Symbols, Patterns and Signals: Classification I

Dr. Iván Palomares Carrascosa, Office 319 Magg's House

Email: i.palomares@bristol.ac.uk

Web: http://dsrs.blogs.Bristol.ac.uk

Back to SPS Outline

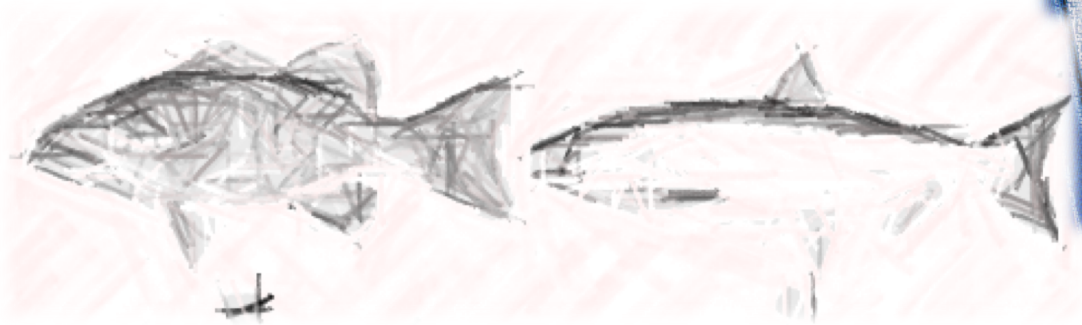
Weeks	*Monday Lecture*	*Wednesday Lecture*	*Labs*	*Thursday Lecture*	*Assessments*
13	Data, Data Modelling and Estimation (I)	Data, Data Modelling and Estimation (II)	Intro to Jupiter Notebook I	-	-
14	Problem Class - Data Acquisition	Data Modelling and Estimation (III)	Intro to Jupiter Notebook II	Problem Class - Deterministic Data Modelling	CW1 (set)
15	Data, Data Modelling and Estimation (IV)	Data, Data Modelling and Estimation (V)	Least Squares	Problem Class - Probabilistic Data Modelling	-
16	Review part I	Classification I	Maximum Likelihood	Classification II	-
17	Clustering I	Problem Class	Fitting	Clustering II	CW1 (deadline)
18	**Computer Science Explore Week**				-
19	Gaussian Mixture Methods	Evaluation Methods	Classification	Problem Class	CW2 (set)
20	Review Part II	Features: Representing Your Data	Evaluating Classifiers	Feature Transform	-
21	Feature Transform	Problem Class: Feature Transformation	Feature Selection	Feature Extraction: removing redundancy	-
22	Feature Extraction: removing redundancy	Problem Class: Feature Extraction	-	Feature Dependency and Graphical Models	-
Easter Break					
23	Feature Dependency and Graphical Models	Review part III	-	Review Part I	CW2 (deadline)
24	Bank Holiday	-	-		

←WE START THE GREEN BLOCK TODAY!

