Statistical Learning

CS 486/686
Introduction to AI
University of Waterloo

Motivation: Things you know

- Agents model uncertainty in the world and utility of different courses of actions
  - Bayes nets are models of probability distributions which involve a graph structure annotated with probabilities
  - Bayes nets for realistic applications have hundreds of nodes
- Where do these numbers come from?

Pathfinder
(Heckerman, 1991)

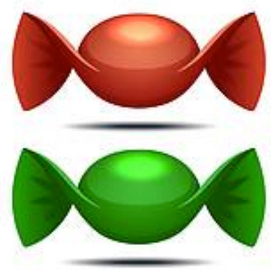
- Medical diagnosis for lymph node disease
- Large net
  - 60 diseases, 100 symptoms and test results, 14000 probabilities
- Built by medical experts
  - 8 hours to determine the variables
  - 35 hours for network topology
  - 40 hours for probability table values

Knowledge acquisition bottleneck

- In many applications, Bayes net structure and parameters are set by experts in the field
  - Experts are scarce and expensive, can be inconsistent or non-existent
- But data is cheap and plentiful (usually)
- Goal of learning:
  - Build models of the world directly from data
  - We will focus on learning models for probabilistic models

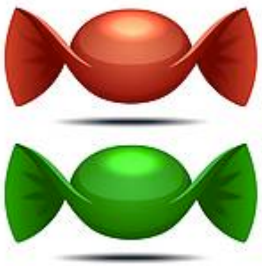
Candy Example (from R&N)

- Favourite candy sold in two flavours
  - Lime and Cherry
- Same wrapper for both flavours
- Sold in bags with different ratios
  - 100% cherry
  - 75% cherry, 25% lime
  - 50% cherry, 50% lime
  - 25% cherry, 75% lime
  - 100% lime



Candy Example

- You bought a bag of candy but do not know its flavour ratio
- After eating k candies
  - What is the flavour ratio of the bag?
  - What will be the flavour of the next candy?



Statistical Learning

- Hypothesis H: probabilistic theory about the world
  - h_1: 100% cherry
  - h_2: 75% cherry, 25% lime
  - h_3: 50% cherry, 50% lime
  - h_4: 25% cherry, 75% lime
  - h_5: 100% lime
- Data D: evidence about the world
  - d_1: 1^{st} candy is cherry
  - d_2: 2^{nd} candy is lime
  - d_3: 3^{rd} candy is lime
  - …

Bayesian learning

- Prior: $P(H)$
- Likelihood: $P(d|H)$
- Evidence: $d=<d_1,d_2,\ldots,d_n>$
- Bayesian learning
  - Compute the probability of each hypothesis given the data
  - $P(H|d)=\alpha\, P(d|H)P(H)$

Bayesian learning

- Suppose we want to make a prediction about some unknown quantity x (i.e. flavour of the next candy)

