

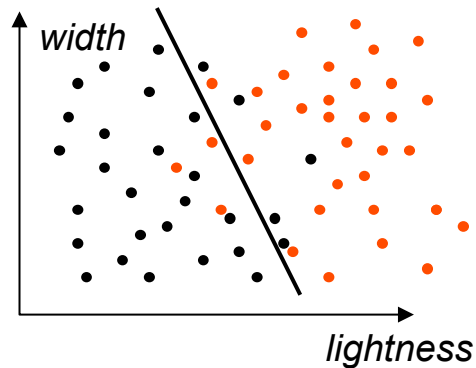
Things to do to data

- **Classification and recognition**
 - assigning things to classes, e.g. recognising spam emails or images containing a sunset
- **Clustering and segmentation**
 - putting similar things together, e.g. regions in images or plagiarised coursework submissions
- **Estimation and detection**
 - measuring things, e.g. tracking football players in video or determining location in buildings

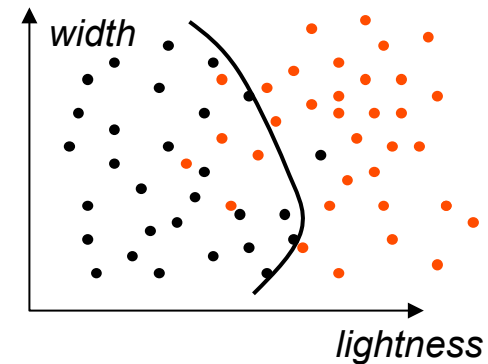
Classification: definition

- **Given**
 - a set of **instances** (e.g., fish or email messages)
 - described by feature vectors $\mathbf{x}^T = [x_1 \dots x_d]$ (numeric or symbolic),
 - and a small set of **classes** $\omega_1, \omega_2, \dots$ (e.g., **salmon**/sea bass, spam/non-spam)
- **a classifier divides the d -dimensional instance space into regions of (almost) uniform class**
 - there are many ways 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 **generalises** well to future, **unseen** data
 - deterministic approaches tend to **over-fit** the data

