Segmentation/Grouping by EM

- See §16.2.1--note that we assume Gaussian for p().
- A segment is modeled as a Gaussian process that emits feature vectors (which could contain color; or color and position; or colour, texture and position).
- Segment parameters are mean and (perhaps) variance or covariance, and prior probability (was λ in the line fitting example).
- If we knew which segment each point belonged to, estimating these parameters would be easy **(this point should be familiar!)**

Segmentation/Grouping by EM

- Since we don't know which point comes from which segment, we have to use an **estimate** of the **probabilities** that a given point belongs to a given segment. We thus compute means and variances by **weighting** the standard formulas by these probabilities.
- Formally, these probabilities can be denoted

$$p(m \mid \mathbf{x}_l, \Theta^{(s)})$$

- This is the probability that $\mathbf{x}_l$ is in cluster m, given the model
- If we assume we know these estimates for the probabilities of the missing values, we can then estimate the means of the Gaussians for each segment.

Segmentation/Grouping by EM

- We estimate the mean for each segment by:
- Variances/covariances work similarly

Segmentation/Grouping by EM

- We estimate the mean for each segment by:

Iteration (step)

$$\mu_m^{(s+1)} = \frac{\sum_{l=1}^{r} \mathbf{x}_l \, p(m \mid \mathbf{x}_l, \Theta^{(s)})}{\sum_{l=1}^{r} p(m \mid \mathbf{x}_l, \Theta^{(s)})}$$

- Variances/covariances work similarly

We can sort out the chicken!

Details optional

Segmentation/Grouping--E Step

- Given parameters, the probability that a given point is associated with each cluster is can be computed by:

$$p(m \mid \mathbf{x}_l, \Theta^{(s)}) = \frac{\alpha_m^{(s)} p(\mathbf{x}_l \mid \theta_m^{(s)})}{\sum_{k=1}^{M} \alpha_k^{(s)} p(\mathbf{x}_l \mid \theta_k^{(s)})}$$

- The book uses $\overline{I_{lm}}$ for $p(m \mid \mathbf{x}_i, \Theta^{(s)})$ (*I* suggests “indicator variable”)

- (Also, my copy of the book’s version of the above equation looks wrong to me--the index *l* applies to points and the index for theta should refer to groups)

Details optional

Segmentation/Grouping--E Step

- Given parameters, the probability that a given point is associated with each cluster is can be computed by:

$$p(m \mid \mathbf{x}_l, \Theta^{(s)}) = \frac{\alpha_m^{(s)} p(\mathbf{x}_l \mid \theta_m^{(s)})}{\sum_{k=1}^{M} \alpha_k^{(s)} p(\mathbf{x}_l \mid \theta_k^{(s)})}$$

Where does that come from?

- The book uses $\overline{I_{lm}}$ for $p(m \mid \mathbf{x}_i, \Theta^{(s)})$ (*I* suggests “indicator variable”)

- (Also, my copy of the book’s version of the above equation looks wrong to me--the index *l* applies to points and the index for theta should refer to groups)

Details optional

$\alpha_m^{(s)} = p(m)$ (Standard notation)

$$p(m \mid \mathbf{x}_l, \Theta^{(s)}) = \frac{p(\mathbf{x}_l \mid m, \theta_m^{(s)}) p(m)}{p(\mathbf{x}_l \mid \theta_m^{(s)})}$$ (Bayes)

$$p(\mathbf{x}_l \mid \theta_m^{(s)}) = \sum_{k=1}^{M} p(\mathbf{x}_l, m, \theta_k^{(s)})$$ (Marginalization)

$$p(\mathbf{x}_l \mid \theta_m^{(s)}) = \sum_{k=1}^{M} p(m) p(\mathbf{x}_l \mid m, \theta_k^{(s)})$$ (Definition of "|")

Therefore

$$p(m \mid \mathbf{x}_l, \Theta^{(s)}) = \frac{\alpha_m^{(s)} p(\mathbf{x}_l \mid \theta_m^{(s)})}{\sum_{k=1}^{M} \alpha_k^{(s)} p(\mathbf{x}_l \mid \theta_k^{(s)})}$$

We can do the egg!

