Lecture 20 of 42

Bayesian Classifiers:
MDL, BOC, and Gibbs

Thursday, 01 March 2006

William H. Hsu
Department of Computing and Information Sciences, KSU
http://www.kddresearch.org
http://www.cis.ksu.edu/~bhsu

Readings:
Sections 6.6-6.8, Mitchell
Chapter 14, Russell and Norvig

Lecture Outline

- Read Sections 6.6-6.8, Mitchell; Chapter 14, Russell and Norvig
- This Week's Paper Review: "Learning in Natural Language", Roth
- Minimum Description Length (MDL) Revisited
  - Probabilistic interpretation of the MDL criterion: justification for Occam's Razor
  - Optimal coding: Bayesian Information Criterion (BIC)
- Bayes Optimal Classifier (BOC)
  - Implementation of BOC algorithms for practical inference
  - Using BOC as a "gold standard"
- Gibbs Classifier and Gibbs Sampling
- Simple (Naïve) Bayes
  - Tradeoffs and applications
  - Handout: "Improving Simple Bayes", Kohavi *et al*
- Next Lecture: Sections 6.9-6.10, Mitchell
  - More on simple (naïve) Bayes
  - Application to learning over text

Bayesian Learning: Synopsis

- Components of Bayes's Theorem: Prior and Conditional Probabilities

$$P(h \mid D) = \frac{P(D \mid h)P(h)}{P(D)} = \frac{P(h \wedge D)}{P(D)}$$

- *P*(*h*) ≡ Prior Probability of (*Correctness of*) Hypothesis *h*
  - Uniform priors: *no background knowledge* $P(h) = \frac{1}{|H|}$
  - Background knowledge can skew priors away from ~ Uniform(*H*)
- *P*(*h* | *D*) ≡ Probability of *h* Given Training Data *D*
- *P*(*h* ∧ *D*) ≡ Joint Probability of *h* and *D*
- *P*(*D*) ≡ Probability of *D*
  - Expresses distribution D: *P*(*D*) ~ D $P(D) = \sum_{x \in D} P(D \mid h) \cdot P(h)$
  - To compute: marginalize joint probabilities
- *P*(*D* | *h*) ≡ Probability of *D* Given *h*
  - Probability of observing *D* given that *h* is correct ("generative" model)
  - *P*(*D* | *h*) = 1 if *h* consistent with *D* (i.e., $\forall x_i . h(x_i) = c(x_i)$), 0 otherwise

Review: MAP and ML Hypotheses

- Bayes's Theorem

$$P(h \mid D) = \frac{P(D \mid h)P(h)}{P(D)} = \frac{P(h \wedge D)}{P(D)}$$

- MAP Hypothesis
  - Maximum *a posteriori* hypothesis, h_{MAP}

$$\begin{aligned} h_{MAP} &= \arg\max_{h \in H} P(h \mid D) \\ &= \arg\max_{h \in H} \frac{P(D \mid h)P(h)}{P(D)} \\ &= \arg\max_{h \in H} P(D \mid h)P(h) \end{aligned}$$

  - Caveat: *maximizing P*(*h* | *D*) versus combining *h* values may not be best
- ML Hypothesis
  - Maximum likelihood hypothesis, h_{ML}

$$h_{ML} = \arg\max_{h_i \in H} P(D \mid h_i)$$

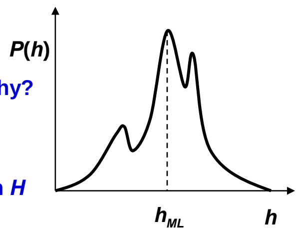
  - Sufficient for computing MAP when priors *P*(*h*) are uniformly distributed
    - Hard to estimate *P*(*h* | *D*) in this case
    - Solution approach: encode knowledge about *H* in *P*(*h*) - explicit bias

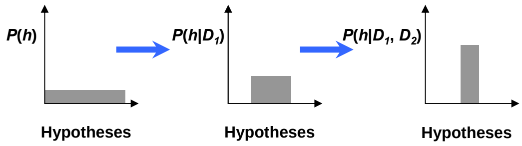
Maximum Likelihood Estimation (MLE)

- **ML Hypothesis**
  - Maximum likelihood hypothesis, h_{ML}

$$h_{ML} = \arg\max_{h_i \in H} P(D \mid h_i)$$

  - Uniform priors: posterior $P(h \mid D)$ hard to estimate - why?
    - Recall: belief revision given evidence (data)
    - "No knowledge" means we need more evidence
    - Consequence: more computational work to search H





