Lecture 11

Simple (Naïve) Bayes and Probabilistic Learning over Text

Thursday, September 30, 1999

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

Readings:
Sections 6.9-6.10, Mitchell

Lecture Outline

- Read Sections 6.9-6.10, Mitchell
- More on Simple Bayes, *aka* Naïve Bayes
  - More examples
  - Classification: choosing between two classes; general case
  - Robust estimation of probabilities
- Learning in Natural Language Processing (NLP)
  - Learning over text: problem definitions
  - Case study: *Newsweeder* (Naïve Bayes application)
  - Probabilistic framework
  - Bayesian approaches to NLP
    - Issues: word sense disambiguation, part-of-speech tagging
    - Applications: spelling correction, web and document searching
- Next Week: Section 6.11, Mitchell; Pearl and Verma
  - Read: "Bayesian Networks without Tears", Charniak
  - Go over Chapter 15, Russell and Norvig; Heckerman tutorial (slides)

Naïve Bayes Algorithm

- Recall: MAP Classifier $v_{MAP} = \arg\max_{v_j \in V} P(v_j \mid x) = \arg\max_{v_j \in V} P(v_j \mid x_1, x_2, \ldots, x_n)$
  $= \arg\max_{v_j \in V} P(x_1, x_2, \ldots, x_n \mid v_j) P(v_j)$
- Simple (Naïve) Bayes Assumption $P(x_1, x_2, \ldots, x_n \mid v_j) = \prod_i P(x_i \mid v_j)$
- Simple (Naïve) Bayes Classifier $v_{NB} = \arg\max_{v_j \in V} P(v_j) \prod_i P(x_i \mid v_j)$
- Algorithm Naïve-Bayes-Learn (D)
  - FOR each target value v_j
    $\hat{P}(v_j) \leftarrow estimate[P(v_j)]$
    FOR each attribute value x_{ik} of each attribute x_i
    $\hat{P}(x_{ik} \mid v_j) \leftarrow estimate[P(v_j)]$
  - RETURN $\{\hat{P}(x_{ik} \mid v_j)\}$
- Function Classify-New-Instance-NB ($x \equiv \langle x_{1k}, x_{2k}, \ldots, x_{nk} \rangle$)
  - $v_{NB} = \arg\max_{v_j \in V} \left[\hat{P}(v_j) \prod_{i=1}^{n} \hat{P}(x_i = x_{ik} \mid v_j) \right]$
  - RETURN v_{NB}

Conditional Independence

- Attributes: Conditionally Independent (CI) Given Data
  - $P(x, y \mid D) = P(x \mid D) \cdot P(y \mid D)$: D "mediates" x, y (not necessarily independent)
  - Conversely, independent variables are not necessarily CI given any function
- Example: Independent but Not CI
  - Suppose $P(x = 0) = P(x = 1) = 0.5$, $P(y = 0) = P(y = 1) = 0.5$, $P(xy) = P(x)P(y)$
  - Let $f(x, y) = x \wedge y$
  - $f(x, y) = 0 \Rightarrow P(x = 1 \mid f = 0) = P(y = 1 \mid f = 0) = 1/3$, $P(x = 1, y = 1 \mid f = 0) = 0$
  - x and y are independent but not CI given f
- Example: CI but Not Independent
  - Suppose $P(x = 1 \mid f = 0) = 1$, $P(y = 1 \mid f = 0) = 0$, $P(x = 1 \mid f = 1) = 0$, $P(y = 1 \mid f = 1) = 1$
  - Suppose $P(f = 0) = P(f = 1) = 1/2$
  - $P(x = 1) = 1/2$, $P(y = 1) = 1/2$, $P(x = 1) \cdot P(y = 1) = 1/4 \neq P(x = 1, y = 1) = 0$
  - x and y are CI given f but not independent
- Moral: *Choose Evidence Carefully* and Understand Dependencies

Naïve Bayes: Example [1]

- Concept: *PlayTennis* $v_{NB} = \arg\max_{v_j \in V} \left[\hat{P}(v_j) \prod_{i=1}^{n} \hat{P}(x_i = x_{ik} \mid v_j) \right]$