https://uob-coms21202.github.io/COMS21202.github.io/

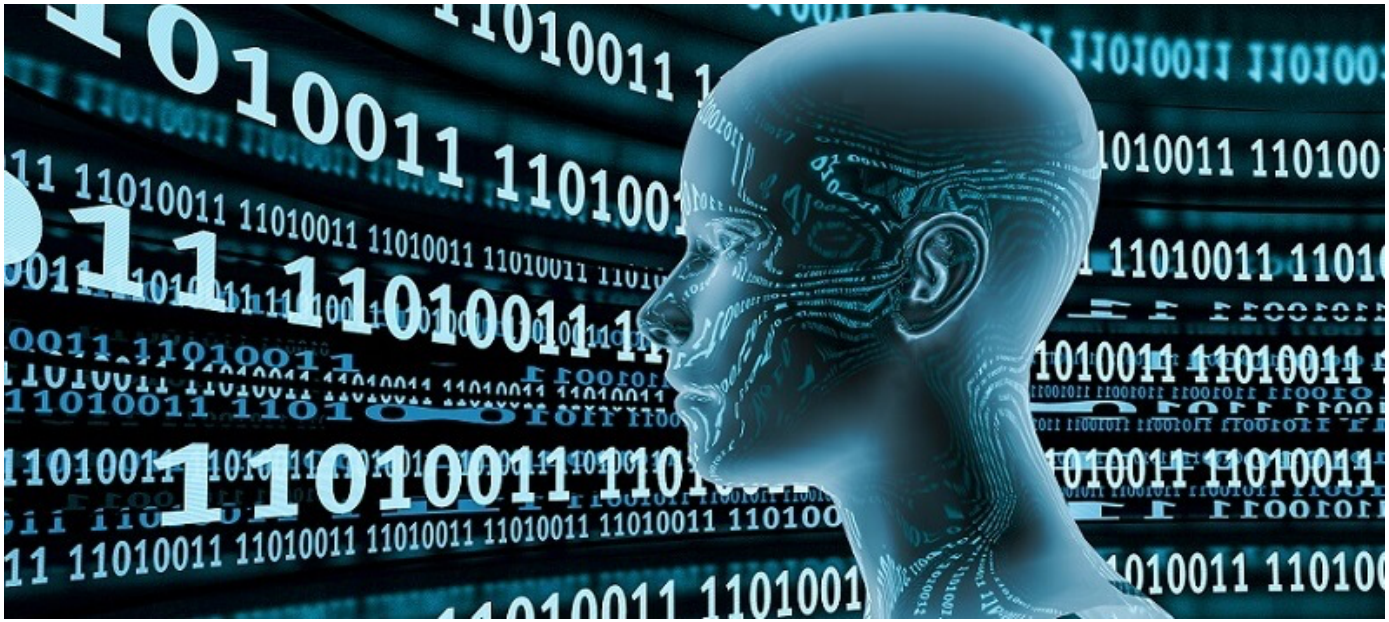
In this lecture…

- What is Machine Learning
- What is the goal of classification
- Formulating simple decision rules for classification
- Naïve-Bayes classification
- Laplace correction



Have you ever heard of something called …

…Machine Learning?



ML is an application of artificial intelligence (AI) that gives computer systems the ability to automatically ***learn by themselves from data and improve from experience****, without being explicitly programmed.*

Have you ever heard of something called …

…Machine Learning?

Classification and recognition (SUPERVISED LEARNING)

- assigning data observations to classes, predicting the unknown class value for a new observation
- Recognizing spam emails or identifying telescope images which show a planet

Clustering and segmentation (UNSUPERVISED LEARNING)

- No classes associated to observations → putting similar observations together
- Finding plagiarised coursework submissions, segmenting a market into subgroups of similar customers

Estimation and detection

- measuring/detecting things from the given observations
- tracking football players in video or determining location in buildings

Classification: definition

Given

- **Observations:** a set of instances (e.g., fish specimens, or email messages)
- **Attributes:** described by feature vectors $\mathbf{x}^T = [x_1 \ldots x_d]$ (numeric or symbolic),
- and a small set of **classes** $\omega_1, \omega_2, \ldots$ (e.g., salmon/seabass, spam/non-spam)

a classifier divides the d-dimensional instance space into regions of (almost) uniform class

- In other words, the data set of instances is divided into subsets, each one associated to one of the possible class values ω_k
- there are many techniques to construct classifiers
- if we only look at the given data, any method is as good as any other
- what we really want is a classifier that generalizes well to future, unseen data
  - deterministic approaches tend to over-fit the data

Classification Example

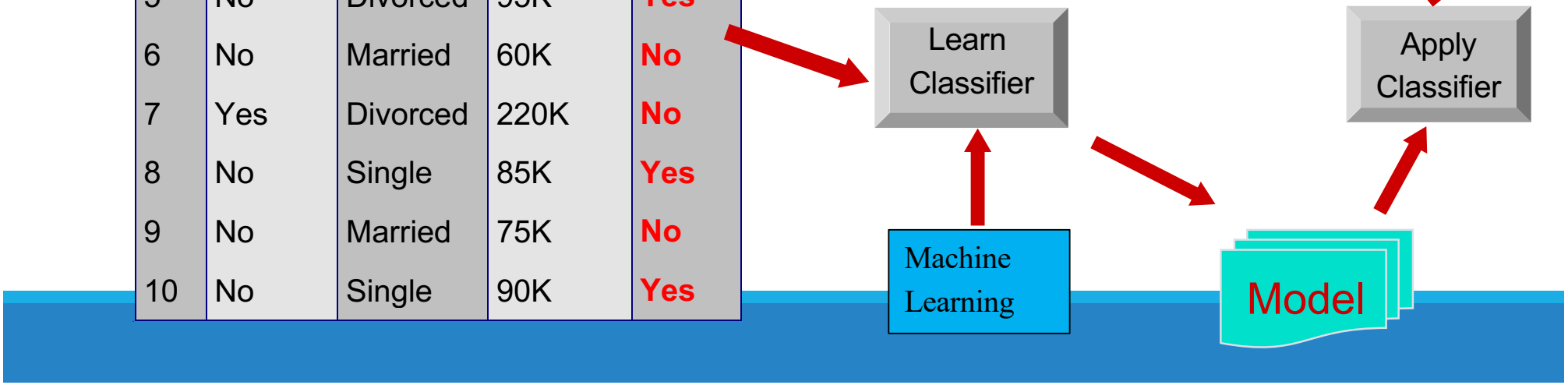
TRAINING SET (labelled)

categorical | categorical | continuous | class

Tid	Refund	Marital Status	Taxable Income	Cheat
1	Yes	Single	125K	No
2	No	Married	100K	No
3	No	Single	70K	No
4	Yes	Married	120K	No
5	No	Divorced	95K	Yes
6	No	Married	60K	No
7	Yes	Divorced	220K	No
8	No	Single	85K	Yes
9	No	Married	75K	No
10	No	Single	90K	Yes

TEST SET (unlabelled)

Refund	Marital Status	Taxable Income	Cheat
No	Single	75K	?
Yes	Married	50K	?
No	Married	150K	?
Yes	Divorced	90K	?
No	Single	40K	?
No	Married	80K	?



