Statistical learning

Chapter 20 of "AI: a Modern Approach", Sections 1–3
Presented by: Jicheng Zhao

Goal

♢ Learn probabilistic theories of the world from experience

♢ We focus on the learning of Bayesian networks

♢ More specifically, input **data** (or **evidence**), learn probabilistic theories of the world (or **hypotheses**)

Outline

♢ Bayesian learning ⇐

♢ Approximate Bayesian learning
  – Maximum *a posteriori* learning (MAP)
  – Maximum likelihood learning (ML)

♢ Parameter learning with complete data
  – ML parameter learning with complete data in **discrete** models
  – ML parameter learning with complete data in **continuous** models (linear regression)
  – Naive Bayes models
  – Bayesian parameter learning

♢ Learning Bayes net structure with complete data

(If time allows)

♢ Learning with hidden variables or incomplete data (EM algorithm)

Full Bayesian learning

View learning as Bayesian updating of a probability distribution over the hypothesis space

H is the hypothesis variable, values $h_1, h_2, \ldots$, prior $\mathbf{P}(H)$

jth observation d_j gives the outcome of random variable D_j
training data $\mathbf{d} = d_1, \ldots, d_N$

Given the data so far, each hypothesis has a posterior probability:

$$P(h_i|\mathbf{d}) = \alpha P(\mathbf{d}|h_i)P(h_i)$$

where $P(\mathbf{d}|h_i)$ is called the likelihood

Predictions use a likelihood-weighted average over all hypotheses:

$$\mathbf{P}(X|\mathbf{d}) = \sum_i \mathbf{P}(X|\mathbf{d}, h_i)P(h_i|\mathbf{d}) = \sum_i \mathbf{P}(X|h_i)P(h_i|\mathbf{d})$$

No need to pick one best-guess hypothesis!

Example

Suppose there are five kinds of bags of candies:

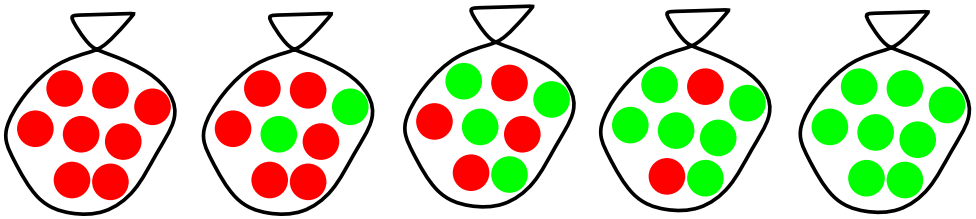
10% are h_1: 100% cherry candies

20% are h_2: 75% cherry candies + 25% lime candies

40% are h_3: 50% cherry candies + 50% lime candies

20% are h_4: 25% cherry candies + 75% lime candies

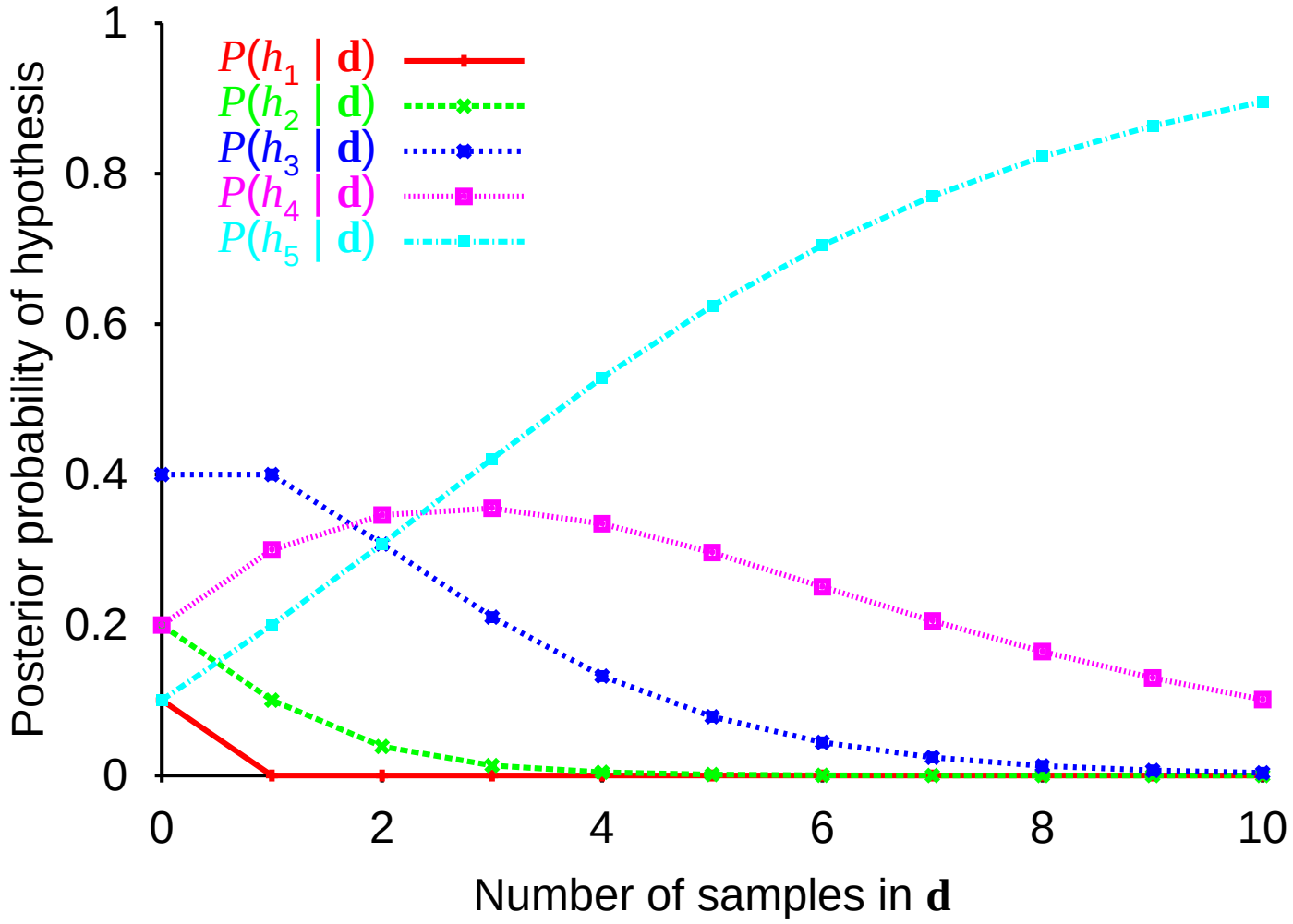
10% are h_5: 100% lime candies



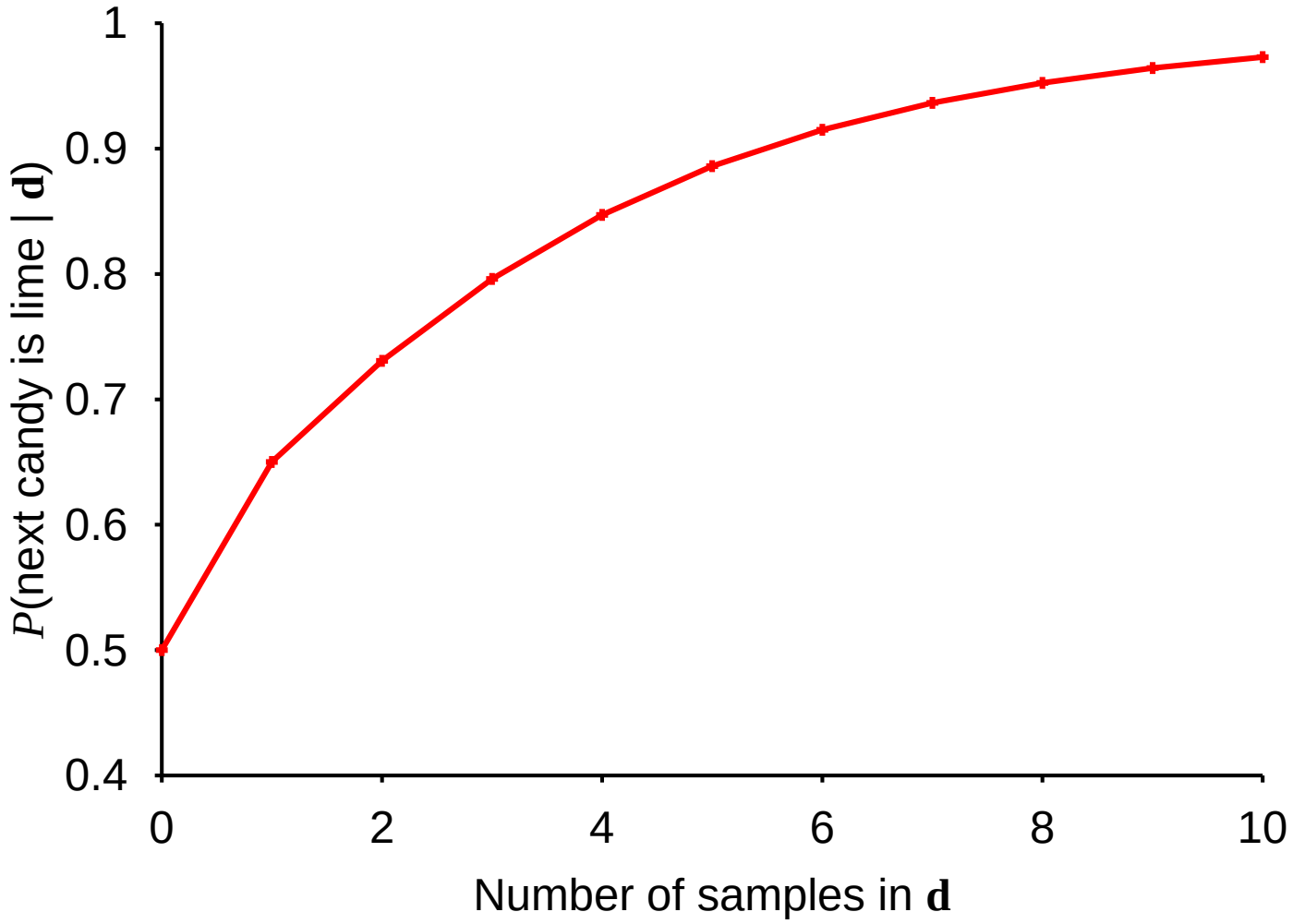
Then we observe candies drawn from some bag: 

What kind of bag is it? What flavour will the next candy be?

Posterior probability of hypotheses



Prediction probability



Properties of full Bayesian learning

1. *The true hypothesis eventually dominates the Bayesian prediction* given that the true hypothesis is in the prior
2. The Bayesian prediction is *optimal*, whether the data set be small or large [?]

On the other hand

1. The hypothesis space is usually very large or infinite
   summing over the hypothesis space is often intractable
   (e.g., 18,446,744,073,709,551,616 Boolean functions of 6 attributes)

MAP approximation

Maximum a posteriori (MAP) learning: choose h_{MAP} maximizing $P(h_i|\mathbf{d})$ instead of calculating $P(h_i|\mathbf{d})$ for all hypothesis h_i

I.e., maximize $P(\mathbf{d}|h_i)P(h_i)$ or $\log P(\mathbf{d}|h_i) + \log P(h_i)$

Overfitting in MAP and Bayesian learning

- Overfitting when the hypothesis space is too expressive such that some hypotheses fit the date set well.
- Use *prior* to penalize complexity

Log terms can be viewed as (negative of)

bits to encode data given hypothesis + bits to encode hypothesis

This is the basic idea of minimum description length (MDL) learning

For deterministic hypotheses (simplest), $P(\mathbf{d}|h_i)$ is 1 if consistent, 0 otherwise

$\Rightarrow$ MAP = simplest hypothesis that is consistent with the data

ML approximation

For large data sets, prior becomes irrelevant

Maximum likelihood (ML) learning: choose h_{ML} maximizing $P(\mathbf{d}|h_i)$

I.e., simply get the best fit to the data; identical to MAP for uniform prior (which is reasonable if all hypotheses are of the same complexity)

ML is the "standard" (non-Bayesian) statistical learning method

1. Researchers distrust the subjective nature of hypotheses priors
2. Hypotheses are of the same complexity
3. Hypotheses priors is of less important when date set is large
4. Huge space of the hypotheses

