Statistical Learning
[RN2 Sec 20.1-20.2]
[RN3 Sec 20.1-20.2]

CS 486/686

University of Waterloo

Lecture 15: Feb 28, 2012

Outline

- Statistical learning
  - Bayesian learning
  - Maximum a posteriori
  - Maximum likelihood
- Learning from complete Data

Statistical Learning

- View: we have uncertain knowledge of the world
- Idea: learning simply reduces this uncertainty

Candy Example

- Favorite candy sold in two flavors:
  - Lime (hugh)
  - Cherry (yum)
- Same wrapper for both flavors
- 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 don't know its flavor ratio
- After eating k candies:
  - What's the flavor ratio of the bag?
  - What will be the flavor of the next candy?

Statistical Learning

- Hypothesis H: probabilistic theory of 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: 1st candy is cherry
  - d_2: 2nd candy is lime
  - d_3: 3rd candy is lime
  - …

Bayesian Learning

- Prior: Pr(H)
- Likelihood: Pr(d|H)
- Evidence: $\mathbf{d} = \langle d_1, d_2, \ldots, d_n \rangle$

- Bayesian Learning amounts to computing the posterior using Bayes' Theorem:
  $Pr(H|\mathbf{d}) = k\ Pr(\mathbf{d}|H)Pr(H)$

Bayesian Prediction

- Suppose we want to make a prediction about an unknown quantity X (i.e., the flavor of the next candy)

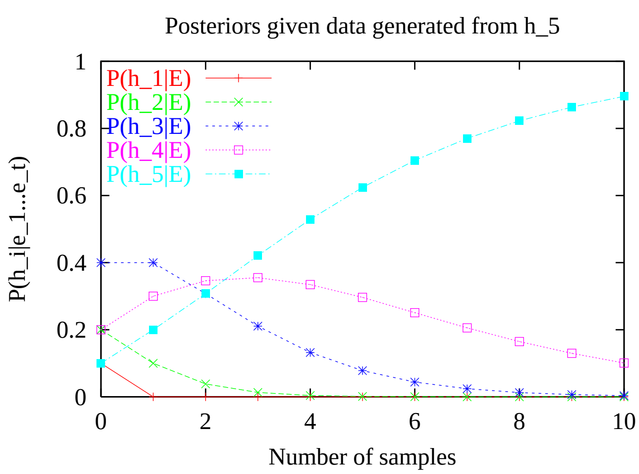
- $Pr(X|\mathbf{d}) = \Sigma_i\ Pr(X|\mathbf{d},h_i)P(h_i|\mathbf{d})$
  $= \Sigma_i\ Pr(X|h_i)P(h_i|\mathbf{d})$

- Predictions are weighted averages of the predictions of the individual hypotheses
- Hypotheses serve as "intermediaries" between raw data and prediction

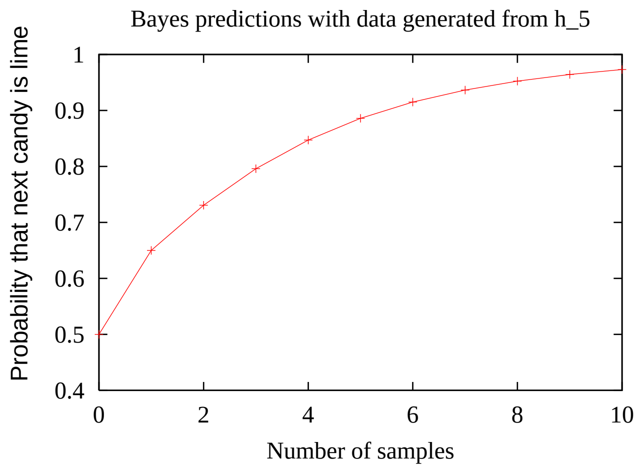
Candy Example

- Assume prior $P(H) = \langle 0.1, 0.2, 0.4, 0.2, 0.1 \rangle$
- Assume candies are i.i.d. (identically and independently distributed)
  - $P(\mathbf{d}|h) = \Pi_j P(d_j|h)$
- Suppose first 10 candies all taste lime:
  - $P(\mathbf{d}|h_5) = 1^{10} = 1$
  - $P(\mathbf{d}|h_3) = 0.5^{10} = 0.00097$
  - $P(\mathbf{d}|h_1) = 0^{10} = 0$

Posterior



Prediction



Bayesian Learning

- Bayesian learning properties:
  - **Optimal** (i.e. given prior, no other prediction is correct more often than the Bayesian one)
  - **No overfitting** (all hypotheses weighted and considered)
- There is a price to pay:
  - When hypothesis space is large Bayesian learning may be intractable
  - i.e. sum (or integral) over hypothesis often intractable
- Solution: approximate Bayesian learning

Maximum a posteriori (MAP)