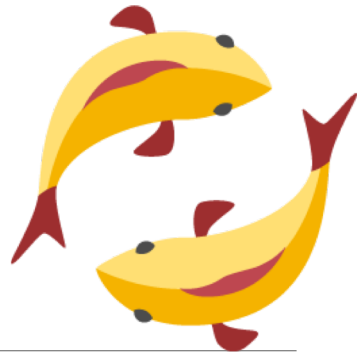
Classification Techniques

- **Decision Tree based Methods**
- Rule-based Methods
- Neural Networks
- **Naïve Bayes** and Bayesian Belief Networks
- Support Vector Machines
- …

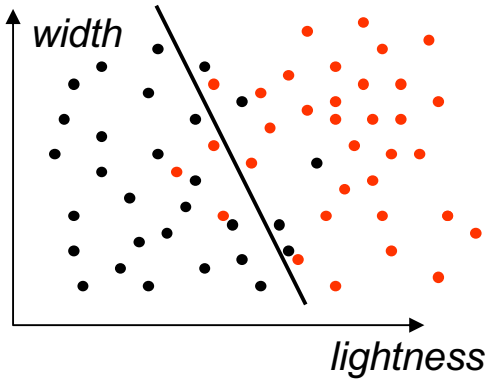
NOTE: Not all classification techniques rely on explicitly building a model (classifier)

(e.g. Naïve Bayes in this lecture doesn't, but decision trees in next lecture do!)

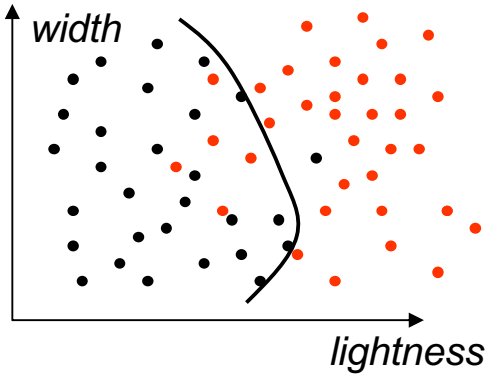
Classification Example



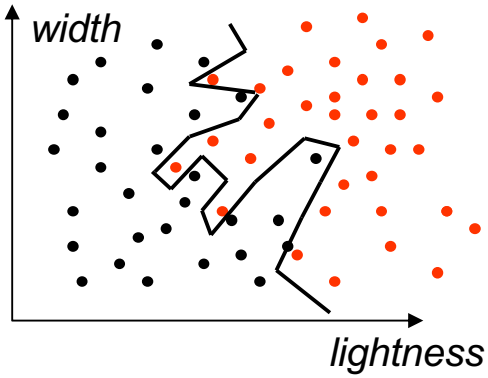
Decision boundary = points of class change = points of class uncertainty



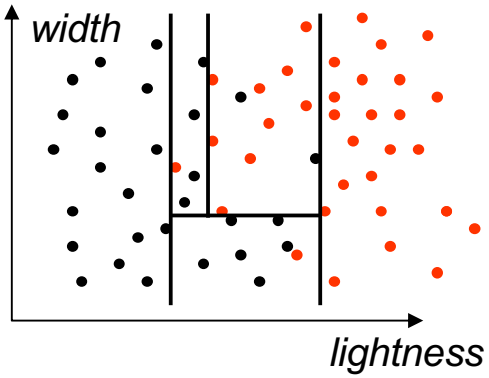
linear decision boundary



non-linear decision boundary



piecewise linear decision boundary



axis-parallel decision boundaries

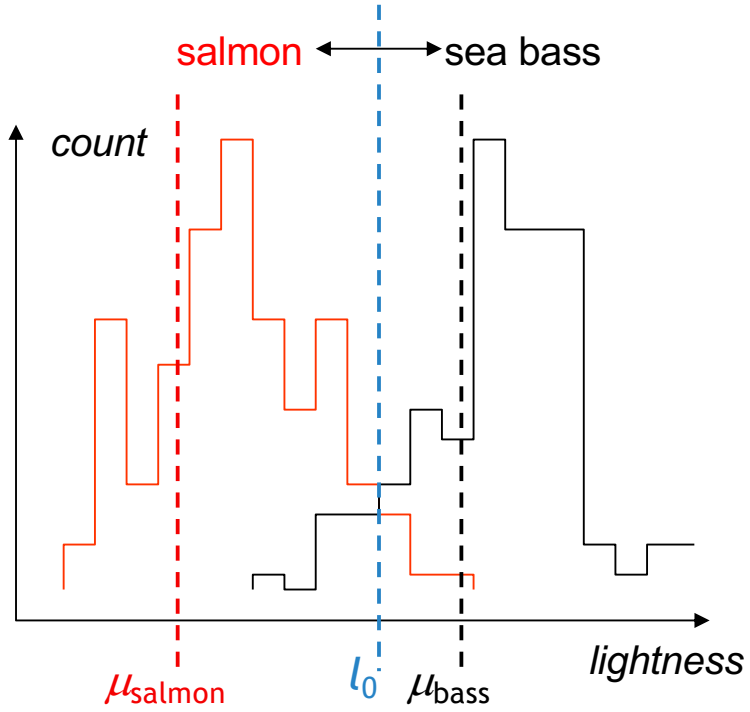
Classification by counting

Involves obtaining counts, histograms etc. from data, and using these to obtain a decision rule