$$P(x|d) = \sum_i P(x|d, h_i)P(h_i|d)$$

$$= \sum_i P(x|h_i)P(h_i|d)$$

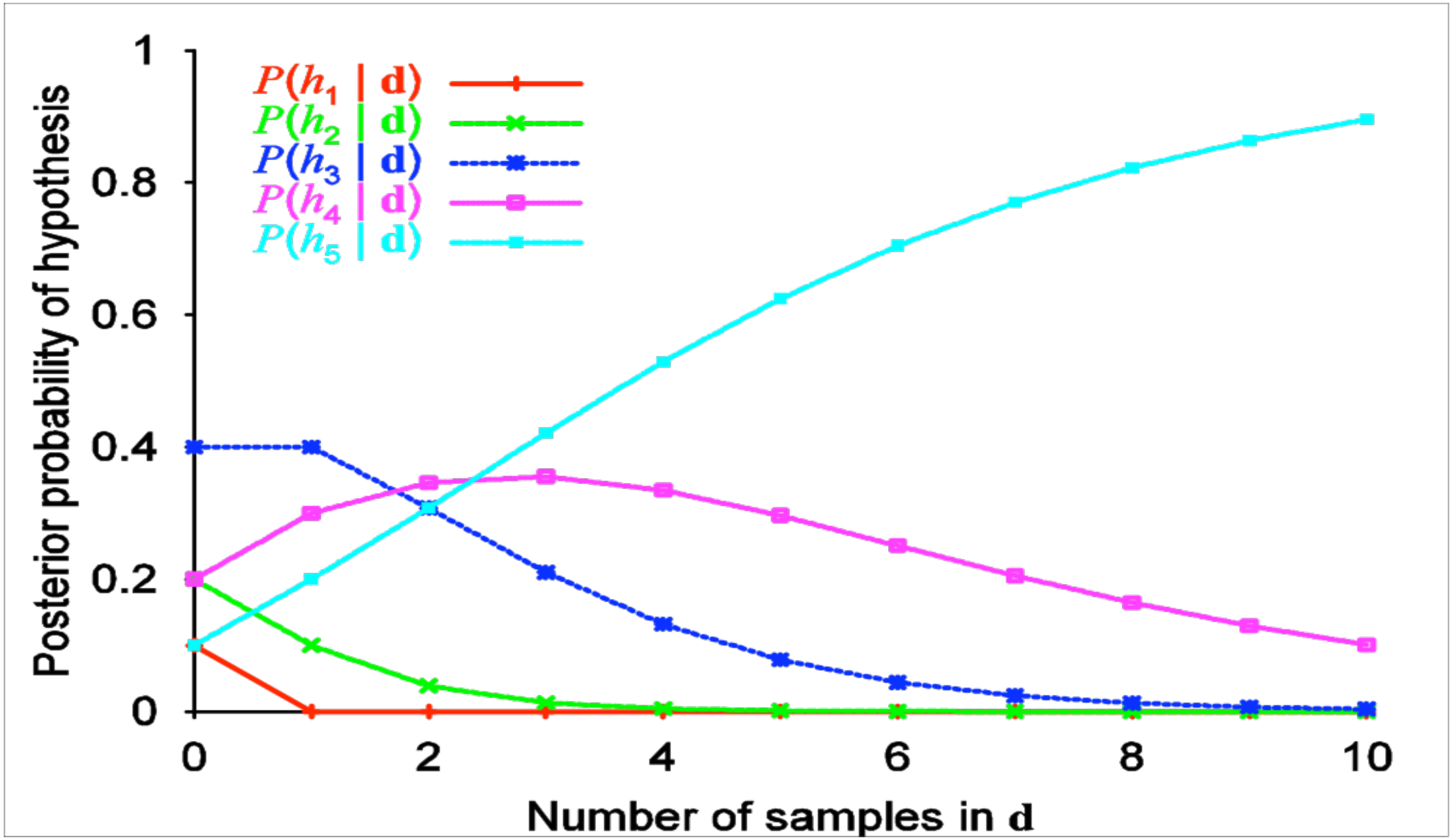
- Predictions are weighted averages of the predictions of the individual hypothesis

Candy Example

- Assume prior $P(H)=\langle 0.1,0.2,0.4,0.2,0.1 \rangle$
- Assume candies are i.i.d: $P(d|h_i)=\Pi_j P(d_j|h_i)$
- Suppose first 10 candies are all lime
  - $P(d|h_1)=0^{10}=0$
  - $P(d|h_2)=0.25^{10}=0.00000095$
  - $P(d|h_3)=0.5^{10}=0.00097$
  - $P(d|h_4)=0.75^{10}=0.056$
  - $P(d|h_5)=1^{10}=1$

