Just a little more on Smoothing

This black art is why NLP is taught in the engineering school – Jason Eisner
Review: Naive Bayes Classifiers for Text:

- As a generative model:
  1. Randomly pick a category $c$ according to $P(c)$
  2. For a document of length $N$, for each word $i$:
     1. Generate $word_i$ according to $P(w_i | c)$

\[
P(c, D = \langle w_1, w_2, \ldots, w_N \rangle) = P(c) \prod_{i=1}^{N} P(w_i | c)
\]

- This is a Naïve Bayes classifier for multinomial variables.
- *Word order is assumed irrelevant - we use same parameters for each position*
  - This *bag of words* model views document not as an ordered list of words, but as a *multiset*
    — Better representation: A vector where the $i^{th}$ component is the count in the document of the $i^{th}$ word in the vocabulary
Review: Naïve Bayes: Learning (First attempt)

- Extract $V$, the vocabulary set of all words within the training corpus.

- Calculate required estimates of $P(c_j)$ and $P(w_i | c_j)$ terms,
  - For each $c_j \in C$ calculate the MLE estimate
    \[
    \hat{P}(c_j) = \frac{\text{count}_{\text{docs}}(c_j)}{|\text{docs}|}
    \]
    where $\text{count}_{\text{docs}}(c_j)$ is the number of docs in $c_j$ and $|\text{docs}|$ is the total number of docs
  - For each word $w_i \in V$ and $c_j \in C$, calculate the MLE estimate
    \[
    \hat{P}(w | c_j) = \frac{\text{count}_{c_j}(w)}{\left( \sum_{w' \in V} \text{count}_{c_j}(w') \right)}
    \]
    where $\text{count}_{c}(x)$ is the number of tokens in all documents of category $c$
Review: Naïve Bayes: Classifying

- Compute $c_{NB}$ for a doc of length N using either

$$c_{NB} = \arg \max_c \hat{P}(c) \prod_{i=1}^{N} \hat{P}(w_i | c)$$

$$c_{NB} = \arg \max_c \hat{P}(c) \prod_{w \in V} \hat{P}(w | c)^{\text{count}(w)}$$

where $\text{count}(w)$: the number of times word $w$ occurs in $doc$

(The two are equivalent..)
Review: Naïve Bayes: Classifying

BUT: for some \( c \), for some \( w \) in a test doc,
if \( w \) never appeared in any doc in \( c \) in the training set,
then the MLS estimate \( \hat{P}(w \mid c) = 0 \)

and thus if \( w \) is in the test doc

\[
\hat{P}(c) \prod_{i=1}^{N} \hat{P}(w_i \mid c) = 0
\]

which is almost certainly wildly wrong
Naïve Bayes: Learning (Second attempt)

- Some *smoothing* must be done over $V$, the set of vocabulary items that saves some probability mass for *unknown words*.

- Idea:
  
  Add an additional fractional count of $\alpha$ to each word and to a single new word $\text{UNK}$ (for unknown), that occurs nowhere within $V$, so our new vocabulary is $V \cup \{ \text{UNK} \}$

  Map all unknown words in documents *to be classified* (test documents) to $\text{UNK}$

- Compute

  $\hat{P}(w | c) = \frac{\text{count}_c(w) + \alpha}{\left( \sum_{w' \in V} \text{count}_c(w') \right) + \alpha(|V| + 1)}$  

  $\hat{P}(\langle \text{UNK} \rangle | c) = \frac{\alpha}{\left( \sum_{w' \in V} \text{count}_c(w') \right) + \alpha(|V| + 1)}$

  where $\alpha$, $0 \leq \alpha \leq 1$, is the smoothing constant, and $\text{count}_c(w)$ is the count of the word $w$ in the set of documents of category $c$
Why $\alpha \leq 1$??

Related question: Why assume $UNK$ occurs < 1 time?

Because words that occurred 0 times in our data should have a count less than words that occurred once!!
Smoothing more generally

To fix estimating $\hat{P}(w) = 0$ given MLE, we can smooth the data.