- **ML Estimation (MLE): Finding h_{ML} for Unknown Concepts**
  - Recall: log likelihood (a log prob value) used - directly proportional to likelihood
  - In practice, estimate the descriptive statistics of $P(D \mid h)$ to approximate h_{ML}
  - e.g., μ_{ML}: ML estimator for unknown mean ($P(D)$ ~ Normal) ≡ sample mean

Minimum Description Length (MDL) Principle: Occam's Razor

- **Occam's Razor**
  - Recall: prefer the shortest hypothesis - an inductive bias
  - Questions
    - Why short hypotheses as opposed to an arbitrary class of *rare* hypotheses?
    - What is special about minimum description length?
  - Answers
    - MDL approximates an optimal coding strategy for hypotheses
    - *In certain cases*, this coding strategy maximizes conditional probability
  - Issues
    - How exactly is "minimum length" being achieved (length of what)?
    - When and why can we use "MDL learning" for MAP hypothesis learning?
    - *What does "MDL learning" really entail (what does the principle buy us)?*
- **MDL Principle**
  - Prefer h that minimizes coding length of model plus coding length of exceptions
  - Model: encode h using a coding scheme C_1
  - Exceptions: encode the conditioned data $D \mid h$ using a coding scheme C_2

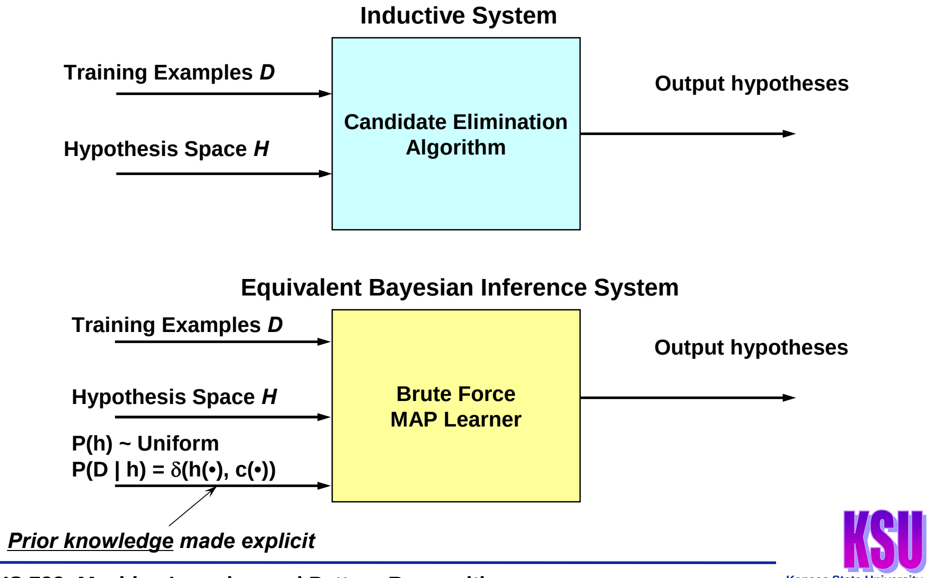
MDL and Optimal Coding: Bayesian Information Criterion (BIC)

- **MDL Hypothesis** $h_{MDL} = \arg\min_{h \in H} \left[L_{C_1}(h) + L_{C_2}(D \mid h) \right]$
  - e.g., $H \equiv$ decision trees, D = labeled training data
  - $L_{C_1}(h) \equiv$ number of bits required to describe tree h under encoding C_1
  - $L_{C_2}(D \mid h) \equiv$ number of bits required to describe D given h under encoding C_2
  - NB: $L_{C_2}(D \mid h) = 0$ if all x classified perfectly by h (*need only describe exceptions*)
  - Hence h_{MDL} *trades off* tree size against training errors
- **Bayesian Information Criterion** $BIC(h) = \lg P(D \mid h) + \lg P(h)$
  - $h_{MAP} = \arg\max_{h \in H} [P(D \mid h) \cdot P(h)] = \arg\max_{h \in H} [\lg P(D \mid h) + \lg P(h)] = \arg\max_{h \in H} BIC(h)$
    $= \arg\min_{h \in H} [-\lg P(D \mid h) - \lg P(h)]$
  - Interesting fact from information theory: the optimal (shortest expected code length) code for an event with probability p is $-\lg(p)$ bits
  - Interpret h_{MAP} as total length of h and D given h *under optimal code*
  - BIC = -MDL (i.e., *argmax* of BIC is *argmin* of MDL criterion)
  - Prefer hypothesis that minimizes length(h) + length (*misclassifications*)