Segmentation/Grouping by EM

- This is a lot like K-means

- Instead of binary cluster membership, each point has some probability of being in each cluster

- In addition to computing means, we generally also compute variances
  - Setting all variances equal in advance is simplest, but not so useful
  - Can assume all variances are the same (“tied”)
  - Can fit different variances to each cluster (most common)
  - Can fit covariance matrices instead of variances (usually not possible if the dimension is over five or so)

Segmentation with EM

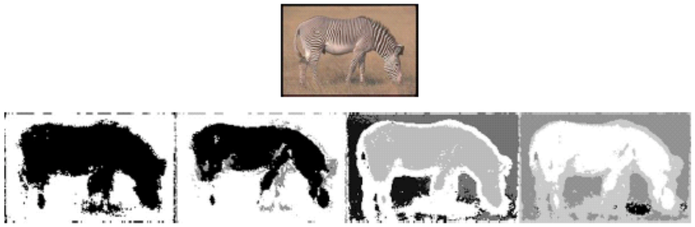


Figure from “Color and Texture Based Image Segmentation Using EM and Its Application to Content Based Image Retrieval”,S.J. Belongie et al., Proc. Int. Conf. Computer Vision, 1998, c1998, IEEE

Motion segmentation with EM (one)

- Recall the baby on the couch
- Alternative algorithms based on previous examples?

Motion segmentation with EM (one)

- Can treat background/foreground assignment as missing values!

Motion segmentation with EM (two)

- Model image sequence as consisting of regions (layers) of parametric motion
  - For example, affine motion is popular

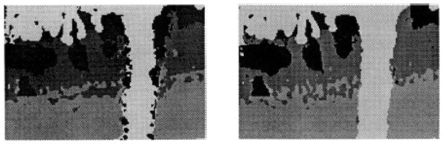
$$\begin{pmatrix} v_x \\ v_y \end{pmatrix} = \begin{pmatrix} a & b \\ c & d \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} + \begin{pmatrix} t_x \\ t_y \end{pmatrix}$$

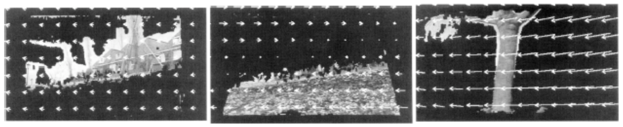
- Now we need to
  - Determine which pixels belong to which region
  - Estimate parameters
  - Yet another example of a missing value problem!

Three frames from the MPEG “flower garden” sequence

Figure from “Representing Images with layers,”, by J. Wang and E.H. Adelson, IEEE Transactions on Image Processing, 1994, c 1994, IEEE

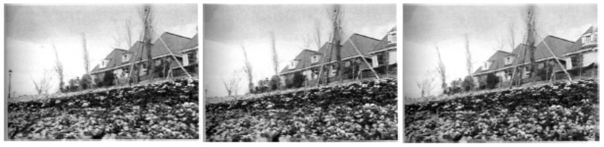
Grey level shows region no. with highest probability





Segments and motion fields associated with them

Figure from “Representing Images with layers,”, by J. Wang and E.H. Adelson, IEEE Transactions on Image Processing, 1994, c 1994, IEEE



If we use multiple frames to estimate the appearance of a segment, we can fill in occlusions; so we can re-render the sequence with some segments removed.

Figure from “Representing Images with layers,”, by J. Wang and E.H. Adelson, IEEE Transactions on Image Processing, 1994, c 1994, IEEE

RANSAC versus EM

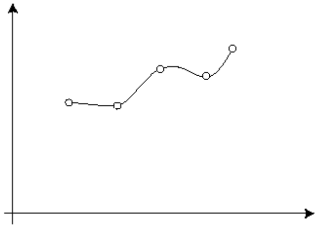
- Many, but not all problems that can be attacked with EM can also be attacked with RANSAC
  - For RANSAC, we need to be able to get a parameter estimate with a manageably small number of random choices.
  - RANSAC is often better

Important

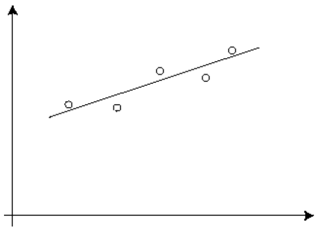
Model Selection

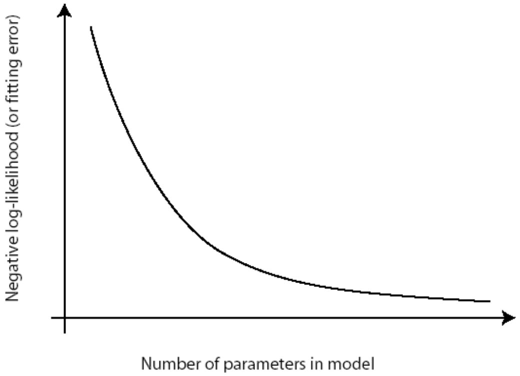
- In general, models with more parameters will fit a dataset better, but are poorer at prediction
- This means we can't simply look at the negative log-likelihood (or fitting error)