1. Assume we know how many types *never* occur in the data.
2. Steal probability mass from types that occur at least once.
3. Distribute this probability mass over the types that never occur.

![Diagram showing n-grams (ordered by frequency) with zero frequencies for unseen data.](image-url)
Smoothing

….is like Robin Hood:

• it steals from the rich
• and gives to the poor
Add-One Smoothing

- **Pro:** Very simple technique

- **Cons:**
  - Probability of frequent words is underestimated
  - Probability of rare (or unseen) words is overestimated
  - Therefore, too much probability mass is shifted towards words
  - All unseen words are smoothed in the same way

- **Using a smaller $\alpha$ improves things but only some**

- **But for our purposes this will be adequate for now**
The smaller the count the worse the estimate of what the count will be in a new, unseen data set.

Data from Mark Liberman
Estimating Counts by Split Halves

Q: What is the average count in the 2nd half for all words with the same count in the first half?

<table>
<thead>
<tr>
<th>Half 1</th>
<th>Half 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1.60491</td>
</tr>
<tr>
<td>1</td>
<td>0.639544</td>
</tr>
<tr>
<td>2</td>
<td>1.59014</td>
</tr>
<tr>
<td>3</td>
<td>2.55045</td>
</tr>
<tr>
<td>4</td>
<td>3.49306</td>
</tr>
<tr>
<td>5</td>
<td>4.45996</td>
</tr>
<tr>
<td>6</td>
<td>5.23295</td>
</tr>
<tr>
<td>7</td>
<td>6.28311</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Half 1</th>
<th>Half 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>8</td>
<td>7.53499</td>
</tr>
<tr>
<td>9</td>
<td>8.27005</td>
</tr>
<tr>
<td>10</td>
<td>9.50197</td>
</tr>
<tr>
<td>11</td>
<td>10.0348</td>
</tr>
<tr>
<td>12</td>
<td>11.2292</td>
</tr>
<tr>
<td>13</td>
<td>12.7391</td>
</tr>
<tr>
<td>14</td>
<td>12.5298</td>
</tr>
<tr>
<td>15</td>
<td>14.1646</td>
</tr>
</tbody>
</table>

Observations: The Maximum Likelihood Estimator for items with

- Small count is *much too high by %*
- Count > 0 is *always somewhat too high by %*
- Count 0 is *very bad (BUT this is spurious… we counted wrong!)*
**Key idea: Deleted Estimation**

- For all words with a given count in the first half, just use the average count in the 2nd half of those same words as the smoothed count.
- Better: do both halves then average.
- Key Idea:
  - *Pool all items with the same frequency*
  - Compute average counts across halves.
  - *Replace raw counts with the smoothed estimates*

<table>
<thead>
<tr>
<th>Half 1</th>
<th>Half 2</th>
<th>Half 1</th>
<th>Half 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1.60491</td>
<td>8</td>
<td>7.53499</td>
</tr>
<tr>
<td>1</td>
<td>0.639544</td>
<td>9</td>
<td>8.27005</td>
</tr>
<tr>
<td>2</td>
<td>1.59014</td>
<td>10</td>
<td>9.50197</td>
</tr>
<tr>
<td>3</td>
<td>2.55045</td>
<td>11</td>
<td>10.0348</td>
</tr>
<tr>
<td>4</td>
<td>3.49306</td>
<td>12</td>
<td>11.2292</td>
</tr>
<tr>
<td>5</td>
<td>4.45996</td>
<td>13</td>
<td>12.7391</td>
</tr>
<tr>
<td>6</td>
<td>5.23295</td>
<td>14</td>
<td>12.5298</td>
</tr>
<tr>
<td>7</td>
<td>6.28311</td>
<td>15</td>
<td>14.1646</td>
</tr>
</tbody>
</table>
Key Idea: Witten-Bell Discounting

- **KEY IDEA:** a zero-frequency word is just an event that hasn’t happened yet.

- **How often do zero-frequency words first appear in this context?**
  - Just as often as zero-frequency words have become 1-frequency words in this context.
  - We can use the MLE estimate by counting the number of times we first encountered a type.

