Support Vector Machines

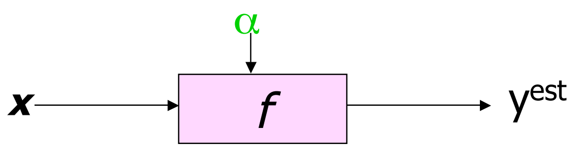
Rezarta Islamaj Dogan

Resources

- Support Vector Machines tutorial
  Andrew W. Moore
  - www.autonlab.org/tutorials/svm.html

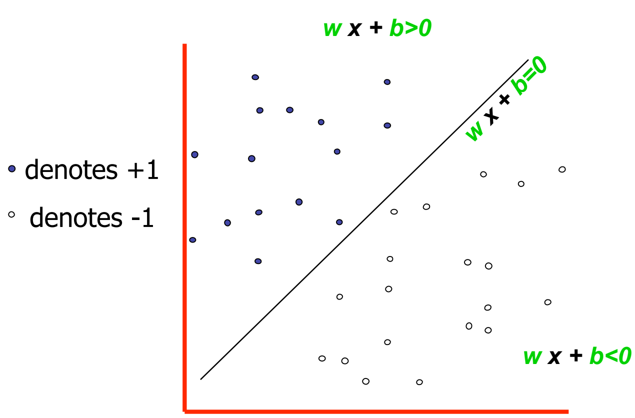
- **A tutorial on support vector machines for pattern recognition.**
  C.J.C. Burges.
  - **Data Mining and Knowledge Discovery, 2(2):955-974, 1998.**

Linear Classifiers



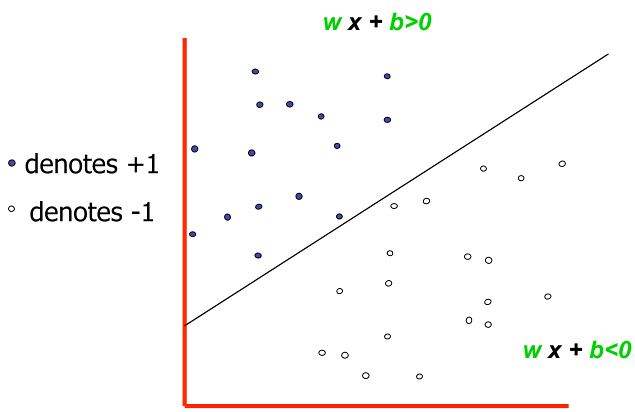
$f(\mathbf{x}, \mathbf{w}, b) = sign(\mathbf{w}\,\mathbf{x} + b)$

Linear Classifiers

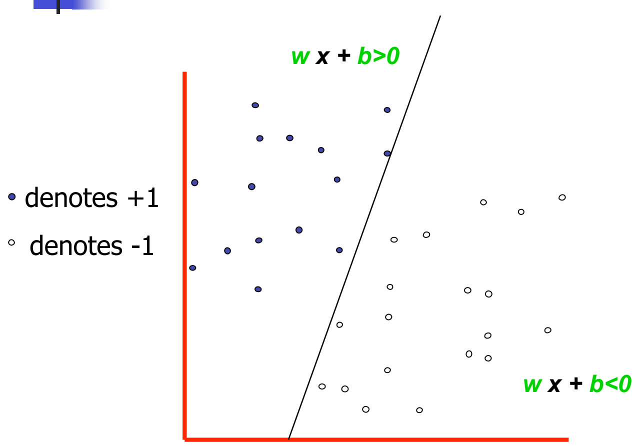


How to classify this data?