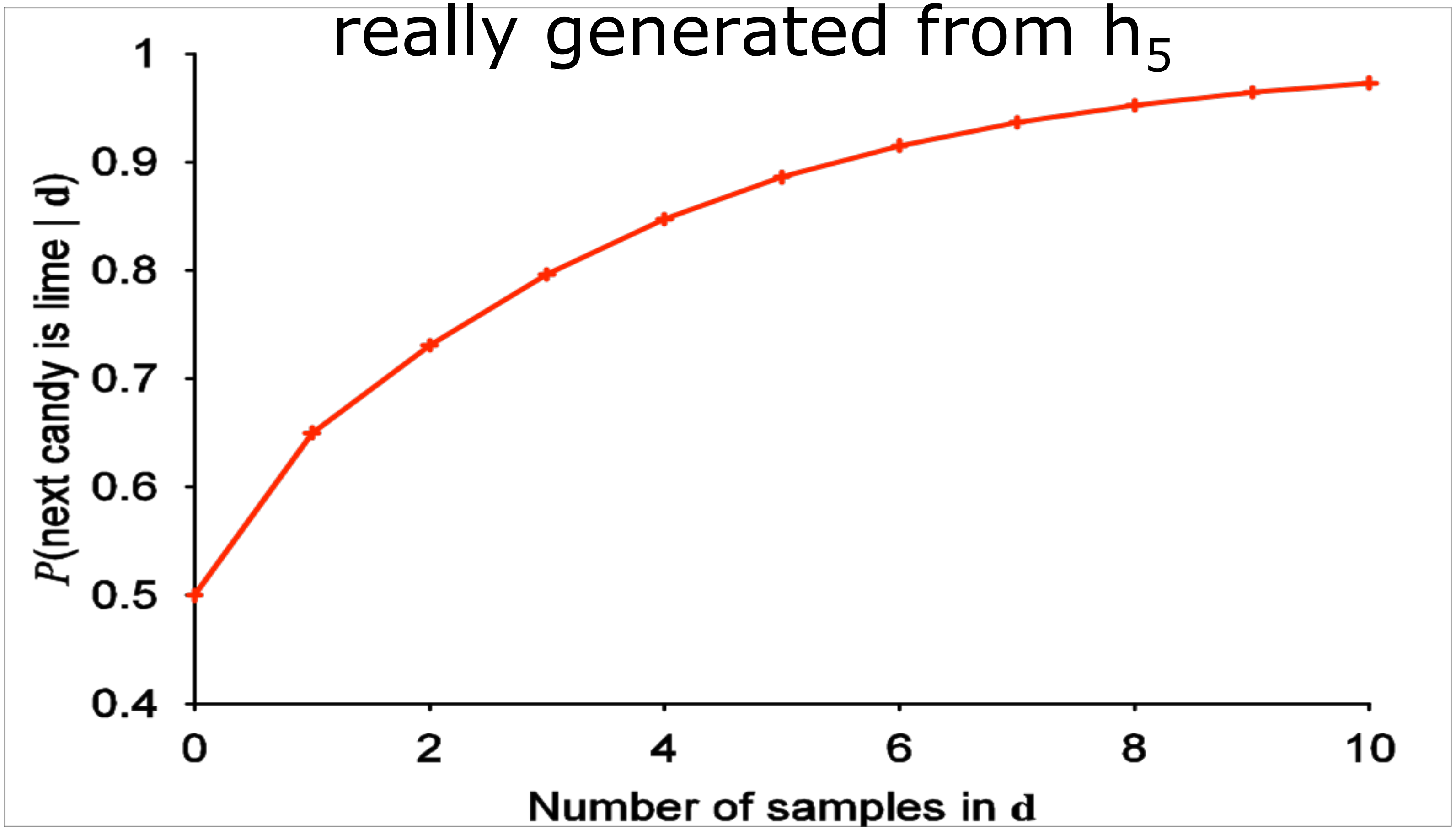
Candy Example: Posterior

Posteriors given that data is really generated from h_5



Candy Example: Prediction

Prediction next candy is lime given that data is really generated from h_5



Bayesian learning

Good News

Optimal: Given prior, no other prediction is correct more often than the Bayesian one