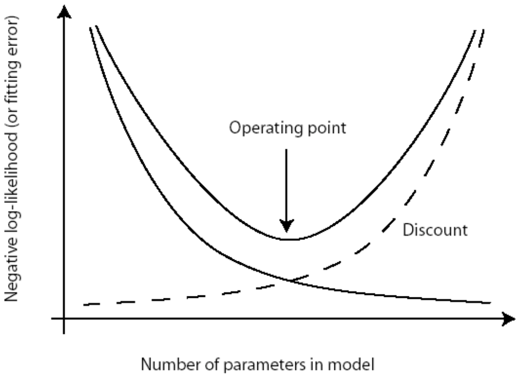
Important



Top is not necessarily a better fit than bottom
(actually, almost always worse)







We can discount the fitting error with some term in the number of parameters in the model.

Discounts

- Let N be the number of data points, p the number of parameters
- AIC (an information criterion)
  - choose model with smallest value of

$$-2L\left(D;\theta^{*}\right)+2p$$

- BIC (Bayes information criterion)
  - choose model with smallest value of

$$-2L\left(D;\theta^{*}\right)+p\log N$$

- Minimum description length
  - same criterion as BIC, but derived in a completely different way

Important

Cross-validation

- Split data set into two pieces, fit to one, and compute negative log-likelihood on the **other**
- One set is “training data”, the other is “testing data” or “held out data”
- Average over different splits
- This estimates the quality of your model
  - Often (rightfully so) used to compare algorithms
- If you are doing model selection, then you choose the model with the smallest value of this average
  - This works because adding parameters causes over fitting of the training data which gives worse performance on test data

Skipped in 2008

Model averaging

- Often smarter to use multiple models for prediction than just one
- Consider that we have various models that we believe to various degrees, denoted by $P(M_i)$
- Suppose we want to estimate X from data, D, via the group of models, M_i
- A Bayesian would compute

$$P(X \mid D)=\sum_{i} P(X \mid M_i,D)P(M_i \mid D)$$

Recognition by finding patterns

- Template matching with correlation (linear filters) is a simple example of recognition by pattern matching
- Some objects behave like quite simple templates
  - Frontal faces

Recognition by finding patterns

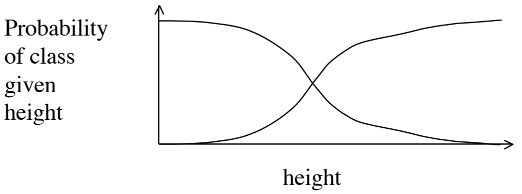
- Example strategy:
  - Find image windows
  - Correct for lighting
  - Pass them to a statistical test (a classifier) that accepts faces and rejects non-faces
- Important high level point:
  - Need to understand relationship between **modeling statistics** and deciding between options (classification AND risk analysis).

Basic ideas in classification

- Concrete example
  - “guess” male / female from height

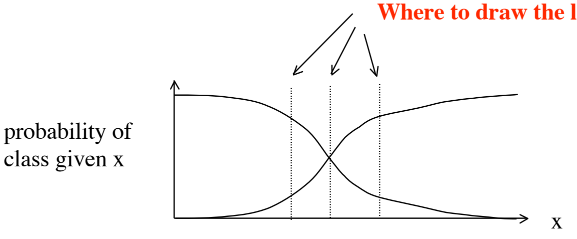
Basic ideas in classification

- Concrete example
  - “guess” male / female from height
- Probabalistic approach
  - Consider P(female|height)

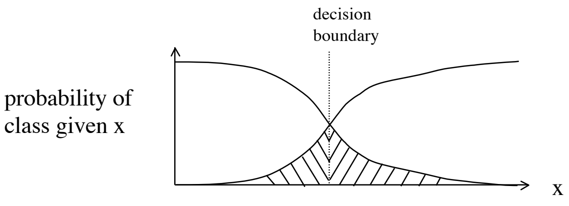


Basic ideas in classification

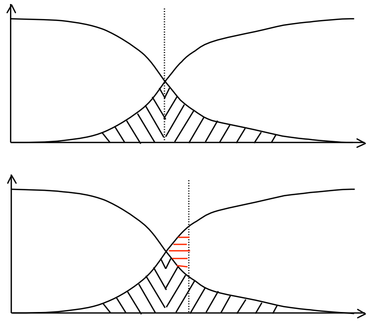
Where to draw the line?