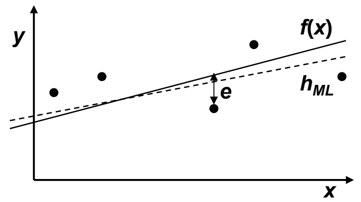
Concluding Remarks on MDL

- **What Can We Conclude?**
  - *Q: Does this prove once and for all that short hypotheses are best?*
  - A: Not necessarily…
    - Only shows: *if* we find log-optimal representations for $P(h)$ and $P(D \mid h)$, then $h_{MAP} = h_{MDL}$
    - No reason to believe that h_{MDL} is preferable for *arbitrary* codings C_1, C_2
  - Case in point: practical probabilistic knowledge bases
    - Elicitation of a full description of $P(h)$ and $P(D \mid h)$ is hard
    - Human implementor might prefer to specify *relative probabilities*
- **Information Theoretic Learning: Ideas**
  - Learning as compression
    - Abu-Mostafa: complexity of learning problems (in terms of minimal codings)
    - Wolff: computing (especially search) as compression
  - (Bayesian) model selection: searching H using probabilistic criteria

Characterizing Learning Algorithms by Equivalent MAP Learners



Maximum Likelihood: Learning A Real-Valued Function [1]



- **Problem Definition**
  - Target function: any real-valued function *f*
  - Training examples $<x_i, y_i>$ where y_i is noisy training value
    - $y_i = f(x_i) + e_i$
    - e_i is random variable (noise) i.i.d. ~ Normal (0, σ), *aka* Gaussian noise
  - Objective: approximate *f* as closely as possible
- **Solution**
  - Maximum likelihood hypothesis h_{ML}
  - Minimizes sum of squared errors (SSE)

$$h_{ML} = \arg\min_{h \in H} \sum_{i=1}^{m} \left(d_i - h(x_i)\right)^2$$

Maximum Likelihood: Learning A Real-Valued Function [2]

- **Derivation of Least Squares Solution**
  - Assume noise is Gaussian (prior knowledge)
  - Max likelihood solution: $h_{ML} = \arg\max_{h \in H} p(D \mid h)$

$$= \arg\max_{h \in H} \prod_{i=1}^{m} p(d_i \mid h)$$

$$= \arg\max_{h \in H} \prod_{i=1}^{m} \frac{1}{\sqrt{2\pi\pi^2}} e^{-\frac{1}{2}\left(\frac{d_i - h(x_i)}{\sigma}\right)^2}$$

- **Problem: Computing Exponents, Comparing Reals - Expensive!**
- **Solution: Maximize Log Prob**

$$h_{ML} = \arg\max_{h \in H} \sum_{i=1}^{m} \left[\ln\left(\frac{1}{\sqrt{2\pi\sigma^2}}\right) - \frac{1}{2}\left(\frac{d_i - h(x_i)}{\sigma}\right)^2 \right]$$

$$= \arg\max_{h \in H} \sum_{i=1}^{m} \left[-\frac{1}{2}\left(\frac{d_i - h(x_i)}{\sigma}\right)^2 \right]$$

$$= \arg\max_{h \in H} \sum_{i=1}^{m} -(d_i - h(x_i))^2$$

$$= \arg\min_{h \in H} \sum_{i=1}^{m} (d_i - h(x_i))^2$$

Bayesian Classification

- **Framework**
  - Find most probable *classification* (as opposed to MAP *hypothesis*)
  - $f: X \to V$ (domain ≡ instance space, range ≡ finite set of values)
  - Instances $x \in X$ can be described as a collection of features $x \equiv (x_1, x_2, \ldots, x_n)$
  - Performance element: Bayesian classifier
    - Given: an example (e.g., Boolean-valued instances: $x_i \in \mathbf{H}$)
    - Output: the most probable value $v_j \in V$ (NB: priors for x constant *wrt* v_{MAP})