- Example: classifying fish *(salmon* vs *sea bass)* in terms of *lightness* attribute
- Decision rule: **If** lightness ≥ l0 **then** sea bass **else** salmon

- How to set threshold l_0?
  i. Simple idea that works in this case: calculate the means of the *lightness* distributions for salmon and sea bass
     - i.e., centroids of histograms
  ii. … and take the mean of those:

$$l_0 = (\mu_{\text{bass}} + \mu_{\text{salmon}}) / 2$$

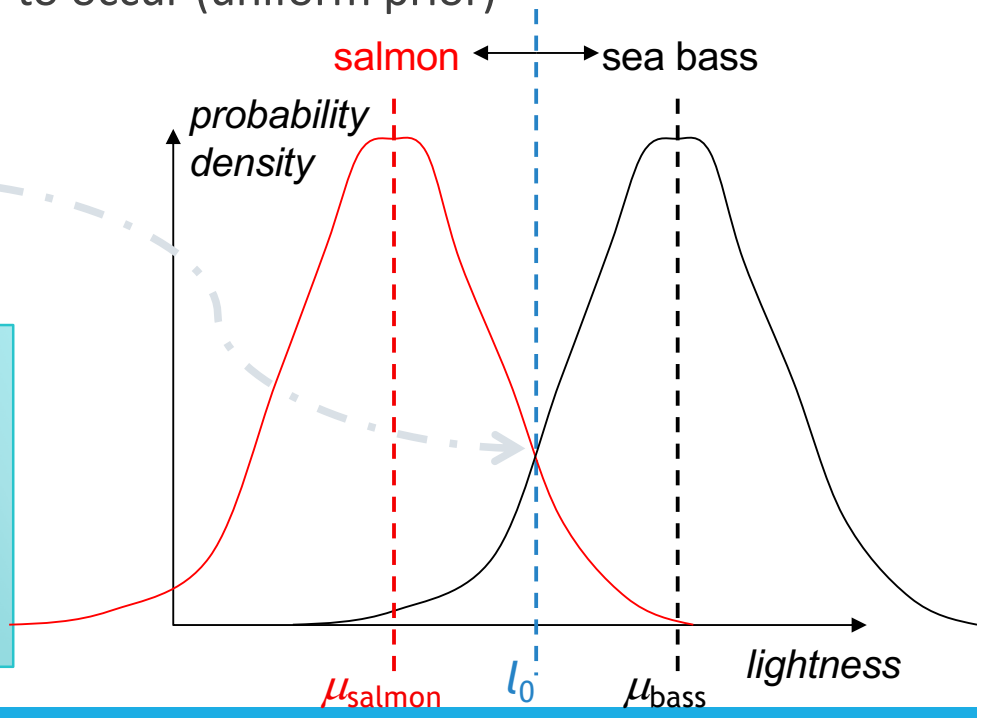


Does this classifier generalize well?

Yes: it results in the fewest errors on unseen fish, assuming that:

- lightness is the only feature we have
- **p(lightness|salmon)** and **p(lightness|sea bass)** are normally distributed, with different means but equal variance σ^2
- sea bass and salmon are equally likely to occur (uniform prior)

$p(l_0 | \text{salmon}) = p(l_0 | \text{sea bass})$



- Note that these **probability densities** are **conditional on the class**
- They are called **likelihoods**
  - i.e., we condition something we can observe (here: lightness) on something hypothetical (here: class)

Coming up with a **decision rule**

I. Describe the data

- single numeric feature → **lightness**
- two classes: **salmon** *vs* **sea bass**

II. Select a classification model

- lightness is normally distributed for each class, with different means but equal variance
- classes are equally likely

III. Estimate the parameters

- calculate means and variance (if required) for each class

IV. Formulate decision rule

- **if** p(lightness|sea bass) ≥ p(lightness|salmon) **then** sea bass **else** salmon
  - In this case independent of variance so can assume $\sigma^2=1$

More on decision rules

The following decision rules are equivalent:

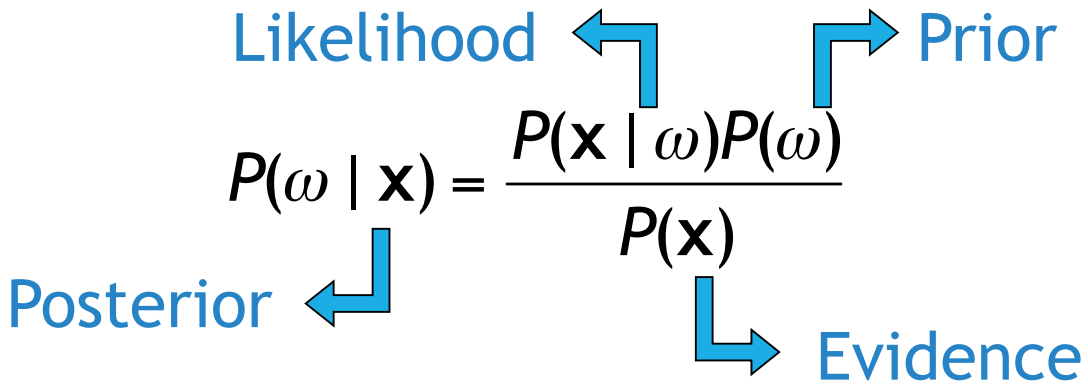
- if **p(lightness|sea bass) ≥ p(lightness|salmon)** then sea bass else salmon (*maximum likelihood, ML*)
- if $\frac{p(\text{lightness} \mid \text{sea bass})}{p(\text{lightness} \mid \text{salmon})} \geq 1$ **then** sea bass **else** salmon (*likelihood ratio*)
- $\arg\max_{\omega \in \{\text{bass,salmon}\}} p(\text{lightness} \mid \omega)$ (works for more than two different classes)

With non-uniform prior probabilities (class probabilities) we should use

- if **p(lightness|sea bass)P(sea bass) ≥ p(lightness|salmon)P(salmon)** then sea bass else salmon (*maximum a posteriori or MAP*)
- **if** $\frac{p(\text{lightness} \mid \text{sea bass})}{p(\text{lightness} \mid \text{salmon})} \geq \frac{P(\text{salmon})}{P(\text{sea bass})}$ **then** sea bass **else** salmon
- $\arg\max_{\omega \in \{\text{bass,salmon}\}} p(\text{lightness} \mid \omega) P(\omega)$

Bayes Theorem revisited

Bayes' theorem is a simple property of conditional probabilities, but has important applications in prediction (classification) and decision making



Given obs. *x*, what can we say about class ω?

- given x (e.g., feature vector with properties of fish), choose ω (e.g., class) that maximises posterior probability: **argmax_ω P(ω|x)**
- since P(x) is independent of ω, we can ignore it and choose class ω that maximizes the likelihood, reweighted by the prior: **argmax_ω P(x|ω)P(ω)**
- in case of a uniform prior, this can be simplified to maximum likelihood: **argmax_ω P(x|ω)**

From prior to posterior probability

The prior probability P(ω) tells us how likely each of the classes is "a priori"

- without looking at it, would you predict the next email to be spam or non-spam?

The posterior probability P(ω|x) tells us how likely each of the classes is after observing instance x

- Reminder: **P(ω|x) = P(ω,x)/P(x),** i.e., changing the reference set from all instances to those described by x
  - P(male)=0.49, P(female)=0.51 → **PRIOR**
  - P(male|John)=0.99, P(female|John)=0.01 → **POSTERIOR**
  - P(male|Hilary)=0.28, P(female|Hilary)=0.72 → **POSTERIOR**

Bayes' theorem tells us how to calculate the posterior, given (i) the prior and (ii) the likelihood

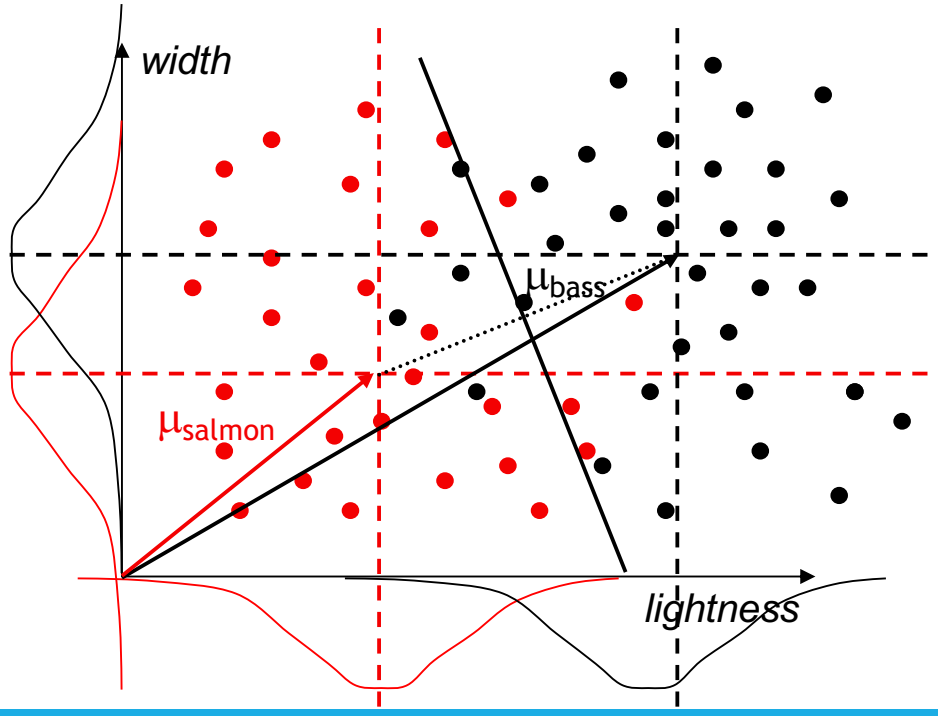
- Evidence acts as normalizing factor, can often be ignored