Basic ideas in classification



Area of intersection under curves gives expected value of making a mistake



Red shows extra that you get wrong with different boundary

Basic ideas in classification

- Concrete example
  - “guess” male / female from height
- Probabalistic approach
  - Consider P(female|height)
- Now consider “risk”
  - Suppose you want to give vaccine based on height for a disease that only males get.
  - There is great benefit to males who may be exposed
  - Vaccines have risk as well as benefit
  - Thus there is also some risk to giving females a vaccine they do not need
- How does this change the boundary?

Loss / Risk

- Some errors may be more expensive than others
  - e.g. a fatal disease that is easily cured by a cheap medicine with minimal side-effects --> false positives in diagnosis are better than false negatives
- We want to set the classification point
- Consider two class classification
  - Let L(1->2) be the loss caused by calling a 1 a 2
  - Want to anaylize the **expected value** of the loss (risk)

Basic ideas in classification

- Expected loss (risk) of using classifier *s*

$$R(s) = \Pr(1 \to 2 \mid \text{using s}) L(1 \to 2) + \Pr(2 \to 1 \mid \text{using s}) L(2 \to 1)$$

Details of formula optional, but the idea is worth understanding

Basic ideas in classification

- Generally, we should classify as 1 if the expected loss of classifying as 1 is better than for 2
- We get

$$\begin{aligned} &1 \text{ if} \quad \Pr(1 \mid x) L(1 \to 2) > \Pr(2 \mid x) L(2 \to 1) \\ &2 \text{ if} \quad \Pr(2 \mid x) L(2 \to 1) > \Pr(1 \mid x) L(1 \to 2) \end{aligned}$$

- Crucial notion: Decision boundary
  - points where the loss is the same for either case

Details of formula optional, but the idea is worth understanding

Basic ideas in classification

- Expected loss (risk) of using classifier *s*

$$R(s) = L(1 \to 2) \bullet P(1 \to 2)(s) + L(2 \to 1) \bullet P(2 \to 1)(s)$$

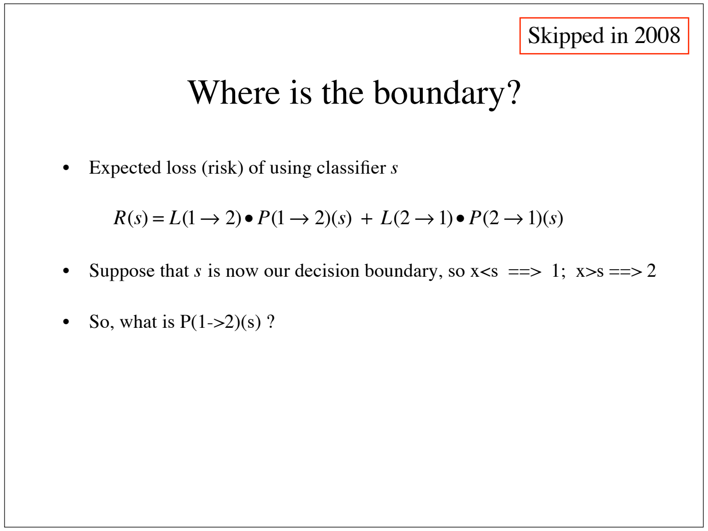
Skipped in 2008

Basic ideas in classification

- Expected loss (risk) of using classifier *s*

$$R(s) = L(1 \to 2) \bullet P(1 \to 2)(s) + L(2 \to 1) \bullet P(2 \to 1)(s)$$

- So, given the class conditional densities, how do we set the boundary?