Day	Outlook	Temperature	Humidity	Wind	PlayTennis?
1	Sunny	Hot	High	Light	No
2	Sunny	Hot	High	Strong	No
3	Overcast	Hot	High	Light	Yes
4	Rain	Mild	High	Light	Yes
5	Rain	Cool	Normal	Light	Yes
6	Rain	Cool	Normal	Strong	No
7	Overcast	Cool	Normal	Strong	Yes
8	Sunny	Mild	High	Light	No
9	Sunny	Cool	Normal	Light	Yes
10	Rain	Mild	Normal	Light	Yes
11	Sunny	Mild	Normal	Strong	Yes
12	Overcast	Mild	High	Strong	Yes
13	Overcast	Hot	Normal	Light	Yes
14	Rain	Mild	High	Strong	No

- Application of Naïve Bayes: Computations
  - $P(PlayTennis) = \{Yes, No\}$ 2 numbers
  - $P(Outlook) = \{Sunny, Overcast, Rain\} \mid PT = \{Yes, No\}$ 6 numbers
  - $P(Temp) = \{Hot, Mild, Cool\} \mid PT = \{Yes, No\}$ 6 numbers
  - $P(Humidity) = \{High, Normal\} \mid PT = \{Yes, No\}$ 4 numbers
  - $P(Wind) = \{Light, Strong\} \mid PT = \{Yes, No\}$ 4 numbers

Naïve Bayes: Example [2]

- Query: New Example $x = \langle Sunny, Cool, High, Strong, ? \rangle$
  - Desired inference: $P(PlayTennis = Yes \mid x) = 1 - P(PlayTennis = No \mid x)$
  - $P(PlayTennis = Yes) = 9/14 = 0.64$ $P(PlayTennis = No) = 5/14 = 0.36$
  - $P(Outlook = Sunny \mid PT = Yes) = 2/9$ $P(Outlook = Sunny \mid PT = No) = 3/5$
  - $P(Temperature = Cool \mid PT = Yes) = 3/9$ $P(Temperature = Cool \mid PT = No) = 1/5$
  - $P(Humidity = High \mid PT = Yes) = 3/9$ $P(Humidity = High \mid PT = No) = 4/5$
  - $P(Wind = Strong \mid PT = Yes) = 3/9$ $P(Wind = Strong \mid PT = No) = 3/5$
- Inference
  - $P(PlayTennis = Yes, \langle Sunny, Cool, High, Strong \rangle) =$
    $P(Yes)\, P(Sunny \mid Yes)\, P(Cool \mid Yes)\, P(High \mid Yes)\, P(Strong \mid Yes) = 0.0053$
  - $P(PlayTennis = No, \langle Sunny, Cool, High, Strong \rangle) =$
    $P(No)\, P(Sunny \mid No)\, P(Cool \mid No)\, P(High \mid No)\, P(Strong \mid No) = 0.0206$
  - $v_{NB} = No$
  - NB: $P(x) = 0.0053 + 0.0206 = 0.0259 \Rightarrow$
    $P(PlayTennis = No \mid x) = 0.0206 / 0.0259 = 0.795$

Naïve Bayes: Subtle Issues [1]

- **Conditional Independence Assumption Often Violated**
  - CI assumption: $P(x_1, x_2, \ldots, x_n \mid v_j) = \prod_i P(x_i \mid v_j)$
  - However, it works well surprisingly well anyway
  - *Note*
    - Don't need estimated conditional probabilities $\hat{P}(v_j \mid x)$ to be correct
    - Only need

$$v_{NB} = \arg\max_{v_j \in V} \left[\hat{P}(v_j) \prod_{i=1}^{n} \hat{P}(x_i = x_{ik} \mid v_j)\right]$$
$$= \arg\max_{v_j \in V} \left[P(v_j) P(x_1, x_2, \ldots, x_n \mid v_j)\right]$$

- **See [Domingos and Pazzani, 1996] for analysis**

Naïve Bayes: Subtle Issues [2]