- **If** \( P(w_i \mid w_{i-k}, \ldots, w_{i-1}) = 0 \),
  - then the estimate \( P_{WB}(w_i \mid w_{i-k}, \ldots, w_{i-1}) \) is **higher**
    if \( w_{i-k}, \ldots, w_{i-1} \) occurs with **many** different words \( w_i \)

- Dan Bikel calls this “diversity smoothing”
Backoff: Key Idea

- Why are we treating all novel events as the same?

- $p(zygote \mid \text{see the})$ vs. $p(baby \mid \text{see the})$
  - Suppose both trigrams have zero count

- **baby beats zygote** as a unigram
- **the baby beats the zygote** as a bigram
- Shouldn’t **see the baby beat see the zygote**?

(This and next slide adapted from Jason Eisner)
The Real Stuff

• Current systems use either
  • “modified Witten-Bell” [Google it]
    — interpolates $P_{WB}$ with a particular backoff distribution, with $\lambda$ computed according to a particular formula.
  • “modified Kneser Ney”

• For simple language modeling, Kneser Ney works better (see Goodman, Goodman & Chen)

• For more complex tasks, Witten-Bell does well

• Which one performs better depends on
  • The application
  • The user’s secret sauce…
A Brief Introduction to Bayesian Networks

AIMA 14.1-14.3
Bayesian networks

• A simple, graphical notation for conditional independence assertions

• and hence for compact specification of full joint distributions

• Syntax:
  – A set of nodes, one per random variable
  – A set of directed edges (link \approx "directly influences"), yielding a directed acyclic graph
  – A conditional distribution for each node given its parents: \( P(X_i | \text{Parents}(X_i)) \)

• In the simplest case, conditional distribution represented as a conditional probability table (CPT) giving the distribution over \( X_i \) for each combination of parent values
The Required-By-Law Burglar Example

I'm at work, and my neighbor John calls, but my neighbor Mary doesn't call. They usually call if my burglar alarm goes off, and occasionally otherwise. Sometimes the alarm is set off by minor earthquakes. Is there a burglar?

Variables: **Burglary, Earthquake, Alarm, JohnCalls, MaryCalls**

Network topology reflects *causal* knowledge:

1. A burglar can cause the alarm to go off
2. An earthquake can cause the alarm to go off
3. The alarm can cause Mary to call
4. The alarm can cause John to call
Belief network for example

Network topology reflects "causal" knowledge:
1. A burglar can cause the alarm to go off
2. An earthquake can cause the alarm to go off
3. The alarm can cause Mary to call (with prob ≈ .70)
4. The alarm can cause John to call (with prob ≈ .90)
Belief network for example

Network topology reflects "causal" knowledge:

1. A burglar can cause the alarm to go off (with probability $\approx .94$)
2. An earthquake can cause the alarm to go off (with prob $\approx .29$)
3. The alarm can cause Mary to call
4. The alarm can cause John to call
Belief network for example

Unconditional probabilities reflect priors of those events:

1. The probability of a burglary (per day?) is .001
2. The probability of an earthquake is .002
Belief network with all (conditional) probability tables

Makes it easy to compute answers to questions like:

- How likely is John to call if there’s an earthquake
- How likely is Mary to call on a random day?
Semantics

- *Local* semantics give rise to *global* semantics
- Local semantics: given its parents, each node is *conditionally independent* of its other ancestors

**REVIEW (clarity is crucial):**
- A and B are *conditionally independent given C* iff
  - \( P(A \mid B, C) = P(A \mid C) \)
  - Alternatively, \( P(B \mid A, C) = P(B \mid C) \)
  - Alternatively, \( P(A \land B \mid C) = P(A \mid C) \times P(B \mid C) \)
- Review: Toothache (T), Spot in Xray (X), Cavity (C)
  - None of these propositions are independent of one other
  - But *T & X are conditionally independent given C*
Semantics

- *Local* semantics give rise to *global* semantics
  - Local semantics: *given its parents, each node is conditionally independent of its other ancestors*
  - Global semantics (why?):

$$P(X_1, \ldots, X_n) = \prod_{i=1}^{n} P(X_i | Parents(X_i))$$

$$P(J, M, A, \neg B, \neg E) = P(\neg B)P(\neg E)P(A | \neg B, \neg E)P(J | A)P(M | A)$$