No Overfitting: Use the prior to penalize complex hypothesis (complex hypothesis are unlikely)

Bad News

Intractable: If hypothesis space is large

Solution

Approximations: Maximum a posteriori (MAP)

Maximum a posteriori (MAP)

Idea: Make prediction on the most probable hypothesis h_{MAP}

$$h_{\mathrm{MAP}} = \arg\max_{h_i} P(h_i|d)$$

$$P(x|d) = P(x|h_{\mathrm{MAP}})$$

Compare to Bayesian Learning which makes predictions on all hypothesis weighted by their probability

MAP – Candy Example

MAP Properties

- MAP prediction is less accurate than Bayesian prediction
  - MAP relies on only one hypothesis
- MAP and Bayesian predictions converge as data increases
- No overfitting
  - Use prior to penalize complex hypothesis
- Finding h_{MAP} may be intractable
  - h_{MAP}=argmax P(h|d)
  - Optimization may be hard!

MAP computation

Optimization

$$h_{\text{MAP}} = \arg\max_h P(h|d)$$
$$= \arg\max_h P(h)P(d|h)$$
$$= \arg\max_h P(h) \prod_i P(d_i|h)$$

Product introduces nonlinear optimization

Take log to linearize

$$h_{\text{MAP}} = \arg\max_h \left[\log P(h) + \sum_i \log P(d_i|h) \right]$$

Maximum Likelihood (ML)

- Idea: Simplify MAP by assuming uniform prior (i.e. $P(h_i)=P(h_j)$ for all i,j)

$$h_{\text{MAP}} = \arg\max_h P(h)P(d|h)$$

$$h_{\text{ML}} = \arg\max_h P(d|h)$$

- Make prediction on h_{ML} only
  - $P(x|d)=P(x|h_{ML})$

ML Properties

- ML prediction is less accurate than Bayesian and MAP
- ML, MAP and Bayesian predictions converge as data increases
- Subject to overfitting
  - Does not penalize complex hypothesis
- Finding h_{ML} is often easier than h_{MAP}
  - $h_{ML} = \mathrm{argmax}_j \sum_i \log P(d_i | h_j)$

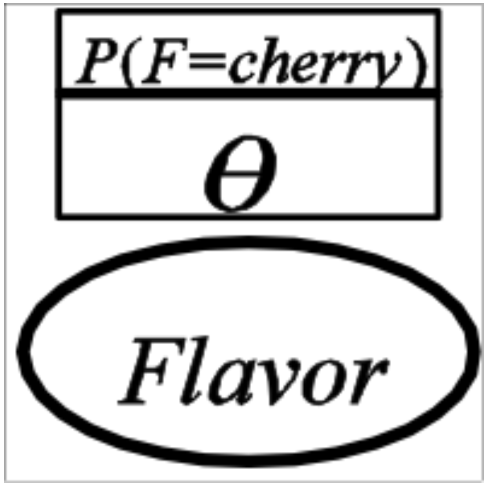
Learning with complete data

- Parameter learning with complete data
  - Parameter learning task involves finding numerical parameters for a probability model whose structure is fixed

- Example: Learning CPT for a Bayes net with a given structure

Simple ML Example

- Hypothesis h_θ
  - $P(cherry)=\theta$ and $P(lime)=1-\theta$
  - θ is our parameter
- Data d:
  - N candies (c cherry and $l=N-c$ lime)
- What should θ be?