- **Naïve Bayes Conditional Probabilities Often Unrealistically Close to 0 or 1**
  - Scenario: what if none of the training instances with target value v_j have $x_i = x_{ik}$?
    - $\hat{P}(x \mid v_j) = 0 \Rightarrow \hat{P}(v_j) \prod_i \hat{P}(x_i = x_{ik} \mid v_j) = 0$
    - Ramification: *one missing term* is enough to disqualify the label v_j
  - e.g., *P(Alan Greenspan | Topic = NBA)* = 0 in news corpus
  - Many such zero counts
- **Solution Approaches (See [Kohavi, Becker, and Sommerfield, 1996])**
  - No-match approaches: replace $P = 0$ with $P = c/m$ (e.g., $c = 0.5$, 1) or $P(v)/m$
  - Bayesian estimate (*m*-estimate) for $\hat{P}(x_{ik} \mid v_j) \leftarrow \frac{n_{ik,j} + mp}{n_j + m}$
    - $n_j \equiv$ number of examples $\ni v = v_j$, $n_{ik,j} \equiv$ number of examples $\ni v = v_j$ and $x_i = x_{ik}$
    - $p \equiv$ prior estimate for $\hat{P}(x_{ik} \mid v_j)$; $m \equiv$ weight given to prior ("virtual" examples)
    - *aka* Laplace approaches: see Kohavi *et al* ($P(x_{ik} \mid v_j) \leftarrow (N + f)/(n + kf)$)
    - $f \equiv$ control parameter; $N \equiv n_{ik,j}$; $n \equiv n_j$; $1 \le v \le k$

Learning to Classify Text

- **Why? (Typical Learning Applications)**
  - *Which news articles are of interest?*
  - Classify web pages by topic
    - Browsable indices: *Yahoo*, *Einet Galaxy*
    - Searchable dynamic indices: *Lycos*, *Excite*, *Hotbot*, *Webcrawler*, *AltaVista*
  - Information retrieval: *What articles match the user's query?*
    - Searchable indices (for digital libraries): *MEDLINE* (*Grateful Med*), *INSPEC*, *COMPENDEX*, etc.
    - Applied bibliographic searches: citations, patent intelligence, etc.
  - *What is the correct spelling of this homonym?* (e.g., *plane* vs. *plain*)
- **Naïve Bayes: Among Most Effective Algorithms in Practice**
- **Implementation Issues**
  - Document representation: attribute vector representation of text documents
  - Large vocabularies (thousands of keywords, millions of key phrases)

Learning to Classify Text: Probabilistic Framework

- **Target Concept *Interesting?* : Document → {+, –}**
- **Problem Definition**
  - Representation
    - Convert each document to a vector of words (w_1, w_2, …, w_n)
    - One attribute per word position in document
  - Learning
    - Use training examples to estimate *P*(+), *P*(–), *P*(*document* | +), *P*(*document* | –)
  - Assumptions
    - Naïve Bayes conditional independence assumption

$$P(document \mid v_j) = \prod_{i=1}^{length(document)} \hat{P}(x_i = w_k \mid v_j)$$

    - Here, w_k denotes word k in a vocabulary of N words ($1 \le k \le N$)
    - $P(x_i = w_k \mid v_j)$ = probability that word in position i is word k, given document v_j
    - $\forall\, i, m$. $P(x_i = w_k \mid v_j) = P(x_m = w_k \mid v_j)$: word CI of position given v_j

Learning to Classify Text: A Naïve Bayesian Algorithm

- **Algorithm *Learn-Naïve-Bayes-Text* (*D*, *V*)**
  - 1. Collect all words, punctuation, and other tokens that occur in *D*
    - *Vocabulary* ← {all distinct words, tokens occurring in any document $x \in D$}
  - 2. Calculate required $P(v_j)$ and $P(x_i = w_k \mid v_j)$ probability terms
    - FOR each target value $v_j \in V$ DO
      - *docs*[*j*] ← {documents $x \in D \ni v(x) = v_j$}
      - $P(v_j) = \frac{|docs[j]|}{|D|}$
      - *text*[*j*] ← Concatenation (*docs*[*j*]) // a single document
      - *n* ← total number of distinct word positions in *text*[*j*]
      - FOR each word w_k in *Vocabulary*
        - n_k ← number of times word w_k occurs in *text*[*j*]
        - $P(w_k \mid v_j) = \frac{n_k + 1}{n + |Vocabulary|}$
  - 3. RETURN <{$P(v_j)$}, {$P(w_k \mid v_j)$}>

Learning to Classify Text: Applying Naïve Bayes Classifier

- **Function *Classify-Naïve-Bayes-Text* (*x*, *Vocabulary*)**
  - Positions ← {word positions in document *x* that contain tokens found in Vocabulary}
  - RETURN $v_{NB} = \arg\max_{v_j \in V} P(v_j) \prod_{i \in Positions} P(x_i \mid v_j)$
- **Purpose of *Classify-Naïve-Bayes-Text***
  - Returns estimated target value for new document
  - x_i: denotes word found in the i^{th} position within *x*