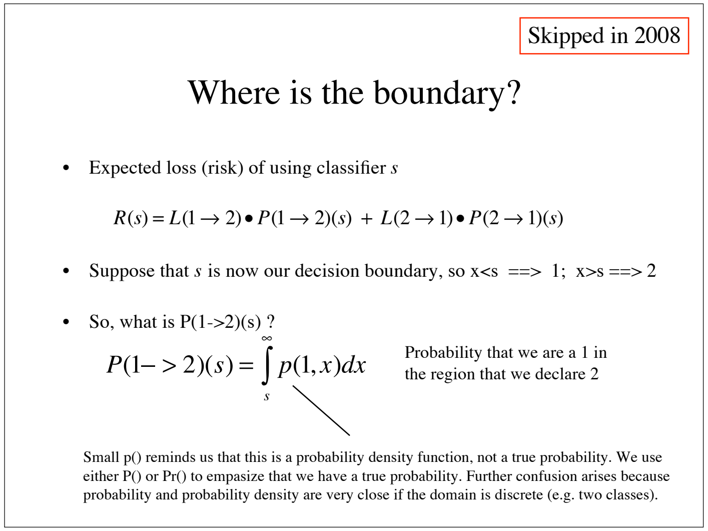
Skipped in 2008

Where is the boundary?

- Expected loss (risk) of using classifier *s*

$$R(s) = L(1 \rightarrow 2) \bullet P(1 \rightarrow 2)(s) \; + \; L(2 \rightarrow 1) \bullet P(2 \rightarrow 1)(s)$$

- Suppose that *s* is now our decision boundary, so x<s ==> 1; x>s ==> 2
- So, what is P(1->2)(s) ?



Skipped in 2008

Where is the boundary?

- Expected loss (risk) of using classifier *s*

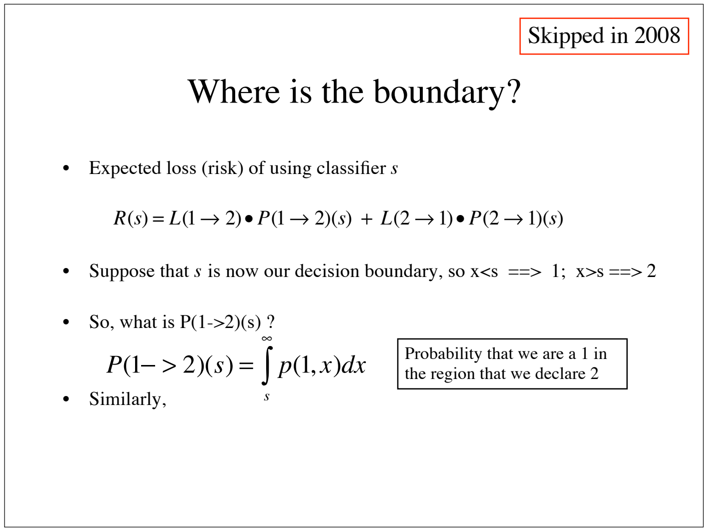
$$R(s) = L(1 \rightarrow 2) \bullet P(1 \rightarrow 2)(s) \; + \; L(2 \rightarrow 1) \bullet P(2 \rightarrow 1)(s)$$

- Suppose that *s* is now our decision boundary, so x<s ==> 1; x>s ==> 2
- So, what is P(1->2)(s) ?

$$P(1->2)(s) = \int_s^\infty p(1,x)dx$$

Probability that we are a 1 in the region that we declare 2

Small p() reminds us that this is a probability density function, not a true probability. We use either P() or Pr() to empasize that we have a true probability. Further confusion arises because probability and probability density are very close if the domain is discrete (e.g. two classes).



Skipped in 2008

Where is the boundary?

- Expected loss (risk) of using classifier *s*

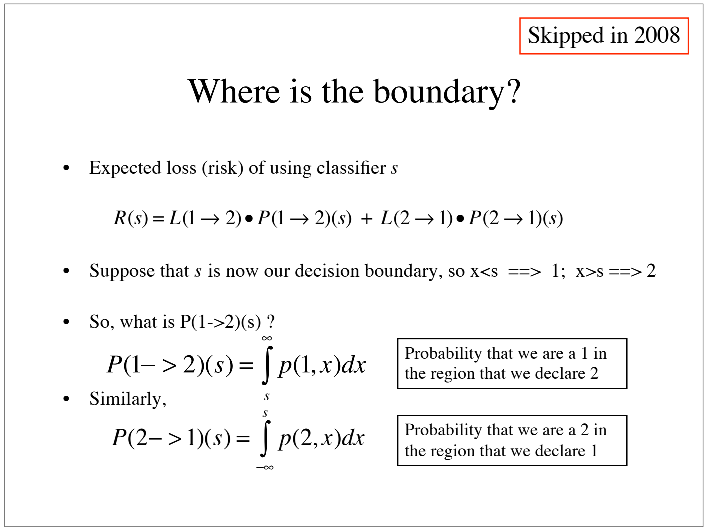
$$R(s) = L(1 \rightarrow 2) \bullet P(1 \rightarrow 2)(s) \; + \; L(2 \rightarrow 1) \bullet P(2 \rightarrow 1)(s)$$