- Idea: make prediction based on most probable hypothesis h_{MAP}
  - $h_{MAP} = \text{argmax}_{h_i} P(h_i|\mathbf{d})$
  - $P(X|\mathbf{d}) \approx P(X|h_{MAP})$

- In contrast, Bayesian learning makes prediction based on **all** hypotheses weighted by their probability

Candy Example (MAP)

- Prediction after
  - 1 lime: $h_{MAP} = h_3$, $Pr(lime|h_{MAP}) = 0.5$
  - 2 limes: $h_{MAP} = h_4$, $Pr(lime|h_{MAP}) = 0.75$
  - 3 limes: $h_{MAP} = h_5$, $Pr(lime|h_{MAP}) = 1$
  - 4 limes: $h_{MAP} = h_5$, $Pr(lime|h_{MAP}) = 1$
  - …

- After only 3 limes, it correctly selects h_5

Candy Example (MAP)

- But what if correct hypothesis is h_4?
  - h_4: P(lime) = 0.75 and P(cherry) = 0.25
- After 3 limes
  - MAP incorrectly predicts h_5
  - MAP yields $P(lime|h_{MAP}) = 1$
  - Bayesian learning yields $P(lime|\mathbf{d}) = 0.8$

MAP properties

- MAP prediction **less accurate** than Bayesian prediction since it relies only on **one** hypothesis h_{MAP}
- But MAP and Bayesian predictions converge as data increases
- Controlled overfitting (prior can be used to penalize complex hypotheses)
- Finding h_{MAP} may be intractable:
  - $h_{MAP} = \text{argmax } P(h|\mathbf{d})$
  - Optimization may be difficult

MAP computation

- Optimization:
  - $h_{MAP} = \text{argmax}_h P(h|\mathbf{d})$
    $= \text{argmax}_h P(h) P(\mathbf{d}|h)$
    $= \text{argmax}_h P(h) \Pi_i P(d_i|h)$

- Product induces non-linear optimization
- Take the log to linearize optimization
  - $h_{MAP} = \text{argmax}_h \log P(h) + \Sigma_i \log P(d_i|h)$

Maximum Likelihood (ML)

- Idea: simplify MAP by assuming uniform prior (i.e., $P(h_i) = P(h_j)\ \forall i,j$)
  - $h_{MAP} = \text{argmax}_h P(h) P(\mathbf{d}|h)$
  - $h_{ML} = \text{argmax}_h P(\mathbf{d}|h)$

- Make prediction based on h_{ML} only:
  - $P(X|\mathbf{d}) \approx P(X|h_{ML})$

Candy Example (ML)

- Prediction after
  - 1 lime: $h_{ML} = h_5$, $Pr(lime|h_{ML}) = 1$
  - 2 limes: $h_{ML} = h_5$, $Pr(lime|h_{ML}) = 1$
  - …
- Frequentist: "objective" prediction since it relies only on the data (i.e., no prior)
- Bayesian: prediction based on data and uniform prior (since no prior ≡ uniform prior)

ML properties

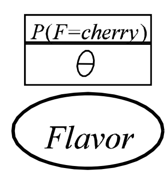
- ML prediction **less accurate** than Bayesian and MAP predictions since it ignores prior info and relies only on **one** hypothesis h_{ML}
- But ML, MAP and Bayesian predictions converge as data increases
- Subject to overfitting (no prior to penalize complex hypothesis that could exploit statistically insignificant data patterns)
- Finding h_{ML} is often easier than h_{MAP}
  - $h_{ML} = \text{argmax}_h \Sigma_i \log P(d_i|h)$

Statistical Learning

- Use Bayesian Learning, MAP or ML
- Complete data:
  - When data has multiple attributes, **all attributes are known**
  - Easy
- Incomplete data:
  - When data has multiple attributes, **some attributes are unknown**
  - Harder

Simple ML example

- Hypothesis h_θ:
  - P(cherry)=θ & P(lime)=1-θ
- Data **d**:
  - c cherries and l limes



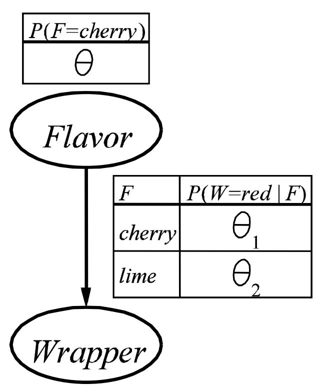
- ML hypothesis:
  - θ is relative frequency of observed data
  - θ = c/(c+l)
  - P(cherry) = c/(c+l) and P(lime)= l/(c+l)

ML computation

- 1) Likelihood expression
  - $P(\mathbf{d}|h_\theta) = \theta^c (1-\theta)^l$
- 2) log likelihood
  - $\log P(\mathbf{d}|h_\theta) = c \log \theta + l \log (1-\theta)$
- 3) log likelihood derivative
  - $d(\log P(\mathbf{d}|h_\theta))/d\theta = c/\theta - l/(1-\theta)$
- 4) ML hypothesis
  - $c/\theta - l/(1-\theta) = 0 \rightarrow \theta = c/(c+l)$