Example: Twenty Newsgroups

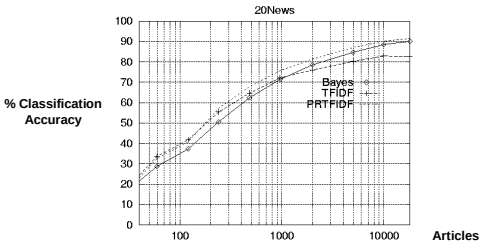
- 20 *USENET* Newsgroups
  - comp.graphics
  - comp.os.ms-windows.misc
  - comp.sys.ibm.pc.hardware
  - comp.sys.mac.hardware
  - comp.windows.x
  - misc.forsale
  - rec.autos
  - rec.motorcycles
  - rec.sports.baseball
  - rec.sports.hockey
  - soc.religion.christian
  - talk.politics.guns
  - talk.politics.mideast
  - talk.politics.misc
  - talk.religion.misc
  - alt.atheism
  - sci.space
  - sci.crypt
  - sci.electronics
  - sci.med
- Problem Definition [Joachims, 1996]
  - Given: 1000 training documents (posts) from each group
  - Return: classifier for new documents that identifies the group it belongs to
- Example: Recent Article from *comp.graphics.algorithms*

> Hi all
>
> I'm writing an adaptive marching cube algorithm, which must deal with cracks. I got the vertices of the cracks in a list (one list per crack).
>
> Does there exist an algorithm to triangulate a concave polygon ? Or how can I bisect the polygon so, that I get a set of connected convex polygons.
>
> The cases of occuring polygons are these:
>
> ...

- Performance of *Newsweeder* (Naïve Bayes): 89% Accuracy

Learning Curve for Twenty Newsgroups

- Newsweeder Performance: Training Set Size versus Test Accuracy
  - 1/3 holdout for testing



- Found: Superset of "Useful and Interesting" Articles
  - Evaluation criterion: user feedback (ratings elicited while reading)

Learning Framework for Natural Language: Statistical Queries (SQ)

- Statistical Queries (SQ) Algorithm [Kearns, 1993]
  - New learning protocol
    - So far: learner receives labeled examples or makes queries with them
    - SQ algorithm: learning algorithm that requests values of statistics on $\mathcal{D}$
    - Example: "What is $P(x_i = 0, v = +)$ for $x \sim \mathcal{D}$?"
  - Definition
    - Statistical query: a tuple $[x, v_j, \tau]$
    - x: an attribute ("feature"), v_j: a value ("label"), τ: an error parameter
    - SQ oracle: returns estimate $\hat{P}^D_{[x_{ik}, v_j, \tau]} = P^D_{[x_{ik}, v_j]} \left\{ \langle x, v(x) \rangle \mid x_i = x_{ik} \wedge v(x) = v_j \right\}$
    - Estimate satisfies error bound: $\left| \hat{P}_{x_{ik}} - P_{x_{ik}} \right| < \tau$
    - SQ algorithm: learning algorithm that searches for h using *only* SQ oracle
- Simulation of the SQ Oracle
  - Take large sample $D = \{\langle x, v(x) \rangle\}$
  - Evaluate simulated query: $\hat{P}_D(\langle x_{ik}, v_j \rangle) = \frac{\left| \{ \langle x, v(x) \rangle = \langle x_{ik}, v_j \rangle \} \right|}{|D|}$

Learning Framework for Natural Language: Linear Statistical Queries (LSQ) Hypotheses

- Linear Statistical Queries (LSQ) Hypothesis [Kearns, 1993; Roth, 1999]
  - Predicts $v_{LSQ}(x)$ (e.g., $\in \{+, -\}$) given $x \in X$ when

$$v_{LSQ}(x') = \arg\max_{v_j \in V} \sum_{i=1}^{n} f_{[x_i, v_j]}\left(\hat{P}^D_{[x_i, v_j]}\right) I_{x_i = 1}$$

  - What does this mean? LSQ classifier…
    - Takes a query example x
    - Asks its built-in SQ oracle for estimates $\hat{P}^D_{[x_i', v_j]}$ on each x_i' (that satisfy error bound τ)
    - Computes f_{ij}(*estimated conditional probability*), coefficients for x_i', label v_j
    - Returns the most likely label according to this linear discriminator
- What Does This Framework Buy Us?
  - Naïve Bayes is one of a large family of LSQ learning algorithms
  - Includes: BOC (must transform x); (hidden) Markov models; max entropy