Simple ML example

Likelihood of this particular data set

$$P(d|h_\theta) = \theta^c(1-\theta)^l$$

Log Likelihood

$$\begin{aligned} L(d|h_\theta) &= \log P(d|h_\theta) \\ &= c\log\theta + l\log(1-\theta) \end{aligned}$$

Simple ML example

- Find θ that maximizes log likelihood

$$\frac{\partial L(d|h_\theta)}{\partial \theta} = \frac{c}{\theta} - \frac{l}{1-\theta} = 0$$

$$\theta = \frac{c}{c+l} = \frac{c}{N}$$

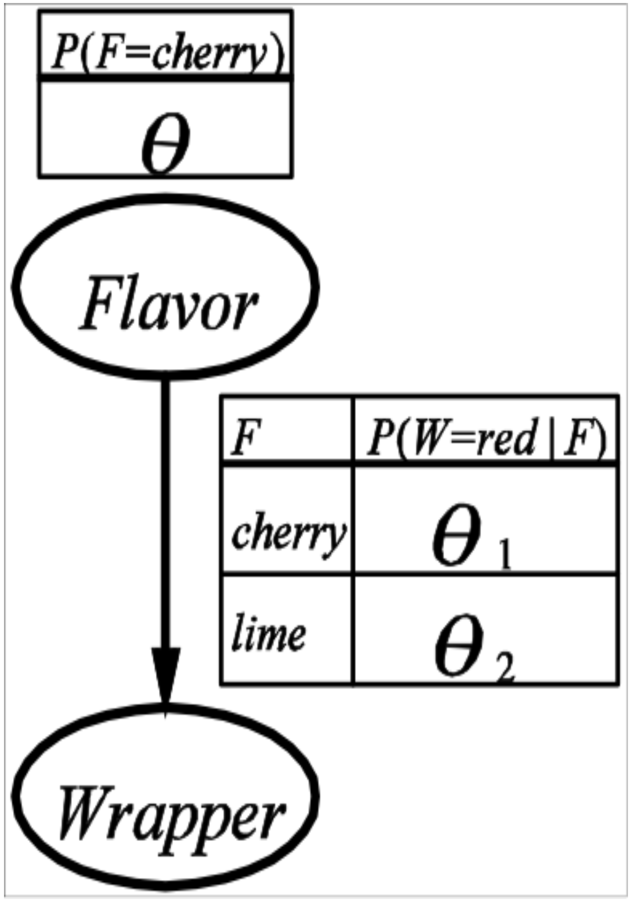
- ML hypothesis asserts that actual proportion of cherries is equal to observed proportion

More complex ML example

- Hypothesis: $h_{\theta, \theta_1, \theta_2}$
- Data:

c Cherries:
G_c green wrappers
R_c red wrappers

l Limes:
G_l green wrappers
R_l red wrappers



More complex ML example

$$P(d|h_{\theta,\theta_1,\theta_2}) = \theta^c(1-\theta)^l\theta_1^{R_c}(1-\theta_1)^{G_c}\theta_2^{R_l}(1-\theta_2)^{G_l}$$

$$\begin{aligned} L(d|h_{\theta,\theta_1,\theta_2}) = & [c\log\theta + l\log(1-\theta)] \\ & + [R_c\log\theta_1 + G_c\log(1-\theta_1)] \\ & + [R_l\log\theta_2 + G_l\log(1-\theta_2)] \end{aligned}$$

More Complex ML

Optimize by taking partial derivatives and setting to zero

$$
\begin{aligned}
\theta &= \frac{c}{c+l} \\
\theta_1 &= \frac{R_c}{R_c + G_c} \\
\theta_2 &= \frac{R_l}{R_l + G_l}
\end{aligned}
$$

ML Comments

- This approach can be extended to any Bayes net
- With complete data
  - ML parameter learning problem decomposes into separate learning problems, one for each parameter!
  - Parameter values for a variable, given its parents are just observed frequencies of variable values for each setting of parent values!

A problem: Zero probabilities

- What happens if we observed zero cherry candies?
  - θ would be set to 0
  - Is this a good prediction?