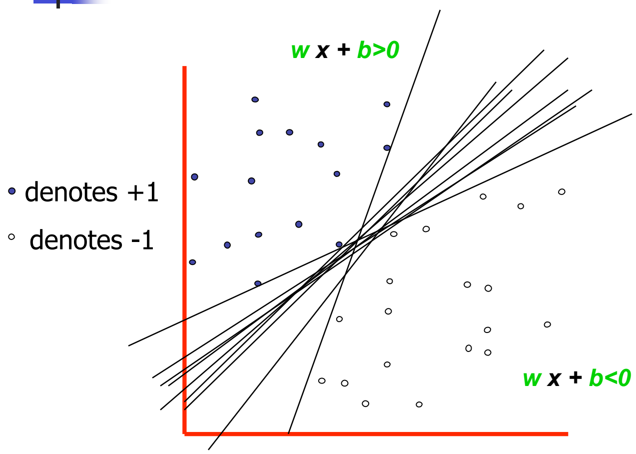
Linear Classifiers



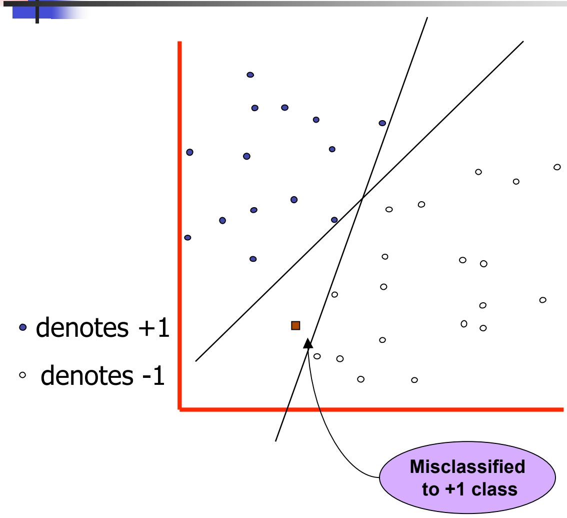
Linear Classifiers



Linear Classifiers



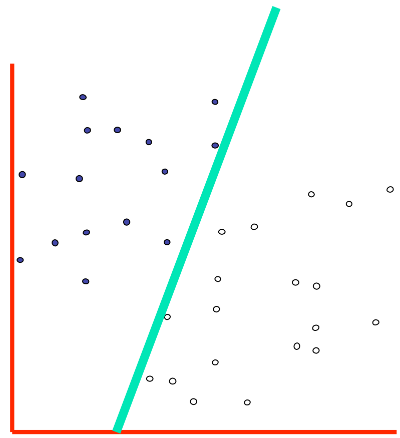
Linear Classifiers



Classifier Margin

• denotes +1

◦ denotes -1

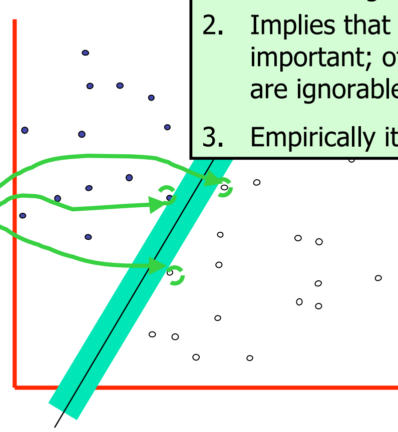


Margin:
the width that the boundary could be increased by before hitting a datapoint.

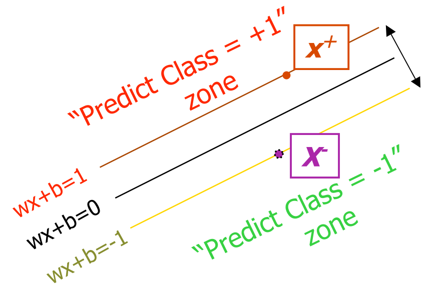
Maximum Margin Classifier

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

Support Vectors are those datapoints that the margin pushes up against



Finding the boundary



M=Margin Width

What we know:

- ***w*** *.* ***x****⁺* + *b* = +1
- ***w*** *.* ***x****⁻* + *b* = -1
- ***w*** *.* *(****x****⁺-****x****⁻)* = 2

$$M = \frac{(x^+ - x^-)\cdot w}{|w|} = \frac{2}{|w|}$$

Learning the Maximum Margin Classifier

Given a guess for w and b

- Compute whether all data points in correct half-planes
- Compute the width of the margin

Search the space of w's and b's to find the widest margin that matches all data points

- Quadratic programming

Linear SVM

Correctly classify all training data

$$wx_i + b \geq 1 \quad \text{if } y_i = +1$$

$$wx_i + b \leq 1 \quad \text{if } y_i = -1$$

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

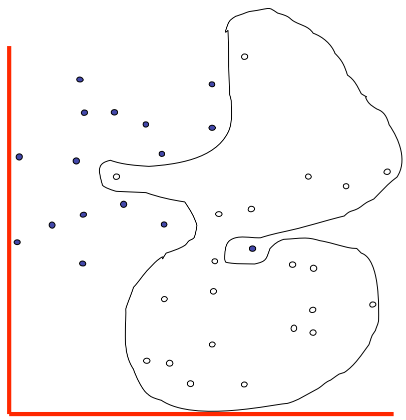
Maximize: $M = \frac{2}{|w|}$ → Minimize: $\frac{1}{2} w^t w$

Solving the Optimization Problem

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

Need to optimize a *quadratic* function subject to *linear* constraints.
Quadratic optimization problems are a well-known class of mathematical programming problems.
The solution involves constructing a *dual problem* where a *Lagrange multiplier* is associated with every constraint in the primary problem.

Maximum Margin Classifier with Noise



Hard Margin: requires that all data points are classified correctly

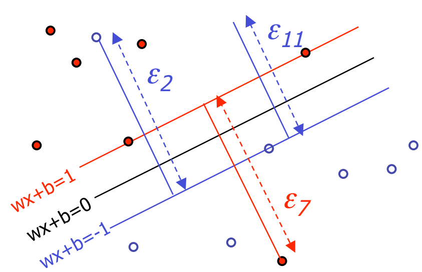
What if the training set is noisy?