Learning Framework for Natural Language: Naïve Bayes and LSQ

- Key Result: Naïve Bayes is A Case of LSQ

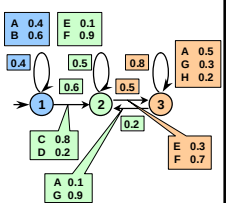
$$f_{[x,v_j]}(\bullet) = \lg \hat{P}^D_{[x,v_j]}$$

$$f_{[x_i,v_j]}(\bullet) = \lg\left(\frac{\hat{P}^D_{[x_i,v_j]}}{\hat{P}^D_{[x,v_j]}}\right), i = 1, 2, \ldots, n$$

- Variants of Naïve Bayes: Dealing with Missing Values
  - Q: *What can we do when x_i is missing?*
  - A: Depends on whether x_i is unknown or truly missing (not recorded or corrupt)
    - Method 1: just leave it out (use when truly missing) - standard LSQ
    - Method 2: treat as *false* or a known default value - modified LSQ
    - Method 3 [Domingos and Pazzani, 1996]: introduce a new value, "?"
  - See [Roth, 1999] and [Kohavi, Becker, and Sommerfield, 1996] for more info

Learning Framework for Natural Language: (Hidden) Markov Models

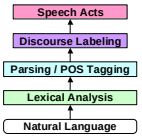
- Definition of Hidden Markov Models (HMMs)
  - Stochastic state transition diagram (HMMs: states, *aka* nodes, are hidden)
  - Compare: probabilistic finite state automaton (Mealy/Moore model)
  - Annotated transitions (*aka* arcs, edges, links)
    - *Output alphabet* (the observable part)
    - Probability distribution over *outputs*
- Forward Problem: One Step in ML Estimation
  - Given: model h, observations (data) D
  - Estimate: $P(D \mid h)$
- Backward Problem: Prediction Step
  - Given: model h, observations D
  - Maximize: $P(h(X) = x \mid h, D)$ for a new X
- Forward-Backward (Learning) Problem
  - Given: model space H, data D
  - Find: $h \in H$ such that $P(h \mid D)$ is maximized (i.e., MAP hypothesis)
- HMMs Also A Case of LSQ (*f* Values in [Roth, 1999])



NLP Issues: Word Sense Disambiguation (WSD)

- Problem Definition
  - Given: m sentences, each containing a usage of a particular ambiguous word
  - Example: "The can will rust." (auxiliary verb versus noun)
  - Label: $v_j \equiv s \equiv$ correct word sense (e.g., $s \in$ {auxiliary verb, noun})
  - Representation: m examples (labeled attribute vectors $\langle (w_1, w_2, \ldots, w_n), s \rangle$)
  - Return: classifier $f: X \rightarrow V$ that disambiguates new $x \equiv (w_1, w_2, \ldots, w_n)$
- Solution Approach: Use Bayesian Learning (e.g., Naïve Bayes)
  - Caveat: can't observe s in the text! $P(w_1, w_2, \ldots, w_n \mid s) = \prod_{i=1}^{n} P(w_i \mid s)$
  - A solution: treat s in $P(w_i \mid s)$ as *missing value*, impute s (assign by inference)
  - [Pedersen and Bruce, 1998]: fill in using Gibbs sampling, EM algorithm (later)
  - [Roth, 1998]: Naïve Bayes, sparse networks of Winnows (SNOW), TBL
- Recent Research
  - T. Pedersen's research home page: http://www.d.umn.edu/~tpederse/
  - D. Roth's Cognitive Computation Group: http://l2r.cs.uiuc.edu/~cogcomp/

NLP Issues: Part-of-Speech (POS) Tagging



- Problem Definition
  - Given: m sentences containing untagged words
  - Example: "The can will rust."
  - Label (one per word, out of ~30-150): $v_j \equiv s \equiv (art, n, aux, vi)$
  - Representation: labeled examples $\langle (w_1, w_2, \ldots, w_n), s \rangle$
  - Return: classifier $f: X \rightarrow V$ that tags $x \equiv (w_1, w_2, \ldots, w_n)$
  - Applications: WSD, dialogue acts (e.g., "That sounds OK to me." $\rightarrow$ *ACCEPT*)
- Solution Approaches: Use Transformation-Based Learning (TBL)
  - [Brill, 1995]: TBL - mistake-driven algorithm that produces sequences of rules
    - Each rule of the form (t_i, v): a test condition (constructed attribute) and a tag
    - t_i: "w occurs within $\pm k$ words of w_i" (context words); collocations (windows)
  - For more info: see [Roth, 1998], [Samuel, Carberry, Vijay-Shankar, 1998]
- Recent Research
  - E. Brill's page: http://www.cs.jhu.edu/~brill/
  - K. Samuel's page: http://www.eecis.udel.edu/~samuel/work/research.html