Instead of $\theta = \dfrac{c}{c+l}$ use $\theta = \dfrac{c+1}{c+l+2}$

Laplace Smoothing

Given observations **x** from N trials

$$\mathbf{x} = (x_1, x_2, \ldots, x_d)$$

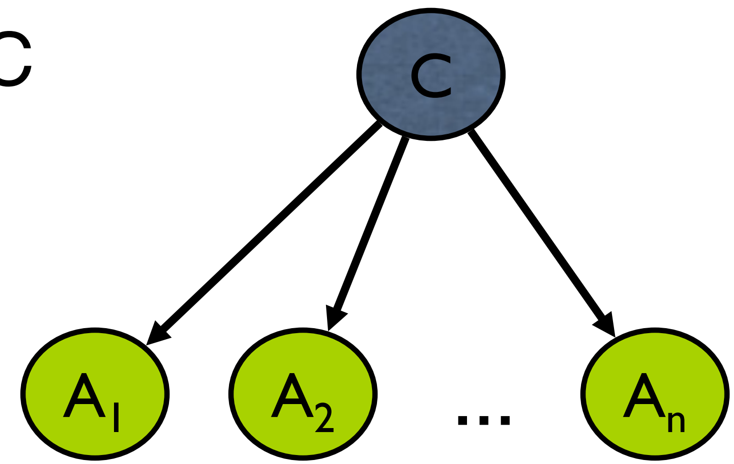
Estimate parameters **θ**

$$\theta = (\theta_1, \theta_2, \ldots, \theta_d)$$

$$\theta_i = \frac{x_i + \alpha}{N + \alpha d} \qquad \alpha > 0$$

Naïve Bayes model

- Want to predict a class C based on attributes A_i
- Parameters:
  - θ =P(C=true)
  - $\theta_{j,1}$=P(A_j=true|C=true)
  - $\theta_{j,2}$=P(A_j=true|C=false)
- Assumption: A_i's are independent given C



Naïve Bayes Model

- With observed attribute values $x_1,x_2,\ldots,x_n$
  - $P(C|x_1,x_2,\ldots,x_n)=\alpha\ P(C)\Pi_i\ P(x_i|C)$
- From ML we know what the parameters should be
  - Observed frequencies (with possible Laplace smoothing)
- Just need to choose the most likely class C

Naïve Bayes comments

- Naïve Bayes scales well
- Naïve Bayes tends to perform well
  - Even though the assumption that attributes are independent given class often does not hold
- Application
  - Text classification

Text classification

- Important practical problem, occurring in many applications
  - Information retrieval, spam filtering, news filtering, building web directories…
- Simplified problem description
  - **Given**: collection of documents, classified as “interesting” or “not interesting” by people
  - **Goal**: learn a classifier that can look at text of new documents and provide a label, without human intervention

Data representation

- Consider all possible significant words that can occur in documents
- Do not include stopwords
- Stem words: map words to their root
- For each root, introduce common binary feature
  - Specifying whether the word is present or not in the document

Example

- “Machine learning is fun”

Use Naïve Bayes Assumption

- Words are independent of each other, given the class, y, of document

$$P(y|\text{document}) \propto \Pi_{i=1}^{|\text{Vocab}|} P(w_i|y)$$

How do we get the probabilities?

Use Naïve Bayes Assumption

Use ML parameter estimation!

$$P(w_i|y) = \frac{\#\text{ documents of class } y \text{ containing word } w_i}{\#\text{ documents of class } y}$$

- Count words over collections of documents
- Use Bayes rule to compute probabilities for unseen documents
- Laplace smoothing is very useful here

Observations

- We may not be able to find θ analytically
- Gradient search to find good value of θ

$$\theta \leftarrow \theta + \alpha \frac{\partial L(\theta|d)}{\partial \theta}$$

Conclusions

- What you should know
  - Bayesian learning, MAP, ML
  - How to learn parameters in Bayes Nets
  - Naïve Bayes assumption
  - Laplace smoothing