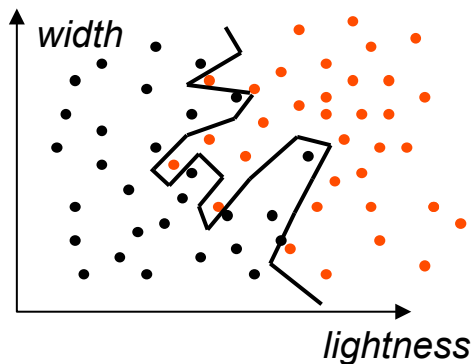
Examples



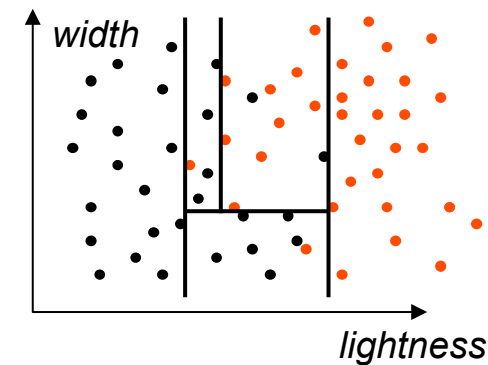
linear decision boundary



non-linear decision boundary



piecewise linear decision boundary



axis-parallel decision boundaries

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

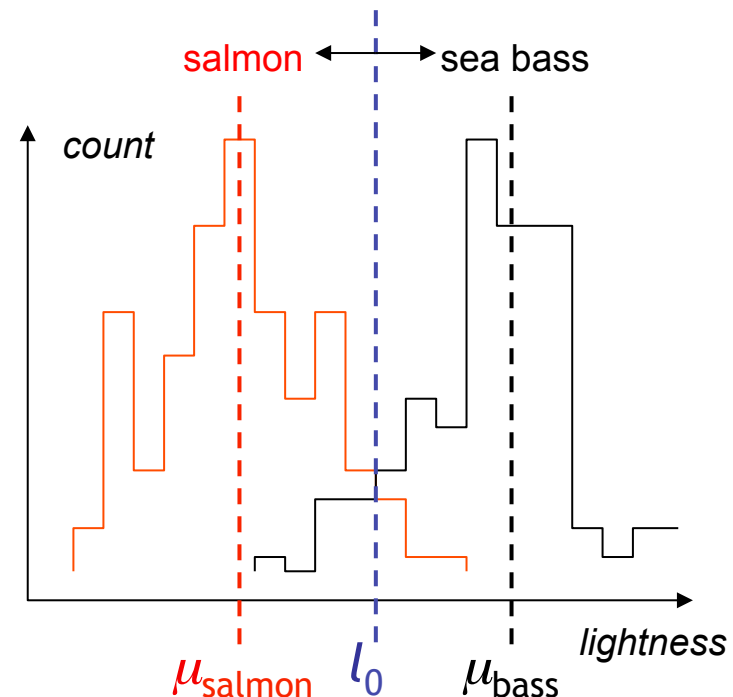
Classification by counting

- Involves obtaining counts, histograms etc. from data, and use these to obtain a **decision rule**
 - example: classifying fish in terms of lightness
 - decision rule: If lightness $\geq l_0$ then **sea bass** else **salmon**

- How to set threshold l_0 ?

- Simple idea that works in this case: calculate the means of the lightness distributions for **salmon** and sea bass
 - i.e., centroids of histograms
- and take the mean of those:

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

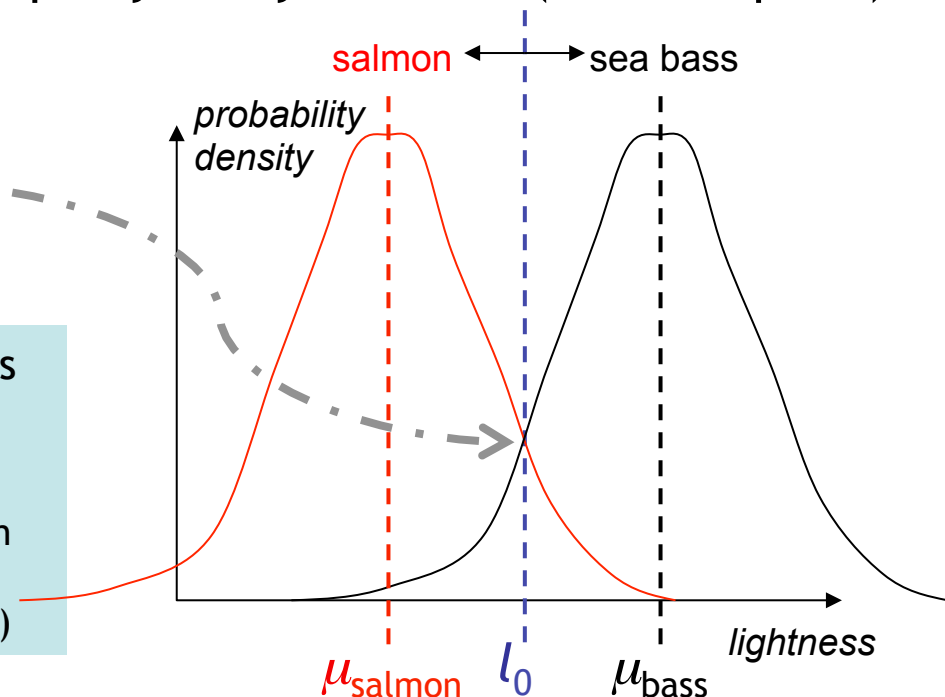


Does this classifier generalise?

- Yes: it results in the fewest errors on unseen fish, under the following assumptions:
 - lightness is the only feature we have
 - $p(\text{lightness} | \text{salmon})$ and $p(\text{lightness} | \text{sea bass})$ 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)



Let's do that again

1. Describe the data

- single numeric feature **lightness**
- two classes **salmon** and **sea bass**

2. Select a model

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

3. Estimate the parameters

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

4. Formulate decision rule


- if $p(\text{lightness} | \text{sea bass}) \geq p(\text{lightness} | \text{salmon})$ 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 $p(\text{lightness} | \text{sea bass}) \geq p(\text{lightness} | \text{salmon})$ then **sea bass** else **salmon** (**maximum likelihood** or **ML**)
 - if $\frac{p(\text{lightness} | \text{sea bass})}{p(\text{lightness} | \text{salmon})} \geq 1$ then **sea bass** else **salmon** (**likelihood ratio**)
 - $\arg \max_{\omega \in \{\text{bass}, \text{salmon}\}} p(\text{lightness} | \omega)$ (works for more than two classes)
- With non-uniform priors we should use
 - if $p(\text{lightness} | \text{sea bass})P(\text{sea bass}) \geq p(\text{lightness} | \text{salmon})P(\text{salmon})$ then **sea bass** else **salmon** (**maximum a posteriori** or **MAP**)
 - if $\frac{p(\text{lightness} | \text{sea bass})}{p(\text{lightness} | \text{salmon})} \geq \frac{P(\text{salmon})}{P(\text{sea bass})}$ then **sea bass** else **salmon**
 - $\arg \max_{\omega \in \{\text{bass}, \text{salmon}\}} p(\text{lightness} | \omega)P(\omega)$