NLP Applications: Intelligent Web Searching

- Problem Definition
  - One role of learning: produce classifiers for web documents (see [Pratt, 1999])
  - Typical WWW engines: *Lycos*, *Excite*, *Hotbot*, *Webcrawler*, *AltaVista*
  - Searchable and browsable engines (taxonomies): *Yahoo*, *Einet Galaxy*
- Key Research Issue
  - Complex query-based searches
  - e.g., medical informatics DB: *"What are the complications of mastectomy?"*
  - Applications: online information retrieval, web portals (customization)
- Solution Approaches
  - Dynamic categorization [Pratt, 1997]
  - Hierachical Distributed Dynamic Indexing [Pottenger *et al*, 1999]
  - Neural hierarchical dynamic indexing
- Recent Research
  - W. Pratt's research home page: http://www.ics.uci.edu/~pratt/
  - W. Pottenger's research home page: http://www.ncsa.uiuc.edu/~billp/

NLP Applications: Info Retrieval (IR) and Digital Libraries

- Information Retrieval (IR)
  - One role of learning: produce classifiers for documents (see [Sahami, 1999])
  - Query-based search engines (e.g., for WWW: *AltaVista*, *Lycos*, *Yahoo*)
  - Applications: bibliographic searches (citations, patent intelligence, etc.)
- Bayesian Classification: Integrating Supervised and Unsupervised Learning
  - Unsupervised learning: organize collections of documents at a "topical" level
  - e.g., *AutoClass* [Cheeseman *et al*, 1988]; self-organizing maps [Kohonen, 1995]
  - More on this topic (document clustering) soon
- Framework Extends Beyond Natural Language
  - Collections of images, audio, video, other media
  - Five Ss : Source, Stream, Structure, Scenario, Society
  - Book on IR [vanRijsbergen, 1979]: http://www.dcs.gla.ac.uk/Keith/Preface.html
- Recent Research
  - M. Sahami's page (Bayesian IR): http://robotics.stanford.edu/users/sahami
  - Digital libraries (DL) resources: http://fox.cs.vt.edu

Terminology

- Simple Bayes, *aka* Naïve Bayes
  - Zero counts: case where an attribute value never occurs with a label in D
  - No match approach: assign an $\varepsilon \equiv c/m$ probability to $P(x_{ik} \mid v_j)$
  - m-estimate *aka* Laplace approach: assign a Bayesian estimate to $P(x_{ik} \mid v_j)$
- Learning in Natural Language Processing (NLP)
  - Training data: text corpora (collections of representative documents)
  - Statistical Queries (SQ) oracle: answers queries about $P(x_{ik}, v_j)$ for $x \sim \mathcal{D}$
  - Linear Statistical Queries (LSQ) algorithm: classification using $f(oracle\ response)$
    - Includes: Naïve Bayes, BOC
    - Other examples: Hidden Markov Models (HMMs), maximum entropy
  - Problems: word sense disambiguation, part-of-speech tagging
  - Applications
    - Spelling correction, conversational agents
    - Information retrieval: web and digital library searches

Summary Points

- More on Simple Bayes, *aka* Naïve Bayes
  - More examples
  - Classification: choosing between two classes; general case
  - Robust estimation of probabilities: SQ
- Learning in Natural Language Processing (NLP)
  - Learning over text: problem definitions
  - Statistical Queries (SQ) / Linear Statistical Queries (LSQ) framework
    - Oracle
    - Algorithms: search for h using only (L)SQs
  - Bayesian approaches to NLP
    - Issues: word sense disambiguation, part-of-speech tagging
    - Applications: spelling; reading/posting news; web search, IR, digital libraries
- Next Week: Section 6.11, Mitchell; Pearl and Verma
  - Read: Charniak tutorial, "Bayesian Networks without Tears"
  - Skim: Chapter 15, Russell and Norvig; Heckerman slides