- Solution: use very powerful kernels

OVERFITTING!

Soft Margin Classification

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



Minimize:

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

Hard Margin v.s. Soft Margin

The old formulation:

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

The new formulation incorporating slack variables:

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

Parameter *C* can be viewed as a way to control overfitting.

Linear SVMs: Overview

The classifier is a *separating hyperplane.*

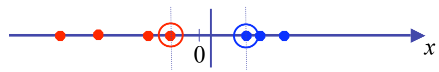
Most "important" training points are support vectors; they define the hyperplane.

Quadratic optimization algorithms can identify which training points x_i are support vectors with non-zero Lagrangian multipliers.

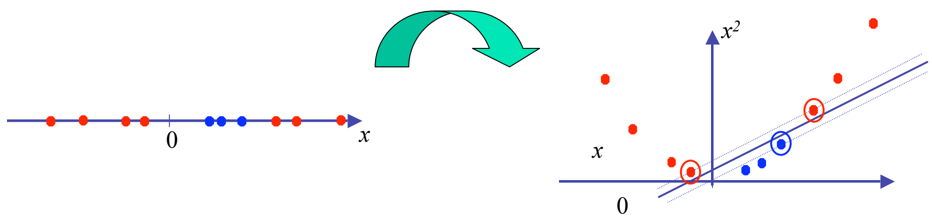
Both in the dual formulation of the problem and in the solution training points appear only inside dot products

Non-linear SVMs

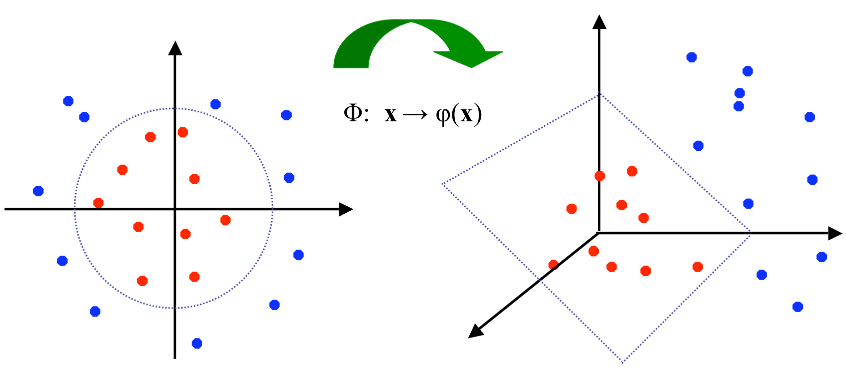
Linearly separable data:



Map data to a higher-dimensional space:



Non-linear SVMs: Feature spaces



The original input space can always be mapped to some higher-dimensional feature space where the training set is separable:

Kernels

The linear classifier relies on dot product between vectors $K(x_i,x_j)=x_i^T x_j$

If every data point is mapped into high-dimensional space via some transformation Φ: $x \rightarrow \varphi(x)$, the dot product becomes:

$$K(x_i,x_j)= \varphi(x_i)^T\varphi(x_j)$$

A *kernel function* is some function that corresponds to an inner product in some expanded feature space.

Kernels

Linear: $K(\mathbf{x_i},\mathbf{x_j})= \mathbf{x_i}^T\mathbf{x_j}$

Polynomial of power p: $K(\mathbf{x_i},\mathbf{x_j})= (1+ \mathbf{x_i}^T\mathbf{x_j})^p$

Gaussian (radial-basis function network):

$$K(\mathbf{x_i},\mathbf{x_j}) = \exp(-\frac{\|\mathbf{x_i}-\mathbf{x_j}\|^2}{2\sigma^2})$$

Sigmoid: $K(\mathbf{x_i},\mathbf{x_j})= \tanh(\beta_0\mathbf{x_i}^T\mathbf{x_j} + \beta_1)$

Nonlinear SVM - Overview

- SVM locates a separating hyperplane in the feature space and classifies points in that space

- It does not need to represent the space explicitly, simply by defining a kernel function

- The kernel function plays the role of the dot product in the feature space.

Properties of SVM

- **Flexibility in choosing a similarity function**
- **Sparseness of solution when dealing with large data sets**
  - only support vectors are used to specify the separating hyperplane
- **Ability to handle large feature spaces**
  - complexity does not depend on the dimensionality of the feature space
- **Overfitting can be controlled by soft margin approach**
- **Nice math property: a simple convex optimization problem which is guaranteed to converge to a single global solution**
- **Feature Selection**

SVM Applications

- text (and hypertext) categorization
- image classification
- bioinformatics (Protein classification, Cancer classification)
- hand-written character recognition

Etc.

Multi-class SVM

SVM only considers two classes

For m-class classification problem:

- SVM 1 learns "Output==1" vs "Output != 1"
- SVM 2 learns "Output==2" vs "Output != 2"
- :
- SVM m learns "Output==m" vs "Output != m"

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.