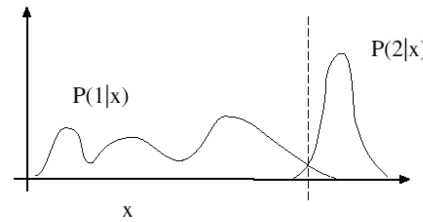
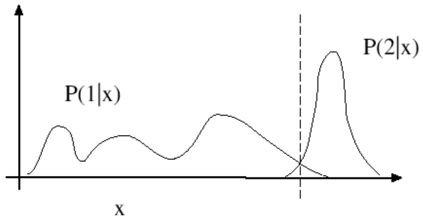
Building classifiers

- Standard scenario
  - Have training data
  - Want to classify new data
- One approach
  - Estimate the probability distributions (we have being thinking about them all along, e.g. P(1|x))
  - Issue: parameter estimates that are “good” may not give optimal classifiers

Finding a decision boundary is not the same as modeling a conditional density.



Finding a decision boundary is not the same as modeling a conditional density.



Important point: P(l|x) can be inaccurate, but the system can work well, as long as the boundary is correct.

Building classifiers

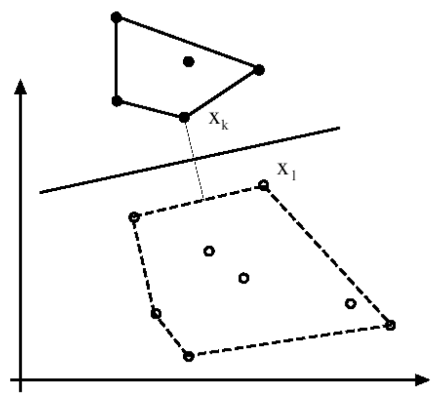
- Standard scenario
  - Have training data
  - Want to classify new data
- One approach
  - Estimate the probability distributions (we have being thinking about them all along, e.g. P(1|x))
  - Issue: parameter estimates that are “good” may not give optimal classifiers
- Another approach
  - Directly go for the boundary

We will start with this one

Support vector machines

- The generic, standard way to do this is with a SVM
- The basic “plug-in classifier” (black box)
- Typically now used for many tasks where before the method of choice was neural networks.
- Very convenient software is now available to do this
- We will cover the approach briefly

Support vector machines



Support vector machines

- If we have a *separating* hyperplane, then if you are on one side $\mathbf{w} \bullet \mathbf{x_i} + b \geq +1$
- If you are on the other side $\mathbf{w} \bullet \mathbf{x_i} + b \leq -1$
- Let y_i be +1 for one class, -1 for the other.

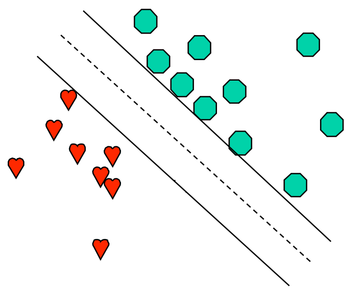
Support vector machines

- Linearly separable data means that we can chose

$$y_i \left(\boldsymbol{w} \cdot \boldsymbol{x}_i + b \right) \geq 1$$

- Consider the best pair of parallel planes that push against points on the two groups.

Support vector machines



Support vector machines

- Consider the best pair of parallel planes that push against points on the two groups.
- The sum of the minimum distances from each group to the other plane can be shown to be:

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

Support vector machines

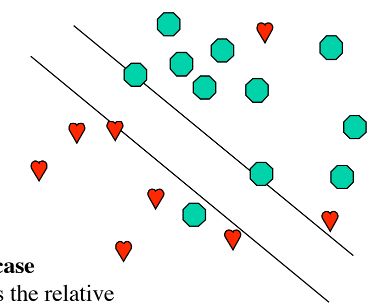
- Solved by

$$\text{minimize} \quad (1/2)\boldsymbol{w}\cdot\boldsymbol{w}$$

$$\text{subject to} \quad y_i\left(\boldsymbol{w}\cdot\boldsymbol{x}_i + b\right) \geq 1$$

- (See book, section 22.5 for how to solve it)
- What if the data is not linearly separable
  - Find “best” plane (see book)
  - The boundary is determined by a few points (the support vectors)

Support vector machines

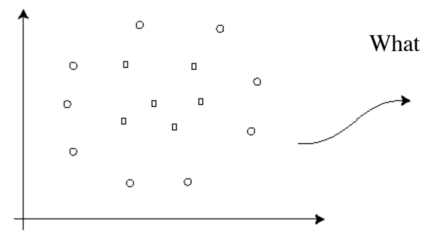


Non-separable case
Cost, C, specifies the relative desire to push the planes apart, verses the number of mistakes.

Support vector machines

- Now that we have the “best” plane, how do we classify?
  - Easy---we have a simple formula for determining which side of the plane we are on!
- Pseudo probabilities can be created from the distance to the plane
- This describes a binary classifier. For more than one class, there are two approaches
  - Multiple one against all
  - All against all, and a consensus measure

Support vector machines (kernel tricks)



What about this case?

Support vector machines (kernel tricks)

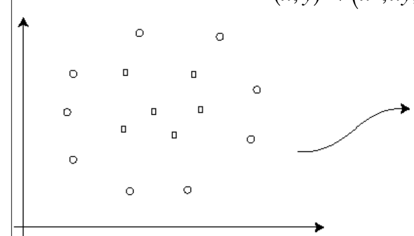
Key observation: The SVM is completely a function of dot products between the vectors.

This means that we can get a non-linear SVM by using a different form of the dot product, K(**x,y**).

This is equivalent to a linear classification in a much higher dimensional space.

Support vector machines (kernel tricks)

$$(x, y) \rightarrow (x^2, xy, y^2, x, y) = (u_0, u_1, u_2, u_3, u_4)$$



Space in which decision boundary is linear - a conic in the original space has the form

$$au_0 + bu_1 + cu_2 + du_3 + eu_4 + f = 0$$

Testing classifiers

- Standard method is to use Cross-Validation
- Test classification accuracy on data not used in training
- Test generalizability by using data that is progressively different than training data
  - new experiment
  - different camera
  - different researchers