- There are only as many Lagrange variables, “alphas”, as there are training instances
- As a bonus, it turns out that the “alphas” are non-zero only for the support vectors (far fewer in number than the training data)