- Suppose that *s* is now our decision boundary, so x<s ==> 1; x>s ==> 2
- So, what is P(1->2)(s) ?

$$P(1->2)(s) = \int_s^\infty p(1,x)dx$$

Probability that we are a 1 in the region that we declare 2

- Similarly,



Skipped in 2008

Where is the boundary?

- Expected loss (risk) of using classifier *s*

$$R(s) = L(1 \rightarrow 2) \bullet P(1 \rightarrow 2)(s) \; + \; L(2 \rightarrow 1) \bullet P(2 \rightarrow 1)(s)$$

- Suppose that *s* is now our decision boundary, so x<s ==> 1; x>s ==> 2
- So, what is P(1->2)(s) ?

$$P(1->2)(s) = \int_s^\infty p(1,x)dx$$

Probability that we are a 1 in the region that we declare 2

- Similarly,

$$P(2->1)(s) = \int_{-\infty}^s p(2,x)dx$$

Probability that we are a 2 in the region that we declare 1

Skipped in 2008

Where is the boundary?

- Expected loss (risk) is then

$$R(s) \;=\; \int_{s}^{\infty} L(1->2)\,p(1,x)dx \;+\int_{0}^{s} L(2->1)\,p(2,x)dx$$

- We want to minimize this

Skipped in 2008

Where is the boundary?

- Expected loss (risk) is then

$$R(s) \;=\; \int_{s}^{\infty} L(1->2)\,p(1,x)dx \;+\int_{0}^{s} L(2->1)\,p(2,x)dx$$

- We want to minimize this. So differentiate and set to 0.

$$\frac{d}{ds}\int_{s}^{\infty} L(1->2)\,p(1,x)dx \;=\; -L(1->2)\,p(1,s)$$

$$\frac{d}{ds}\int_{-\infty}^{s} L(2->1)\,p(2,x)dx \;=\; L(2->1)\,p(2,s)$$

(follows from the definition of integration and differentiation)

Skipped in 2008

Where is the boundary?

- Expected loss (risk) is then

$$R(s) \;=\; \int_{s}^{\infty} L(1->2)\,p(1,x)dx \;+\int_{0}^{s} L(2->1)\,p(2,x)dx$$

- We want to minimize this. Using previous pieces:

$$\frac{d}{ds}R(s) = L(2->1)\,p(2,s) - L(1->2)\,p(1,s)$$

- Setting to zero reveals the boundary for minimal risk:

$$L(2->1)\,p(2,s) = L(1->2)\,p(1,s)$$

Skipped in 2008

Basic ideas in classification

- Put differently

$$\begin{array}{ll} 1 \text{ if} & P(1|\,\mathrm{x})L(1 \rightarrow 2) > \mathrm{P}(2|\,\mathrm{x})L(2 \rightarrow 1) \\ 2 \text{ if} & P(2|\,\mathrm{x})L(2 \rightarrow 1) > \mathrm{P}(1|\,\mathrm{x})L(1 \rightarrow 2) \end{array}$$

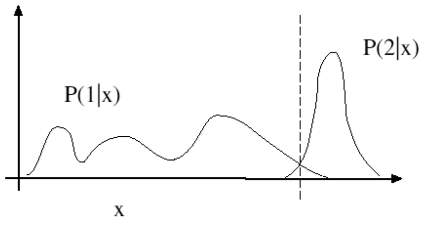
- (Switching to conditional probability is OK here)

- Crucial intuitive notion: Decision boundary is at the points where the loss is the same for either case