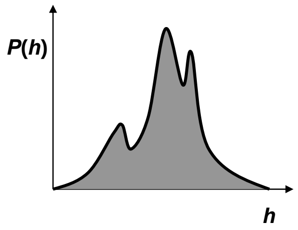
$$v_{MAP} = \arg\max_{v_j \in V} P(v_j \mid x) = \arg\max_{v_j \in V} P(v_j \mid x_1, x_2, \ldots, x_n)$$

$$= \arg\max_{v_j \in V} P(x_1, x_2, \ldots, x_n \mid v_j) P(v_j)$$

- **Parameter Estimation Issues**
  - Estimating $P(v_j)$ is easy: for each value v_j, count its frequency in $D = \{<x, f(x)>\}$
  - However, it is infeasible to estimate $P(x_1, x_2, \ldots, x_n \mid v_j)$: too many 0 values
  - In practice, *need to make assumptions* that allow us to estimate $P(x \mid d)$

Bayes Optimal Classifier (BOC)

- **Intuitive Idea**
  - $h_{MAP}(x)$ is not necessarily the most probable classification!
  - Example
    - Three possible hypotheses: $P(h_1 \mid D) = 0.4$, $P(h_2 \mid D) = 0.3$, $P(h_3 \mid D) = 0.3$
    - Suppose that for new instance x, $h_1(x) = +$, $h_2(x) = -$, $h_3(x) = -$
    - What is the most probable classification of x?
- **Bayes Optimal Classification (BOC)** $v^* = v_{BOC} = \arg\max_{v_j \in V} \sum_{h_i \in H} \left[P(v_j \mid h_i) \cdot P(h_i \mid D) \right]$
  - Example
    - $P(h_1 \mid D) = 0.4$, $P(- \mid h_1) = 0$, $P(+ \mid h_1) = 1$
    - $P(h_2 \mid D) = 0.3$, $P(- \mid h_2) = 1$, $P(+ \mid h_2) = 0$
    - $P(h_3 \mid D) = 0.3$, $P(- \mid h_3) = 1$, $P(+ \mid h_3) = 0$
  - $\sum_{h_i \in H} \left[P(+ \mid h_i) \cdot P(h_i \mid D) \right] = 0.4$
    $\sum_{h_i \in H} \left[P(- \mid h_i) \cdot P(h_i \mid D) \right] = 0.6$
  - Result: $v^* = v_{BOC} = \arg\max_{v_j \in V} \sum_{h_i \in H} \left[P(v_j \mid h_i) \cdot P(h_i \mid D) \right] = -$



BOC and Concept Learning

- **Back to Concept Learning (Momentarily)**
  - Recall: every consistent hypothesis has MAP probability

$$P(h \mid D) = \begin{cases} \dfrac{1}{|VS_{H,D}|} & \text{if } h \text{ is consistent with } D \\ 0 & \text{otherwise} \end{cases}$$

  - Bayes optimal prediction

$$P(x = + \mid D) = \sum_{h_i \in H} \left[P(x = + \mid h_i) \cdot P(h_i \mid D) \right] = \frac{1}{|VS_{H,D}|} \sum_{h_i \in H} h_i(x)$$

    - Weighted sum of the predictions of all consistent hypotheses
    - *"Each hypothesis contributes in proportion to its own likelihood"*
- **Properties of Bayes Optimal Prediction**
  - Classifer does not necessarily correspond to any hypothesis $h \in H$
  - BOC algorithm searches for its classifier in a wider concept class
  - Allows linear combinations of H's elements

BOC and Evaluation of Learning Algorithms

- Method: Using The BOC as A "Gold Standard"
  - Compute classifiers
    - Bayes optimal classifier
    - Sub-optimal classifier: gradient learning ANN, simple (Naïve) Bayes, etc.
  - Compute results: apply classifiers to produce predictions
  - Compare results to BOC's to evaluate ("percent of optimal")
- Evaluation in Practice
  - Some classifiers work well *in combination*
    - Combine classifiers with each other
    - Later: weighted majority, mixtures of experts, bagging, boosting
    - *Why is the BOC the best in this framework, too?*
  - Can be used to evaluate "global optimization" methods too
    - e.g., genetic algorithms, simulated annealing, and other stochastic methods
    - Useful if convergence properties are to be compared
  - *NB*: not always feasible to compute BOC (often intractable)