Generalizes Naïve Bayes
Naïve Bayes as a Bayesian Network

Flu

P(Flu) = 0.10

X₁
runnynose

P(Flu, X₁, ..., X₅) = P(Flu) \cdot P(X₁ \mid Flu) \cdot \cdots \cdot P(X₅ \mid Flu)
Belief Network Construction Algorithm

1. Choose some ordering of variables $X_1, \ldots, X_n$

2. For $i = 1$ to $n$
   a. add $X_i$ to the network
   b. select parents from $X_1, \ldots, X_{i-1}$ such that

   \[ P(X_i|\text{Parents}(X_i)) = P(X_i|X_1, \ldots, X_{i-1}) \]

3. Yields the smallest networks when *causers* *(diseases)* precede consequences *(symptoms)*
Construction Example

Suppose we choose the ordering M, J, A, B, E

\[ P(J|M) = P(J) \? \quad \text{No} \]
\[ P(A|J,M) = P(A) \? \quad P(A|J,M) = P(A|J) \? \quad \text{No} \]
\[ P(B|A,J,M) = P(B) \? \quad \text{No} \]
\[ P(B|A,J,M) = P(B|A) \? \quad \text{Yes} \]
\[ P(E|B,A,J,M) = P(E|A) \? \quad \text{No} \]
\[ P(E|B,A,J,M) = P(E|A,B) \? \quad \text{Yes} \]
1) order the variables (any ordering will do) A, B, C, D, E,..
2) pop the first item off the list, and test to see what a minimal set of parents is
i) add A (trivial)
ii) add B
   test: is it true that $P(B|A) = P(B|\sim A) = P(B)$
   if not, then add a link from A to B
iii) add C
   test: is $P(C) = P(C|A,B) = P(C|\sim A,B) = P(C|B,\sim A) = P(C|\sim A,\sim B)$
   if so C is not linked to anything
   test: is $P(C) = P(C|A) = P(C|\sim A)$ if so then there is no link from A to C
   test: is $P(C) = P(C|B) = P(C|\sim B)$ if so then there is no link from B to C
   If some of the above tests fail, then there there will be links.
One then has to test whether A alone, or B alone is sufficient?
   test: is $P(C|B) = P(C|B,A) = P(C|B, \sim A)$ if so then we only need a link from B to C
   test: is $P(C|A) = P(C|A,B) = P(C|A, \sim B)$ if so then we only need a link from A to C
iv) add D
   things keep getting uglier, but the same idea continues.
Lessons from the Example

- Network less compact: 13 numbers (compared to 10)
- Ordering of variables can make a big difference!
- Intuitions about causality useful
How to build a belief net?

- **Structure:**
  - Ask people: Try to model causality in structure
    - Real causal structures yield the best networks
  - Automatically: How to cast finding structure as search?

- **Conditional Probability Tables**
  - Ask people: Get approximate probabilities from experts
  - Automatically: Estimate probabilities from data
Belief Net Extensions

- Hidden variables
- Decision (action) variables
- Continuous distributions
- “Plate” models
  - E.g. Latent Dirichlet Allocation (LDA)
- Dynamical Belief Nets (DBNs)
  - E.g. Hidden Markov Models (HMMs)
Belief Networks and Hidden Variables: An Example
Compact Conditional Probability Tables

- CPT size grows exponentially; continuous variables have infinite CPTs!
- Solution: compactly defined canonical distributions
  - E.g., Boolean functions
  - Gaussian and other standard distributions
Summary

- **Belief nets provide a compact representation of joint probability distributions**
  - Guaranteed to be a consistent specification
  - Graphically show conditional independence
  - Can be extended to include “action” nodes

- **Networks that capture causality tend to be sparser**

- **Estimating belief nets is easy if everything is observable**
  - But requires serious search if the network structure is not known

- **Not covered: Probabilistic inference algorithms given belief nets**