Ask Rev Bayes why it works

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

Likelihood  **Prior**

$$P(\omega | \mathbf{x}) = \frac{P(\mathbf{x} | \omega)P(\omega)}{P(\mathbf{x})}$$

Posterior   **Evidence**

Given \mathbf{x} , what can we say about ω ?

- given \mathbf{x} (e.g., feature vector with properties of fish), choose ω (e.g., class) that maximises posterior probability: $\operatorname{argmax}_{\omega} P(\omega | \mathbf{x})$
- since $P(\mathbf{x})$ is independent of ω , we can ignore it and choose ω that maximises the likelihood, reweighted by the prior: $\operatorname{argmax}_{\omega} P(\mathbf{x} | \omega)P(\omega)$
- in case of a uniform prior, this can be simplified to maximum likelihood: $\operatorname{argmax}_{\omega} P(\mathbf{x} | \omega)$

From prior to posterior

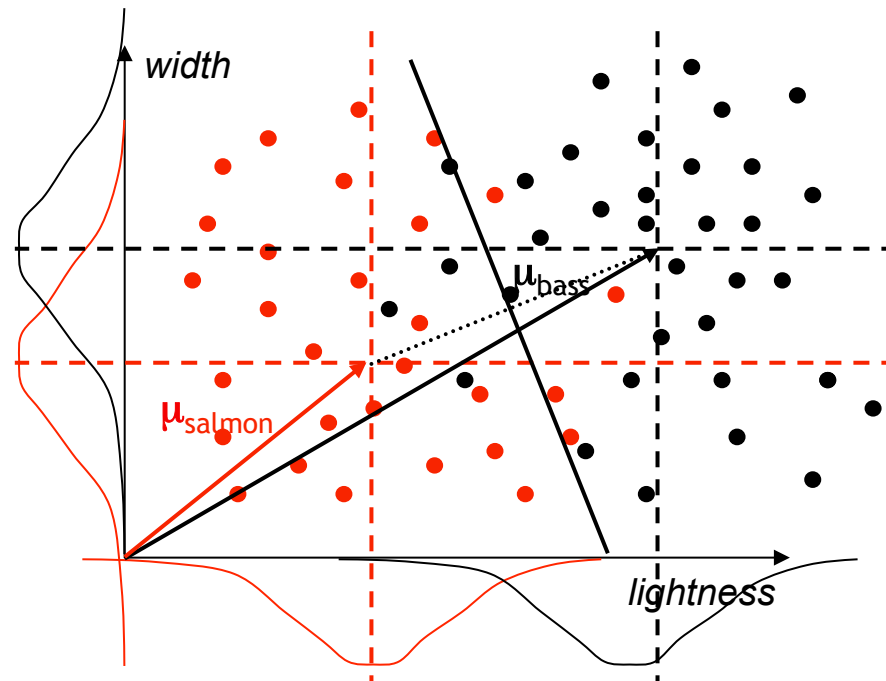
- The **prior probability** $P(\omega)$ tells us how likely each of the classes is *a priori*
 - without looking at it, would you predict the next email to be spam or non-spam?
- The **posterior probability** $P(\omega | \mathbf{x})$ tells us how likely each of the classes is after observing instance \mathbf{x}
 - Reminder: $P(\omega | \mathbf{x}) = P(\omega, \mathbf{x}) / P(\mathbf{x})$, i.e., changing the reference set from all instances to those described by \mathbf{x}
 - $P(\text{male})=0.49$, $P(\text{female})=0.51$
 - $P(\text{male} | \text{john})=0.99$, $P(\text{female} | \text{john})=0.01$
 - $P(\text{male} | \text{hilary})=0.38$, $P(\text{female} | \text{hilary})=0.62$
- Bayes' theorem tells us how to calculate the posterior from the prior and the likelihood
 - Evidence acts as normalising 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:

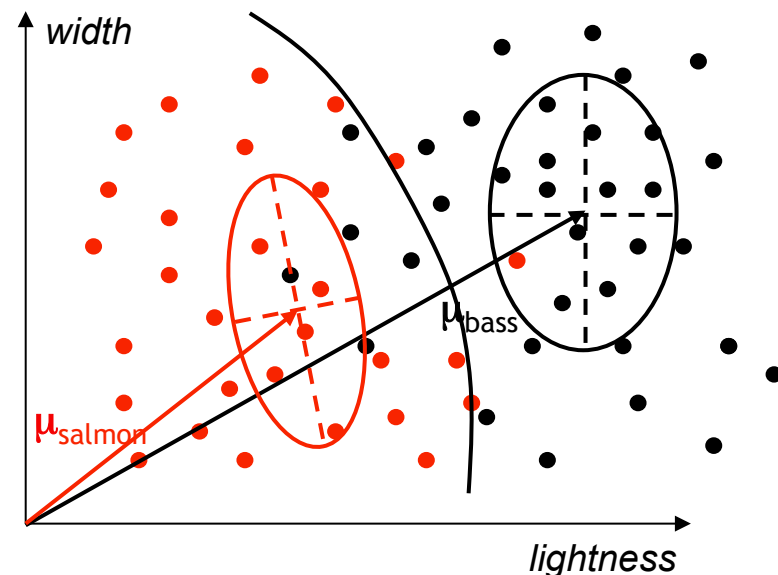
if $p(\text{lightness} | \text{sea bass})p(\text{width} | \text{sea bass})$
 $\geq p(\text{lightness} | \text{salmon})p(\text{width} | \text{salmon})$
then sea bass else salmon



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:
if $p(\text{lightness}, \text{width} | \text{sea bass})$
 $\geq p(\text{lightness}, \text{width} | \text{salmon})$
then sea bass else salmon



Example decision boundaries

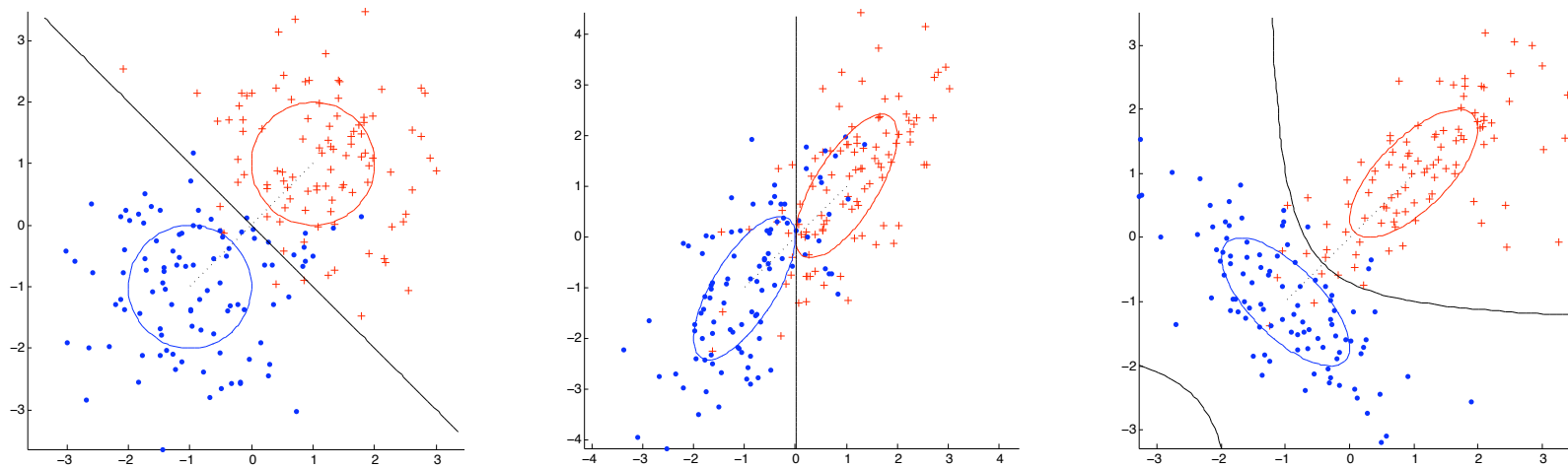


Figure 9.3. **(left)** If the features are uncorrelated and have the same variance, maximum-likelihood classification leads to the basic linear classifier, whose decision boundary is orthogonal to the line connecting the means. **(middle)** As long as the per-class covariance matrices are identical, 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.