KSU
Kansas State University
Department of Computing and Information Sciences

BOC for Development of New Learning Algorithms

- Practical Application: BOC as Benchmark
  - Measuring "how close" local optimization methods come to finding BOC
  - Measuring how efficiently global optimization methods converge to BOC
  - Tuning high-level parameters (of relatively low dimension)
- Approximating the BOC
  - Genetic algorithms (covered later)
    - Approximate BOC in a practicable fashion
    - Exploitation of (mostly) task parallelism and (some) data parallelism
  - Other random sampling (stochastic search)
    - Markov chain Monte Carlo (MCMC)
    - e.g., Bayesian learning in ANNs [Neal, 1996]
- BOC as Guideline
  - Provides a baseline when feasible to compute
  - Shows deceptivity of *H* (how many local optima?)
  - Illustrates role of incorporating background knowledge

KSU
Kansas State University
Department of Computing and Information Sciences

Gibbs Classifier

- **Difficulties with BOC**
  - Computationally expensive if |*H*| is high
  - Intractable (i.e., NP-hard) for many non-trivial learning problems
- **Solution Approach**
  - A stochastic classifier: result of random sampling
  - Gibbs algorithm: simple random sampling
    - Select a hypothesis *h* from *H* according to *P*(*h* | *D*)
    - Use this *h* to classify new *x*
- **Quality of Gibbs Classification: Example**
  - Assume target concepts are drawn from *H* according to *P*(*h*)
  - Suprising fact: error bound $E[error(h_{Gibbs})] \leq 2E[error(h_{BayesOptimal})]$
  - Suppose assumption correct: uniform priors on *H*
    - Select any $h \in VS_{H,D} \sim$ Uniform(*H*)
    - Expected error no worse than twice Bayes optimal!

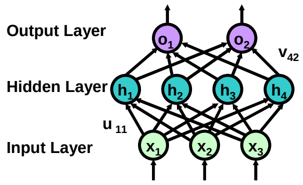
Gibbs Classifier: Practical Issues

- **Gibbs Classifier in Practice**
  - BOC comparison yields an *expected case* ratio bound of 2
  - Can we afford mistakes made when individual hypotheses fall outside?
  - General questions
    - *How many examples must we see for h to be accurate with high probability?*
    - *How far off can h be?*
  - Analytical approaches for answering these questions
    - Computational learning theory
    - Bayesian estimation: statistics (e.g., aggregate loss)
- **Solution Approaches**
  - Probabilistic knowledge
    - Q: *Can we improve on uniform priors?*
    - A: It depends on the problem, but sometimes, yes (stay tuned)
  - Global optimization: Monte Carlo methods (Gibbs sampling)
    - Idea: if sampling *one h* yields a ratio bound of 2, how about sampling *many*?
    - Combine many random samples to simulate integration

Bayesian Learning: Parameter Estimation

- **Bayesian Learning: General Case**
  - Model parameters θ
    - These are the basic trainable parameters (e.g., ANN weights)
    - Might describe graphical structure (e.g., decision tree, Bayesian network)
    - Includes any "low level" model parameters that we can train
  - Hyperparameters (higher-order parameters) γ
    - Might be control statistics (e.g., mean and variance of priors on weights)
    - Might be "runtime options" (e.g., max depth or size of DT; BN restrictions)
    - Includes any "high level" control parameters that we can tune
- **Concept Learning: Bayesian Methods**
  - Hypothesis *h* consists of (θ, γ)
  - γ values used to *control* update of θ values
  - e.g., priors ("seeding" the ANN), stopping criteria

Case Study: BOC and Gibbs Classifier for ANNs [1]



- **Methodology**
  - θ (model parameters): a_j, u_{ij}, b_k, v_{jk}
  - γ (hyperparameters): σ_a, σ_u, σ_b, σ_v
- **Computing Feedforward ANN Output**
  - Output layer activation: $f_k(x) = b_k + \Sigma_j v_{jk} h_j(x)$
  - Hidden layer activation: $h_j(x) = tanh(a_j + \Sigma_i u_{ij} x_i)$
- **Classifier Output: Prediction**
  - Given new input from "inference space"
  - Want: Bayesian optimal test output

$$P\left(y^{(m+1)} \mid x^{(m+1)}, \left(x^{(1)}, y^{(1)}\right), \ldots, \left(x^{(m)}, y^{(m)}\right)\right)$$
$$= \int P\left(y^{(m+1)} \mid x^{(n+1)}, \theta\theta\right) P\left(\theta, \gamma \mid \left(x^{(1)}, y^{(1)}\right), \ldots, \left(x^{(m)}, y^{(m)}\right)\right) d\theta d\gamma$$