Outline

♢ Bayesian learning

♢ Approximate Bayesian learning
- Maximum *a posteriori* learning (MAP)
- Maximum likelihood learning (ML)

♢ Parameter learning with complete data ⇐
- ML parameter learning with complete data in **discrete** models
- ML parameter learning with complete data in **continuous** models (linear regression)
- Naive Bayes models
- Bayesian parameter learning

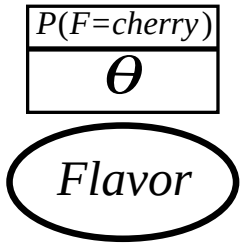
♢ Learning Bayes net structure with complete data

(If time allows)

♢ Learning with hidden variables or incomplete data (EM algorithm)

ML parameter learning in Bayes nets

Bag from a new manufacturer; fraction θ of cherry candies?
Any θ is possible: continuum of hypotheses h_θ
θ is a parameter for this simple (binomial) family of models



We assume all hypotheses are equally possible *a priori*
$\Rightarrow$ ML approach

Suppose we unwrap N candies, c cherries and $\ell = N - c$ limes
These are i.i.d. (independent, identically distributed) observations, so

$$P(\mathbf{d}|h_\theta) = \prod_{j=1}^{N} P(d_j|h_\theta) = \theta^c \cdot (1-\theta)^\ell$$

Maximize this w.r.t. θ—which is easier for the log-likelihood:

$$L(\mathbf{d}|h_\theta) = \log P(\mathbf{d}|h_\theta) = \sum_{j=1}^{N} \log P(d_j|h_\theta) = c\log\theta + \ell\log(1-\theta)$$

$$\frac{dL(\mathbf{d}|h_\theta)}{d\theta} = \frac{c}{\theta} - \frac{\ell}{1-\theta} = 0 \quad \Rightarrow \quad \theta = \frac{c}{c+\ell} = \frac{c}{N}$$

Seems sensible, but causes problems with 0 counts!

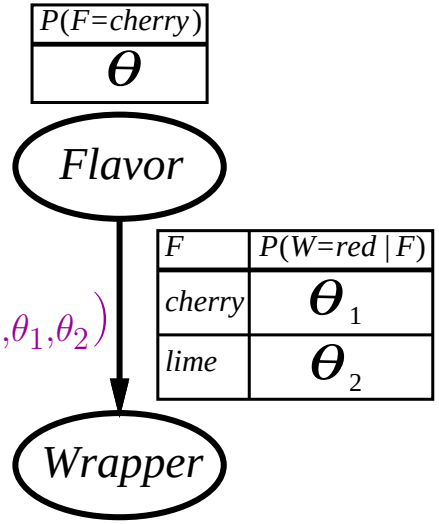
This means that if the data set is small enough that some events have not yet been observed, the ML hypotheses assigns zero to those events. - tricks in dealing with this including initialize the counts for each event to 1 instead of 0.

ML appraoch:

1. Write down an expression forthe likelihood of the data as a function of the parameter(s);
2. Write down the derivative of the log likelihood with respect to each parameter;
3. Find the parameter values such that the derivatives are zero.

Multiple parameters

Red/green wrapper depends probabilistically on flavor:



Likelihood for, e.g., cherry candy in green wrapper:

$$\begin{aligned} & P(F = cherry,\, W = green | h_{\theta,\theta_1,\theta_2}) \\ &= P(F = cherry | h_{\theta,\theta_1,\theta_2}) P(W = green | F = cherry, h_{\theta,\theta_1,\theta_2}) \\ &= \theta \cdot (1 - \theta_1) \end{aligned}$$

N candies, r_c red-wrapped cherry candies, etc.:

$$P(\mathbf{d} | h_{\theta,\theta_1,\theta_2}) = \theta^c (1-\theta)^\ell \cdot \theta_1^{r_c} (1-\theta_1)^{g_c} \cdot \theta_2^{r_\ell} (1-\theta_2)^{g_\ell}$$

$$\begin{aligned} L &= [c \log \theta + \ell \log(1-\theta)] \\ &+ [r_c \log \theta_1 + g_c \log(1-\theta_1)] \\ &+ [r_\ell \log \theta_2 + g_\ell \log(1-\theta_2)] \end{aligned}$$

Multiple parameters contd.