Two independent features

- Now we need to estimate two class-conditional mean vectors: μ_{salmon} and μ_{bass}
- We can again ignore the variances if they are equal for both features and for both classes
  - i.e., assume covariance matrix for each class is I

ML decision rule:
Given a (lightness,width) fish observation:

if p(lightness|sea bass)p(width|sea bass) ≥ p(lightness|salmon)p(width|salmon) **then** sea bass **else** salmon

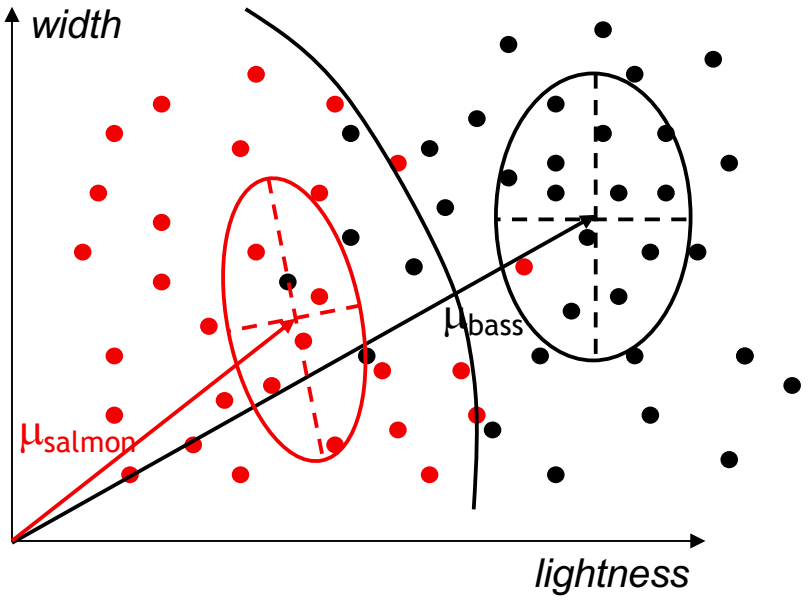


If features are not independent

- In the general case we need to estimate mean vectors as well as full covariance matrices for each class
- This may result in a non-linear decision boundary if covariance matrices are different across classes

ML decision rule:
Given a (lightness,width) fish observation:

if p(lightness, width|sea bass) ≥ p(lightness, width|salmon) **then** sea bass **else** salmon



The Naive-Bayes classifier

"Naively" assumes independent features within each class:

$$P(\mathbf{x} \mid \omega) = P\left(\begin{bmatrix} x_1 \\ \vdots \\ x_d \end{bmatrix} \mid \omega\right)$$

- **unconditional independence:** knowledge about one feature does not tell us anything about the others
- **class-conditional independence:** within each class, knowledge about one feature does not tell us anything about the others

$$\approx P(x_1 \mid \omega)P(x_2 \mid \omega)\ldots P(x_d \mid \omega) = \prod_{i=1}^{d} P(x_i \mid \omega)$$

Now the MAP (*Maximum A Posteriori*) decision rule becomes

$$\arg\max_{\omega} P(\omega \mid \mathbf{x}) = \arg\max_{\omega} P(\mathbf{x} \mid \omega)P(\omega) \approx \arg\max_{\omega} \left(\prod_{i=0}^{d} P(x_i \mid \omega) \right) P(\omega)$$