The naive Bayesian 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

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

- Now the MAP decision rule becomes

$$\arg \max_{\omega} P(\omega \mid \mathbf{x}) = \arg \max_{\omega} P(\mathbf{x} \mid \omega)P(\omega) \approx \arg \max_{\omega} \left(\prod_{i=1}^d P(x_i \mid \omega) \right) P(\omega)$$

Also works for symbolic features!



- From **training set** of spam and non-spam emails, select subset of words w_i ($1 \leq i \leq n$) for vocabulary, and estimate the likelihoods $P(w_i | \text{spam})$ and $P(w_i | \neg \text{spam})$
 - note: symbolic (boolean) features, indicating whether i -th word occurs in email or not \rightarrow **Bernoulli random variable**
 - estimated from relative frequencies in training set
 - good features have big difference in likelihoods for spam and \neg spam
- We (naively) assume **class-conditional independence** of word occurrences, so the likelihoods for a particular email become
 - $P(\text{email} | \text{spam}) = P(w_1 | \text{spam})P(w_2 | \text{spam}) \dots P(w_n | \text{spam})$
 - $P(\text{email} | \neg \text{spam}) = P(w_1 | \neg \text{spam})P(w_2 | \neg \text{spam}) \dots P(w_n | \neg \text{spam})$
- **New email is spam if :**
 - $P(\text{email} | \text{spam})P(\text{spam}) \geq P(\text{email} | \neg \text{spam})P(\neg \text{spam})$

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

$$\theta^+ = (0.5, 0.67, 0.33) \quad \theta^- = (0.67, 0.33, 0.33)$$

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

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

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	−

Smoothing

= Laplace correction

= adding pseudo-counts

= MAP estimation

Reminder: MAP estimation

- Suppose we want to express our prior belief that coins are typically fair

- can be done by assuming a prior $p(\theta)$ that peaks around $\theta=1/2$ – convenient here is $p(\theta) \propto \theta(1-\theta)$ (not Gaussian!)

1. $P(D | \theta) = a\theta^D(1-\theta)^{N-D}, p(\theta) = b\theta(1-\theta)$

a and b are independent of θ , hence don't matter!

2. $\ln P(D | \theta)p(\theta) = \ln a + D \ln \theta + (N-D) \ln(1-\theta) + \ln b + \ln \theta + \ln(1-\theta)$

3. $\frac{d}{d\theta} \ln P(D | \theta)p(\theta) = \frac{D}{\theta} - \frac{N-D}{1-\theta} + \frac{1}{\theta} - \frac{1}{1-\theta}$

4. $\frac{D+1}{\theta_{MAP}} - \frac{N-D-1}{1-\theta_{MAP}} = 0 \rightarrow \theta_{MAP} = \frac{D+1}{N+2}$

- Thus, this prior adds two ‘virtual coin tosses’, one coming up heads, the other coming up tails

Laplace correction

- For an event that has k possible outcomes, add 1 pseudo-count for each outcome.
 - Useful if we need to estimate, e.g., $P(w_i | \text{spam})$ for words that don't occur in spam emails.
 - Unsmoothed relative frequencies would give $P(w_i=0 | \text{spam})=n/n=1$ and $P(w_i=1 | \text{spam})=0/n=0$, where n is the number of spam emails.
Hence $P(\text{email} | \text{spam})=0$ for any email that contains w_i .
 - Laplace correction ($k=2$ as w_i has 2 possible outcomes) gives $P(w_i=0 | \text{spam})=1/(n+2)$ and $P(w_i=1 | \text{spam})=(n+1)/(n+2)$, hence smoothed likelihoods will never be 0 or 1.
- The effect is more pronounced for smaller n (number of trials).

Model and assumptions

- The model here is that, for each class, the occurrence of each vocabulary word in the email is modelled as an independent random variable (multivariate Bernoulli)
 - One parameter to estimate for each word and class
- Simplifying assumptions
 - Using only word occurrences of fixed vocabulary
 - ignores words that don't occur in vocabulary
 - doesn't treat words that occur more than once differently (can be relaxed by using multinomial model)
 - ignores word order ('bag of words')
 - Assuming class-conditional independence of word occurrences
 - often leads to over/under-estimating joint probabilities
 - cannot handle cases in which words correlate positively with the class if taken separately, but negatively if taken jointly