Derivatives of L contain only the relevant parameter:

$$\frac{\partial L}{\partial \theta} = \frac{c}{\theta} - \frac{\ell}{1-\theta} = 0 \qquad \Rightarrow \quad \theta = \frac{c}{c+\ell}$$

$$\frac{\partial L}{\partial \theta_1} = \frac{r_c}{\theta_1} - \frac{g_c}{1-\theta_1} = 0 \qquad \Rightarrow \quad \theta_1 = \frac{r_c}{r_c+g_c}$$

$$\frac{\partial L}{\partial \theta_2} = \frac{r_\ell}{\theta_2} - \frac{g_\ell}{1-\theta_2} = 0 \qquad \Rightarrow \quad \theta_2 = \frac{r_\ell}{r_\ell+g_\ell}$$

With complete data, **parameters can be learned separately**

Parameters values for a variable only depends on the observations of itself and its parents

ML parameter learning (continuous model)

Hypothesis: Learning the parameters (σ and μ) of a Gaussian density function on a single variable

Data: given data generated from this distrubution: $x_1, \cdots, x_N$.

$$P(x) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$$

The log likelihood is

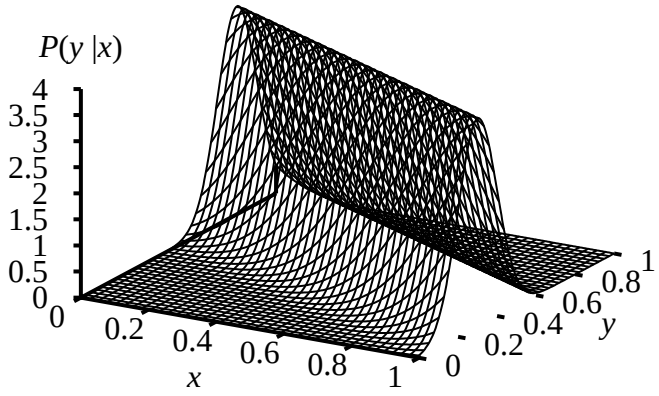
$$L = \sum_{j=1}^{N} \log \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{(x_j-\mu)^2}{2\sigma^2}} = N(-\log\sqrt{2\pi} - \log\sigma) - \sum_{j=1}^{N} \frac{(x_j-\mu)^2}{2\sigma^2}$$

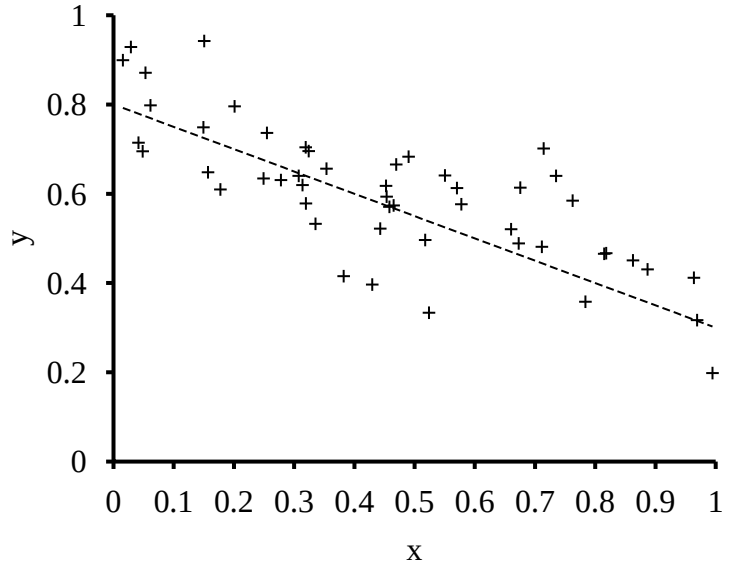
Setting the derivatives to zero

$$\Rightarrow \mu = \frac{\Sigma_j x_j}{N}$$

$$\Rightarrow \sigma = \sqrt{\frac{\Sigma_j (x_j-\mu)^2}{N}} \qquad (1)$$

ML parameter learning (continuous model)





Maximizing $P(y|x) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{(y-(\theta_1 x+\theta_2))^2}{2\sigma^2}}$ w.r.t. θ_1, θ_2

