Lecture 13

Statistical Learning

Marco Chiarandini

Deptartment of Mathematics & Computer Science
University of Southern Denmark

Slides by Stuart Russell and Peter Norvig

Course Overview

- ✔ Introduction
  - ✔ Artificial Intelligence
  - ✔ Intelligent Agents
- ✔ Search
  - ✔ Uninformed Search
  - ✔ Heuristic Search
- ✔ Adversarial Search
  - ✔ Minimax search
  - ✔ Alpha-beta pruning
- ✔ Knowledge representation and Reasoning
  - ✔ Propositional logic
  - ✔ First order logic
  - ✔ Inference
- ✔ Uncertain knowledge and Reasoning
  - ✔ Probability and Bayesian approach
  - ✔ Bayesian Networks
  - ✔ Hidden Markov Chains
  - ✔ Kalman Filters
- ✔ Learning
  - ✔ Decision Trees
  - Maximum Likelihood
  - EM Algorithm
  - Learning Bayesian Networks
  - Neural Networks
  - ✘ Support vector machines

Last Time

- Decision Trees for classification
  - entropy, information measure
- Performance evaluation
  - overfitting
  - cross validation
  - peeking
  - pruning
- Extensions
  - Ensemble learning
  - boosting
  - bagging

Outline

- ◇ Bayesian learning
- ◇ Maximum *a posteriori* and maximum likelihood learning
- ◇ Bayes net learning
  - – ML parameter learning with complete data
  - – linear regression

Full Bayesian learning

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

Full Bayesian learning

- View learning as Bayesian updating of a probability distribution over the hypothesis space
- H hypothesis variable, values $h_1, h_2, \ldots$, prior $\mathbf{P}(H)$

Full Bayesian learning

- View learning as Bayesian updating of a probability distribution over the hypothesis space
- H hypothesis variable, values $h_1, h_2, \ldots$, prior $\mathbf{P}(H)$
- d_j gives the outcome of random variable D_j (the jth observation) training data $\mathbf{d} = d_1, \ldots, d_N$

Full Bayesian learning

- View learning as Bayesian updating of a probability distribution over the hypothesis space
- H hypothesis variable, values $h_1, h_2, \ldots$, prior $\mathbf{P}(H)$
- d_j gives the outcome of random variable D_j (the jth observation) 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

Full Bayesian learning

- View learning as Bayesian updating of a probability distribution over the hypothesis space
- H hypothesis variable, values $h_1, h_2, \ldots$, prior $\mathbf{P}(H)$
- d_j gives the outcome of random variable D_j (the jth observation) 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

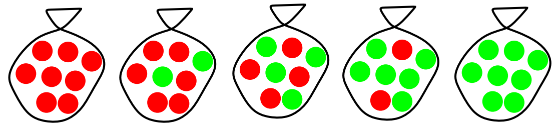
- Predictions use a likelihood-weighted average over the hypotheses:

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

  No need to pick one best-guess hypothesis!

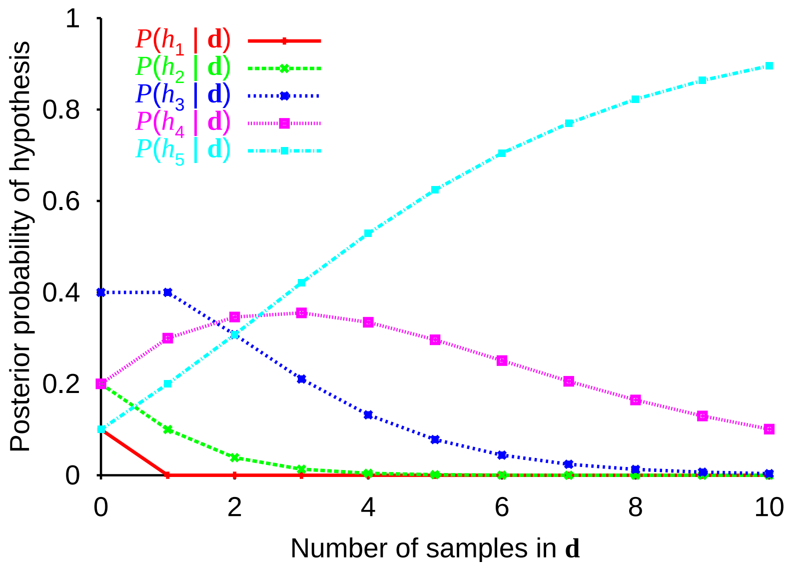
Example

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

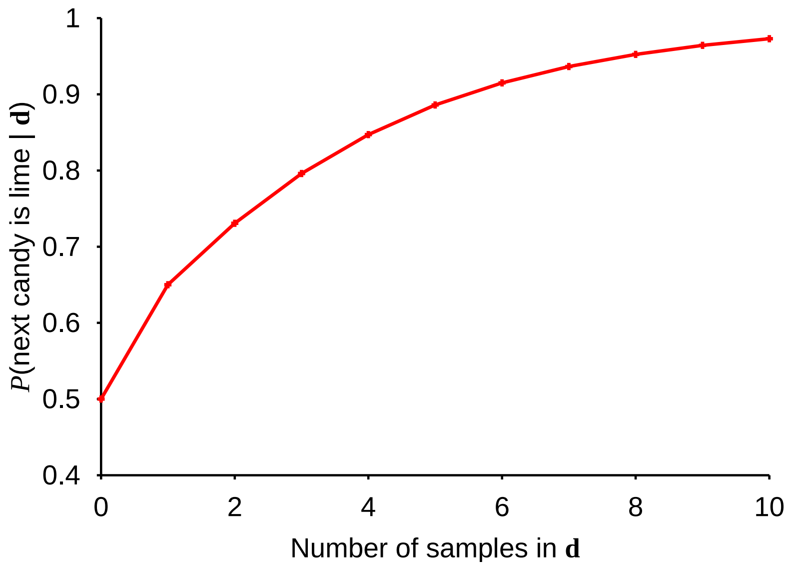


Then we observe candies drawn from some bag: 
What kind of bag is it? What flavour will the next candy be?