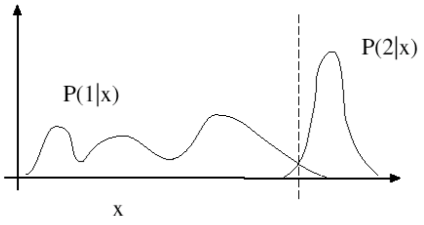
Building classifiers

- 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

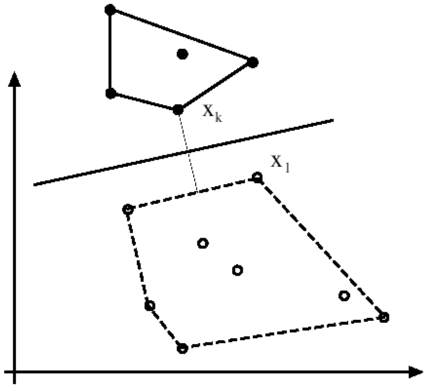
- Standard scenario
  - Have training data
  - Want to classify new data
- One approach
  - Estimate the probability distributions (we have being thinking about them all along, e.g. P(1|x))
  - Issue: parameter estimates that are “good” may not give optimal classifiers
- Another approach
  - Directly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

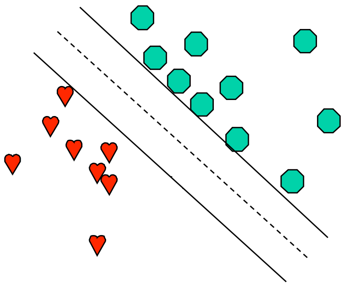
Support vector machines

- Linearly separable data means that we can chose

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

- 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:

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

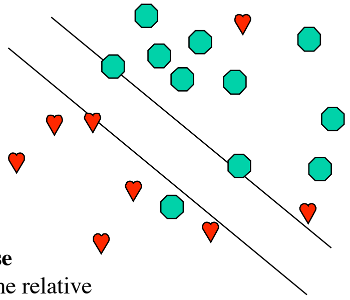
Support vector machines

- Solved by

$$\begin{aligned} &\text{minimize} \quad (1/2)\boldsymbol{w} \cdot \boldsymbol{w} \\ &\text{subject to} \quad y_i\,(\boldsymbol{w} \cdot \boldsymbol{x}_i + b) \geq 1 \end{aligned}$$

- (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

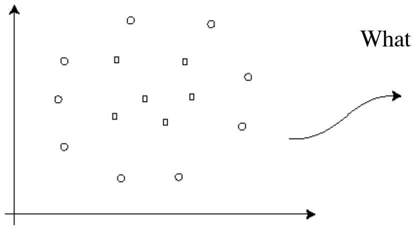


Non-separable case
Cost, C, specifies the relative desire to push the planes apart, verses the number of mistakes.

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)

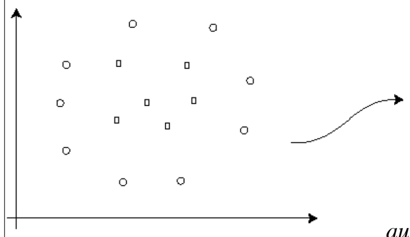
Key observation: The SVM is completely a function of dot products between the vectors.

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.

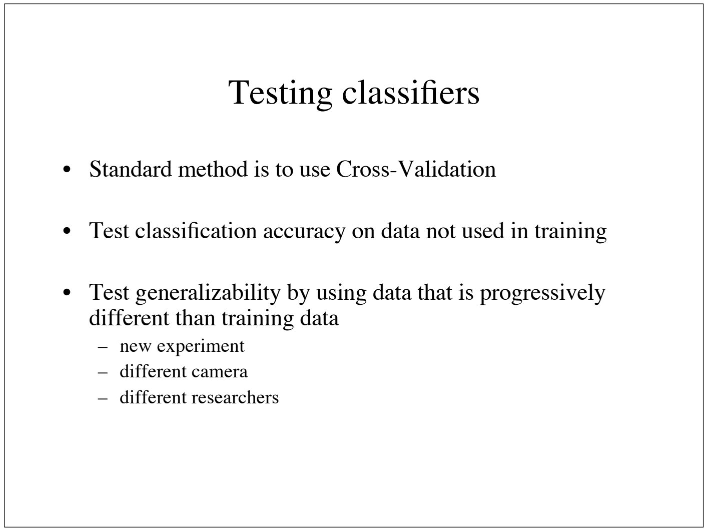
Support vector machines (kernel tricks)

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



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