$$P\left(\theta \mid \left(x^{(1)}, y^{(1)}\right), \ldots, \left(x^{(m)}, y^{(m)}\right)\right) = \frac{P\left(\left(x^{(1)}, y^{(1)}\right), \ldots, \left(x^{(m)}, y^{(m)}\right) \mid \theta\right) P(\theta)}{P\left(\left(x^{(1)}, y^{(1)}\right), \ldots, \left(x^{(m)}, y^{(m)}\right)\right)}$$
$$\propto L\left(\theta \mid \left(x^{(1)}, y^{(1)}\right), \ldots, \left(x^{(m)}, y^{(m)}\right)\right) P(\theta)$$

Case Study: BOC and Gibbs Classifier for ANNs [2]

- **Problem**
  - True parameter space is infinite (real-valued weights and thresholds)
  - Worse yet, we know nothing about target distribution $P(h \mid D)$
- **Solution: Markov chain Monte Carlo (MCMC) Methods**
  - Sample from a conditional density for $h = (\theta, \gamma)$
  - Integrate over these samples numerically (as opposed to analytically)
- **MCMC Estimation: An Application of Gibbs Sampling**
  - Want: a function $v(\theta)$, e.g., $f_k(x^{(n+1)}, \theta)$
  - Target: $E[v] = \int v(\theta) Q(\theta)\, d\theta$
  - MCMC estimate: $E[v] \approx \frac{1}{N} \sum_{t=1}^{N} v\left(\theta^{(t)}\right)$
  - FAQ [MacKay, 1999]: http://wol.ra.phy.cam.ac.uk/mackay/Bayes_FAQ.html

BOC and Gibbs Sampling

- **Gibbs Sampling: Approximating the BOC**
  - Collect many Gibbs samples
  - Interleave the update of parameters and hyperparameters
    - e.g., train ANN weights using Gibbs sampling
    - Accept a candidate Δw if it improves error *or rand() ≤ current threshold*
    - After every few thousand such transitions, sample hyperparameters
  - Convergence: lower *current threshold* slowly
  - Hypothesis: return model (e.g., network weights)
  - Intuitive idea: sample models (e.g., ANN snapshots) *according to likelihood*
- **How Close to Bayes Optimality Can Gibbs Sampling Get?**
  - Depends on how many samples taken (how slowly *current threshold* is lowered)
  - Simulated annealing terminology: annealing schedule
  - More on this when we get to genetic algorithms

Terminology

- Minimum Description Length (MDL)
  - Bayesian Information Criterion (BIC) $BIC(h) = \lg P(D \mid h) + \lg P(h)$
  - BIC = additive inverse of MDL (i.e., $BIC(h) = -MDL(h)$)
- Bayesian Classification: Finding Most Probable *v* Given Examples *x*
- Bayes Optimal Classifier (BOC)
  - Probabilistic learning criteria: measures of *P*(*prediction* | *D*) or *P*(*hypothesis* | *D*)
  - BOC: a gold standard for probabilistic learning criteria
- Gibbs Classifier
  - Randomly sample *h* according to *P*(*h* | *D*), then use to classify
  - Ratio bound: error no worse than 2 • Bayes optimal error
  - MCMC methods (Gibbs sampling): Monte Carlo integration over *H*
- Simple Bayes *aka* Naïve Bayes
  - Assumption of conditional independence of attributes given classification
  - Naïve Bayes classifier: factors conditional distribution of *x* given label *v*

$$v_{NB} = \arg\max_{v_j \in V} P(v_j) \prod_i P(x_i \mid v_j)$$

Summary Points

- Minimum Description Length (MDL) Revisited
  - Bayesian Information Criterion (BIC): justification for Occam's Razor
- Bayes Optimal Classifier (BOC)
  - Using BOC as a "gold standard"
- Gibbs Classifier
  - Ratio bound
- Simple (Naïve) Bayes
  - Rationale for assumption; pitfalls
- Practical Inference using MDL, BOC, Gibbs, Naïve Bayes
  - MCMC methods (Gibbs sampling)
  - Glossary: http://www.media.mit.edu/~tpminka/statlearn/glossary/glossary.html
  - To learn more: http://bulky.aecom.yu.edu/users/kknuth/bse.html
- Next Lecture: Sections 6.9-6.10, Mitchell
  - More on simple (naïve) Bayes
  - Application to learning over text