Example decision boundaries

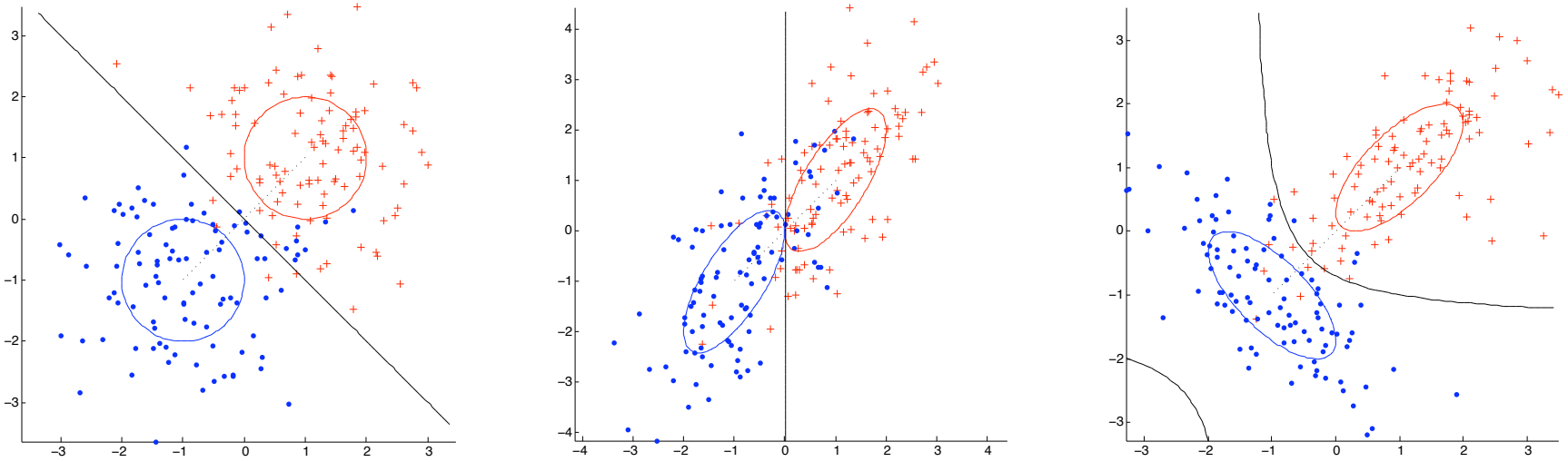


Figure 9.3. **(left)** If the features are uncorrelated and have the same variance, maximum-likelihood classification leads to the basic linear classifier, whose decision boundary is orthogonal to the line connecting the means. **(middle)** As long as the per-class covariance matrices are identical, the Bayes-optimal decision boundary is linear – if we were to decorrelate the features by rotation and scaling, we would again obtain the basic linear classifier. **(right)** Unequal covariance matrices lead to hyperbolic decision boundaries, which means that one of the decision regions is non-contiguous.

NB classifier also works for symbolic features!

From a training set of spam and non-spam emails, select subset of n words w_i $(1 \le i \le n)$ for vocabulary. Then, estimate the likelihoods $P(w_i|spam)$ and $P(w_i|\neg spam)$

- note: symbolic (boolean) features indicate whether *i-th* word appears or not in an e-mail message → **Bernoulli random variable**
- Likelihoods estimated from relative frequencies in training set
- Best features have big difference in likelihoods for spam and ¬spam

We (naively) assume class-conditional independence of word occurrences, so the likelihoods for a particular email become

- $P(email|spam) = P(w_1|spam)P(w_2|spam) \ldots P(w_n|spam)$
- $P(email|\neg spam) = P(w_1|\neg spam)P(w_2|\neg spam) \ldots P(w_n|\neg spam)$

New email is spam if :

- $P(email|spam) \cdot P(spam) \geq P(email|\neg spam) \cdot P(\neg spam)$

Spam email example

Example 9.4 (Prediction using a naive Bayes model). Suppose our vocabulary contains three words a, b and c, and we use a multivariate Bernoulli model for our e-mails, with parameters

$$\boldsymbol{\theta}^{+} = (0.5, 0.67, 0.33) \qquad \boldsymbol{\theta}^{-} = (0.67, 0.33, 0.33)$$

This means, for example, that the presence of b is twice as likely in spam (+), compared to ham.

The e-mail to be classified contains words a and b but not c, and hence is described by the bit vector $\mathbf{x} = (1, 1, 0)$. We obtain likelihoods

$$P(\mathbf{x}|+) = 0.5 \cdot 0.67 \cdot (1 - 0.33) = 0.222 \qquad P(\mathbf{x}|-) = 0.67 \cdot 0.33 \cdot (1 - 0.33) = 0.148$$

The ML classification of $\mathbf{x}$ is thus spam. In the case of two classes it is often convenient to work with likelihood ratios and odds. The likelihood ratio can be calculated as $\frac{P(\mathbf{x}|+)}{P(\mathbf{x}|-)} = \frac{0.5}{0.67}\frac{0.67}{0.33}\frac{1-0.33}{1-0.33} = 3/2 > 1$. This means that the MAP classification of $\mathbf{x}$ is also spam if the prior odds is more than 2/3, but ham if it is less than that. For example, with 33% spam and 67% ham the prior odds is $\frac{P(+)}{P(-)} = \frac{0.33}{0.67} = 1/2$, resulting in a posterior odds of $\frac{P(+|\mathbf{x})}{P(-|\mathbf{x})} = \frac{P(\mathbf{x}|+)}{P(\mathbf{x}|-)}\frac{P(+)}{P(-)} = 3/2 \cdot 1/2 = 3/4 < 1$. In this case the likelihood ratio for $\mathbf{x}$ is not strong enough to push the decision away from the prior.

Spam email example (cont.)