$= $ minimizing $E = \sum_{j=1}^{N} (y_j - (\theta_1 x_j + \theta_2))^2$

That is, minimizing the sum of squared errors gives the ML solution for a linear fit **assuming Gaussian noise of fixed variance**

Naive Bayes models

- Observation: Attributes of one example
- Hypothesis: The class this example belongs to

All attributes are conditionly independent of each other, given the class.

$$P(C|x_1, \cdots, x_n) = \alpha P(C) \prod_i P(x_i|C).$$

Choosing the most likely class

- Simple: 2n+1 parameters, no need to search for h_{ML}
- Surprisingly well in a wide range of applications
- Can deal with noisy data and can give probabilistic prediction

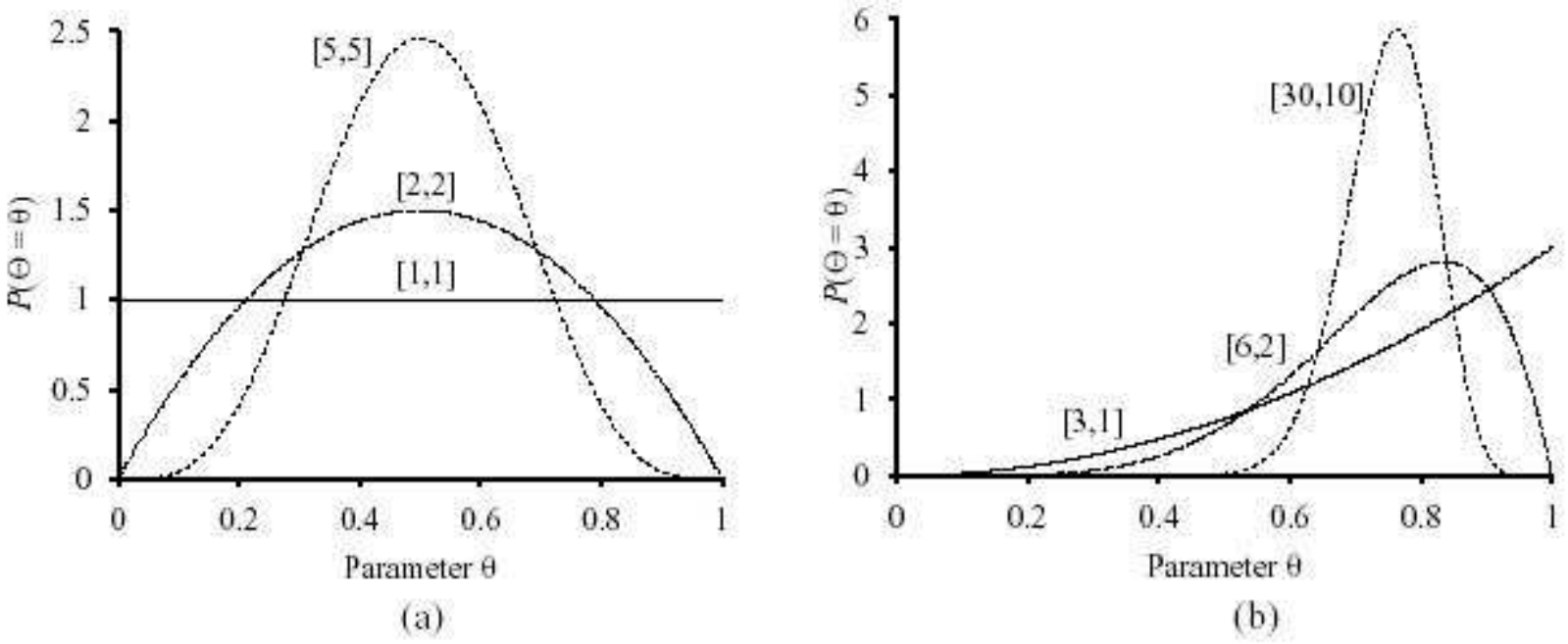
Parameter learning in Bayes nets

- Assume prior as beta distributions:

$$beta[a, b](\theta) = \alpha\theta^{a-1}(1-\theta)^{b-1}$$

α *is the normalization constant*

- Full Bayesian learning



Beta distribution:

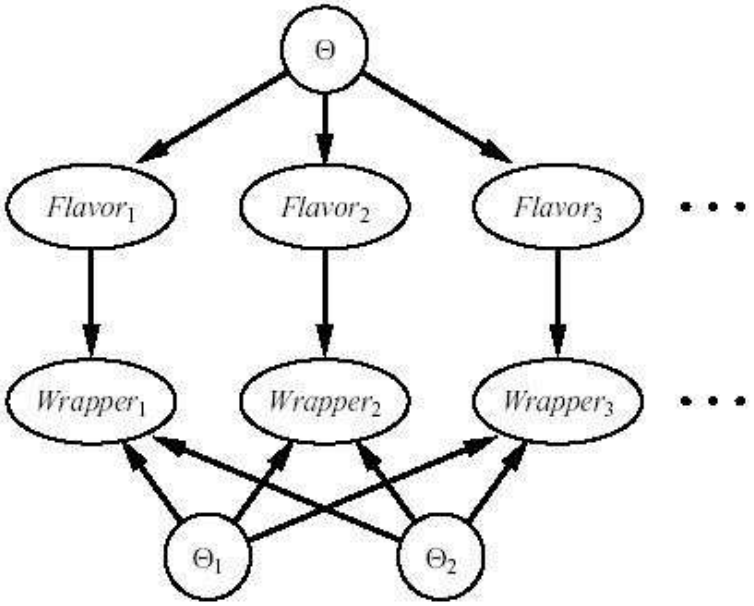
- Mean value of the distribution is: $\frac{a}{a+b}$
- larger values of a suggest a belief that is closer to 1 than to 0
- larger values of $a+b$ make the distribution more peaked (greater certainty about Θ)
- **if the prior of Θ is a beta distribution, after a data point is observed, the posterior distribution of Θ is also a beta distribution**

$$\begin{aligned} P(\theta|D_1 = cherry) &= \alpha P(D_1 = cherry|\theta)P(\theta) \\ &= \alpha'\theta \cdot beta[a,b](\theta) = \alpha'\theta \cdot \theta^{a-1}(1-\theta)^{b-1} \\ &= \alpha'\theta^a(1-\theta)^{b-1} = beta[a+1,b](\theta) \end{aligned} \quad (2)$$

The distribution is converging to a narrow peak around the true vale of Θ as data comes in.

For large data set, Bayesian learning converges to give the same results as ML learning.

Parameter learning in Bayes nets (contd.)



$P(\Theta, \Theta_1, \Theta_2)$ Usually, we assume **parameter independence**. Each parameter has its own beta distribution.

Outline

♢ Bayesian learning

♢ Approximate Bayesian learning
  – Maximum *a posteriori* learning (MAP)
  – Maximum likelihood learning (ML)

♢ Parameter learning with complete data
  – ML parameter learning with complete data in **discrete** models
  – ML parameter learning with complete data in **continuous** models (linear regression)
  – Naive Bayes models
  – Bayesian parameter learning

♢ Learning Bayes net structure with complete data ⇐

(If time allows)

♢ Learning with hidden variables or incomplete data (EM algorithm)

Learning Bayes net structures

Outline

♢ Bayesian learning

♢ Approximate Bayesian learning
- Maximum *a posteriori* learning (MAP)
- Maximum likelihood learning (ML)

♢ Parameter learning with complete data
- ML parameter learning with complete data in **discrete** models
- ML parameter learning with complete data in **continuous** models (linear regression)
- Naive Bayes models
- Bayesian parameter learning

♢ Learning Bayes net structure with complete data

(If time allows)

♢ Learning with hidden variables or incomplete data (EM algorithm)

Learning with hidden variables or incompte date

Summary

Full Bayesian learning gives best possible predictions but is intractable

MAP learning balances complexity with accuracy on training data

Maximum likelihood assumes uniform prior, OK for large data sets

1. Choose a parameterized family of models to describe the data
requires substantial insight and sometimes new models

2. Write down the likelihood of the data as a function of the parameters
may require summing over hidden variables, i.e., inference

3. Write down the derivative of the log likelihood w.r.t. each parameter

4. Find the parameter values such that the derivatives are zero
may be hard/impossible; modern optimization techniques help