More complicated 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

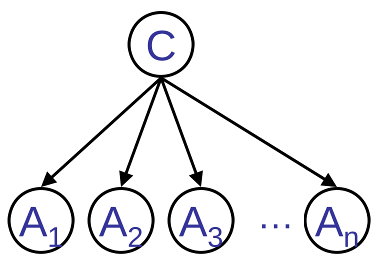


ML computation

- 1) Likelihood expression
  - $P(\mathbf{d}|h_{\theta,\theta_1,\theta_2}) = \theta^c(1-\theta)^l \theta_1^{r_c}(1-\theta_1)^{g_c} \theta_2^{r_l}(1-\theta_2)^{g_l}$
- …
- 4) ML hypothesis
  - $c/\theta - l/(1-\theta) = 0 \Rightarrow \theta = c/(c+l)$
  - $r_c/\theta_1 - g_c/(1-\theta_1) = 0 \Rightarrow \theta_1 = r_c/(r_c+g_c)$
  - $r_l/\theta_2 - g_l/(1-\theta_2) = 0 \Rightarrow \theta_2 = r_l/(r_l+g_l)$

Naïve Bayes model

- Want to predict a class C based on attributes A_i
- Parameters:
  - $\theta = P(C=true)$
  - $\theta_{i1} = P(A_i=true|C=true)$
  - $\theta_{i2} = P(A_i=true|C=false)$
- Assumption: A_i's are independent given C



Naïve Bayes model for Restaurant Problem

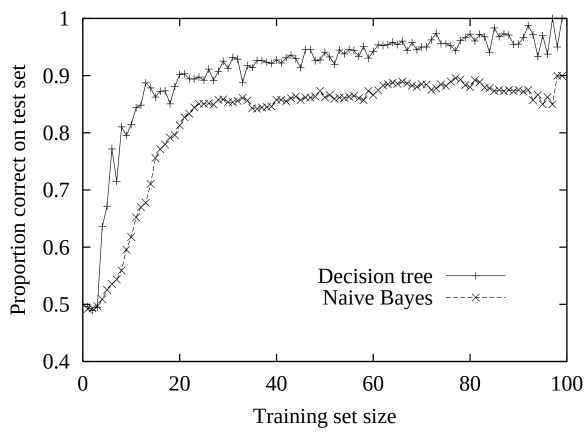
- Data:

Example	Attributes										Target
	Alt	*Bar*	*Fri*	*Hun*	*Pat*	*Price*	*Rain*	*Res*	*Type*	*Est*	*Wait*
X_1	T	F	F	T	Some	$$$	F	T	French	0–10	T
X_2	T	F	F	T	Full	$	F	F	Thai	30–60	F
X_3	F	T	F	F	Some	$	F	F	Burger	0–10	T
X_4	T	F	T	T	Full	$	F	F	Thai	10–30	T
X_5	T	F	T	F	Full	$$$	F	T	French	>60	F
X_6	F	T	F	T	Some	$$	T	T	Italian	0–10	T
X_7	F	T	F	F	None	$	T	F	Burger	0–10	F
X_8	F	F	F	T	Some	$$	T	T	Thai	0–10	T
X_9	F	T	T	F	Full	$	T	F	Burger	>60	F
X_{10}	T	T	T	T	Full	$$$	F	T	Italian	10–30	F
X_{11}	F	F	F	F	None	$	F	F	Thai	0–10	F
X_{12}	T	T	T	T	Full	$	F	F	Burger	30–60	T

- ML sets
  - θ to relative frequencies of *wait* and *~wait*
  - θ_{i1}, θ_{i2} to relative frequencies of each attribute value given *wait* and *~wait*

Naïve Bayes model vs decision trees

- Wait prediction for restaurant problem



Why is naïve Bayes less accurate than decision tree?

Bayesian network parameter learning (ML)

- Parameters $\theta_{V,pa(V)=\mathbf{v}}$:
  - CPTs: $\theta_{V,pa(V)=\mathbf{v}} = P(V|pa(V)=\mathbf{v})$
- Data **d**:
  - d_1 : $\langle V_1=v_{1,1}, V_2=v_{2,1}, \ldots, V_n = v_{n,1}\rangle$
  - d_2 : $\langle V_1=v_{1,2}, V_2=v_{2,2}, \ldots, V_n = v_{n,2}\rangle$
  - …
- Maximum likelihood:
  - Set $\theta_{V,pa(V)=\mathbf{v}}$ to the relative frequencies of the values of V given the values **v** of the parents of V

Next Class

- Next Class:
  - Continue statistical learning
  - Learning from incomplete data