Example 9.5 (Training a naive Bayes model). We now show how the parameter vectors in the previous example might have been obtained. Consider the following e-mails consisting of five words a, b, c, d, e:

e_1: $b\ d\ e\ b\ b\ d\ e$
e_2: $b\ c\ e\ b\ b\ d\ d\ e\ c\ c$
e_3: $a\ d\ a\ d\ e\ a\ e\ e$
e_4: $b\ a\ d\ b\ e\ d\ a\ b$

e_5: $a\ b\ a\ b\ a\ b\ a\ e\ d$
e_6: $a\ c\ a\ c\ a\ c\ a\ e\ d$
e_7: $e\ a\ e\ d\ a\ e\ a$
e_8: $d\ e\ d\ e\ d$

We are told that the e-mails on the left are spam and those on the right are ham. So we decide to use these as a small training set to train our Bayesian classifier. First, we decide that d and e are so-called *stop words* that are too common to convey class information. The remaining words, a, b and c, constitute our vocabulary.

In the multivariate Bernoulli model e-mails are represented by bit vectors, as in Table 9.1 (right). Adding the bit vectors for each class results in $(2,3,1)$ for spam and $(3,1,1)$ for ham. Each count is to be divided by the number of documents in a class, in order to get an estimate of the probability of a document containing a particular vocabulary word. Probability smoothing now means to add two pseudo-documents, one containing each word and one containing none of them. This results in the estimated parameter vectors $\hat{\theta}^+ = (3/6, 4/6, 2/6) = (0.5, 0.67, 0.33)$ for spam and $\hat{\theta}^- = (4/6, 2/6, 2/6) = (0.67, 0.33, 0.33)$ for ham.

E-mail	a?	b?	c?	Class
e_1	0	1	0	+
e_2	0	1	1	+
e_3	1	0	0	+
e_4	1	1	0	+
e_5	1	1	0	−
e_6	1	0	1	−
e_7	1	0	0	−
e_8	0	0	0	−

Spam email example (cont.)

Example 9.5 (Training a naive Bayes model). We now show how the parameter vectors in the previous example might have been obtained. Consider the following e-mails consisting of five words a, b, c, d, e:

e_1: $b\ d\ e\ b\ b\ d\ e$
e_2: $b\ c\ e\ b\ b\ d\ d\ e\ c\ c$
e_3: $a\ d\ a\ d\ e\ a\ e\ e$
e_4: $b\ a\ d\ b\ e\ d\ a\ b$

e_5: $a\ b\ a\ b\ a\ b\ a\ e\ d$
e_6: $a\ c\ a\ c\ a\ c\ a\ e\ d$
e_7: $e\ a\ e\ d\ a\ e\ a$
e_8: $d\ e\ d\ e\ d$

We are told that the e-mails on the left are spam and those on the right are ham. So we decide to use these as a small training set to train our Bayesian classifier. First, we decide that d and e are so-called *stop words* that are too common to convey class information. The remaining words, a, b and c, constitute our vocabulary.

In the multivariate Bernoulli model e-mails are represented by bit vectors, as in Table 9.1 (right). Adding the bit vectors for each class results in $(2,3,1)$ for spam and $(3,1,1)$ for ham. Each count is to be divided by the number of documents in a class, in order to get an estimate of the probability of a document containing a particular vocabulary word. Probability smoothing now means to add two pseudo-documents, one containing each word and one containing none of them. This results in the estimated parameter vectors $\hat{\theta}^+ = (3/6, 4/6, 2/6) = (0.5, 0.67, 0.33)$ for spam and $\hat{\theta}^- = (4/6, 2/6, 2/6) = (0.67, 0.33, 0.33)$ for ham.

E-mail	a?	b?	c?	Class
e_1	0	1	0	+
e_2	0	1	1	+
e_3	1	0	0	+
e_4	1	1	0	+
e_5	1	1	0	−
e_6	1	0	1	−
e_7	1	0	0	−
e_8	0	0	0	−
f9	0	0	0	+/-
f10	1	1	1	+/-

Smoothing
= Laplace correction
= adding pseudo-counts
= MAP estimation
(see next slide)

Smoothing - Laplace correction

- Useful if we need to estimate, e.g., $\mathbf{P(w_i|spam)}$ for a words w_i that does not occur in spam emails (i.e. an attribute value that *never* appears in observations of a given class)
- Unsmoothed relative frequencies would give $\mathbf{P(w_i=0|spam)=n/n=1}$ and $\mathbf{P(w_i=1|spam)=0/n=0}$, where **n** is the number of spam emails.
  BIG PROBLEM! → $\mathbf{P(email|spam)=0}$ for *any* email that contains $\mathbf{w_i}$

SOLUTION: For an event that has *k* possible outcomes, add *1* pseudo-count for each outcome.

- Laplace correction (**k=2** as $\mathbf{w_i}$ has 2 possible outcomes) gives $\mathbf{P(w_i=1|spam)=1/(n+2)}$ and $\mathbf{P(w_i=0|spam)=(n+1)/(n+2)}$, hence smoothed likelihoods will never be 0 or 1.