Posterior probability of hypotheses



Prediction probability



MAP approximation

- Summing over the hypothesis space is often intractable
  (e.g., 18,446,744,073,709,551,616 Boolean functions of 6 attributes)

MAP approximation

- Summing over the hypothesis space is often intractable
  (e.g., 18,446,744,073,709,551,616 Boolean functions of 6 attributes)

- Maximum a posteriori (MAP) learning: choose h_{MAP} maximizing $P(h_i|\mathbf{d})$
  I.e., maximize $P(\mathbf{d}|h_i)P(h_i)$ or $\log P(\mathbf{d}|h_i) + \log P(h_i)$
  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

MAP approximation

- Summing over the hypothesis space is often intractable
  (e.g., 18,446,744,073,709,551,616 Boolean functions of 6 attributes)

- Maximum a posteriori (MAP) learning: choose h_{MAP} maximizing $P(h_i|\mathbf{d})$
  I.e., maximize $P(\mathbf{d}|h_i)P(h_i)$ or $\log P(\mathbf{d}|h_i) + \log P(h_i)$
  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, $P(\mathbf{d}|h_i)$ is 1 if consistent, 0 otherwise
  $\Longrightarrow$ MAP = simplest consistent hypothesis

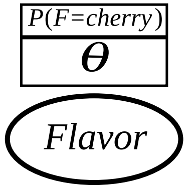
ML approximation

- 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

ML parameter learning in Bayes nets

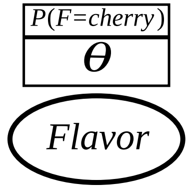


Bag from a new manufacturer; fraction θ of cherry candies?

Any θ is possible: continuum of hypotheses h_θ
θ is a parameter for this simple (binomial) family of models

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

ML parameter learning in Bayes nets



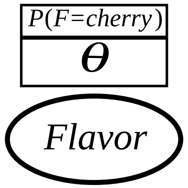
Bag from a new manufacturer; fraction θ of cherry candies?

Any θ is possible: continuum of hypotheses h_θ
θ is a parameter for this simple (binomial) family of models

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

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

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

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

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

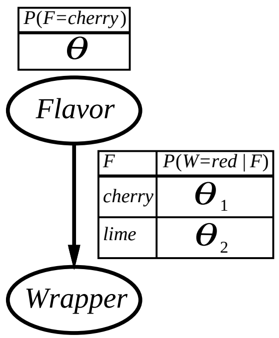
Maximize this w.r.t. θ—which is easier for the log-likelihood:

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

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

Seems sensible, but causes problems with 0 counts!

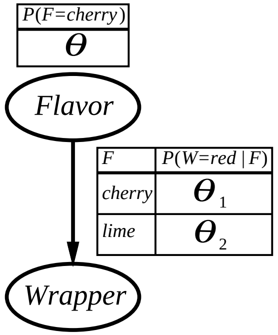
Multiple parameters



Red/green wrapper depends probabilistically on flavor:
Likelihood for, e.g., cherry candy in green wrapper:

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

Multiple parameters



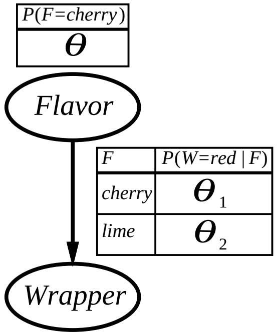
Red/green wrapper depends probabilistically on flavor:
Likelihood for, e.g., cherry candy in green wrapper:

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

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

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

Multiple parameters



Red/green wrapper depends probabilistically on flavor:
Likelihood for, e.g., cherry candy in green wrapper:

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

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:

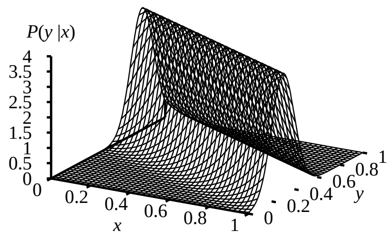
$$\frac{\partial L}{\partial \theta} = \frac{c}{\theta} - \frac{\ell}{1-\theta} = 0 \quad \Longrightarrow \quad \theta = \frac{c}{c+\ell}$$

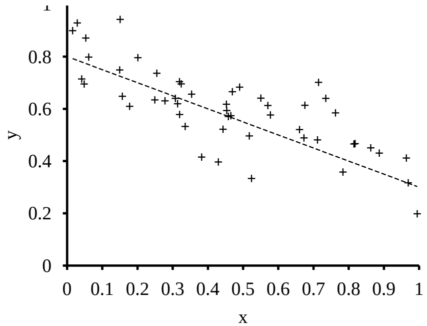
$$\frac{\partial L}{\partial \theta_1} = \frac{r_c}{\theta_1} - \frac{g_c}{1-\theta_1} = 0 \quad \Longrightarrow \quad \theta_1 = \frac{r_c}{r_c+g_c}$$

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

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

Example: linear Gaussian 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

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

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

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