Introduction to Computational Linguisyics

http://www-rohan.sdsu.edu/~gawron/compling

Naive Bayes

Gawron

Department of Linguistics, San Diego State University

2011-04-08

Overview

1. Sense disambiguation
2. Naive Bayes
3. NB theory

An ambiguous word

The noun "bass" has 8 senses in WordNet.

1. bass^{1} - (the lowest part of the musical range)
2. bass^{2}, bass part^{1} - (the lowest part in polyphonic music)
3. bass^{3}, basso^{1} - (an adult male singer with the lowest voice)
4. sea bass^{1}, bass^{4} - (the lean flesh of a saltwater fish of the family Serranidae)
5. freshwater bass^{1}, bass^{5} - (any of various North American freshwater fish with lean flesh (especially of the genus Micropterus))
6. bass^{6}, bass voice^{1}, basso^{2} - (the lowest adult male singing voice)
7. bass^{7} - (the member with the lowest range of a family of musical instruments)
8. bass^{8} - (nontechnical name for any of numerous edible marine and freshwater spiny-finned fishes)

The adjective "bass" has 1 sense in WordNet.

1. bass^{1}, deep^{6} - (having or denoting a low vocal or instrumental range) *"a deep voice"; "a bass voice is lower than a baritone voice"; "a bass clarinet"*

Examples

- He plays the bass well.
- He caught a huge bass.
- They served fried bass for lunch.
- She carried her bass clarinet into class.
- Charlie is the bass in a barbershop quartet.

The Naive Bayes classifier

- The Naive Bayes classifier is a probabilistic classifier.
- We will estimate the probability of a word like *bass* in a document d being a use of word sense s as follows:

$$P(s \mid d) \propto P(s) \prod_{1 \leq k \leq n_d} P(t_k \mid s)$$

  where n_d is the number of **context features** in document d, taken from a set of context features (words) for disambiguating *bass*.
- $P(t_k|s)$ is the conditional probability of context feature t_k occurring in the same document with sense s.
- $P(t_k|s)$ as a measure of how much evidence t_k contributes that s is the correct sense.
- $P(s)$ is the prior probability of sense s.
- If a context does not provide clear evidence for one sense vs. another, we choose the s with highest $P(s)$.

Maximum a posteriori class

- Our goal in Naive Bayes classification is to find the "best" class.
- The best class is the most likely or maximum a posteriori (MAP) sense s_{map}:

$$s_{\text{map}} = \arg\max_{s \in \mathbb{S}} \hat{P}(s|w) = \arg\max_{s \in \mathbb{S}} \hat{P}(s) \prod_{1 \leq k \leq n_d} \hat{P}(t_k|s)$$

- Note that

$$P(s) \prod_{1 \leq k \leq n_d} P(t_k \mid s) \tag{1}$$

is **not** the conditional probability, $P(s \mid d)$. What it actually is is the joint probability of the sense and the document.

$$P(s,\, d) = P(s) \prod_{1 \leq k \leq n_d} P(t_k \mid s)$$

- For now, the conditional probability is not needed, because the joint probability, which is easier to estimate, is **proportional** to it.

Taking the log

- Multiplying lots of small probabilities can result in floating point underflow.
- Since $\log(xy) = \log(x) + \log(y)$, we can sum log probabilities instead of multiplying probabilities.
- Since log is a monotonic function, the class with the highest score does not change.
- So what we usually compute in practice is:

$$c_{\text{map}} = \arg\max_{s \in \mathbb{C}} [\log \hat{P}(s) + \sum_{1 \leq k \leq n_d} \log \hat{P}(t_k|s)]$$

Naive Bayes classifier

Naive Bayes classifier

- Classification rule:

$$c_{\text{map}} = \arg\max_{s \in \mathbb{C}} [\log \hat{P}(s) + \sum_{1 \leq k \leq n_d} \log \hat{P}(t_k|s)]$$

Naive Bayes classifier

- Classification rule:

$$c_{\text{map}} = \arg\max_{s \in \mathbb{C}} \left[\log \hat{P}(s) + \sum_{1 \le k \le n_d} \log \hat{P}(t_k|s) \right]$$

- Simple interpretation:

Naive Bayes classifier

- Classification rule:

$$c_{\text{map}} = \arg\max_{s \in \mathbb{C}} [\log \hat{P}(s) + \sum_{1 \leq k \leq n_d} \log \hat{P}(t_k|s)]$$

- Simple interpretation:
  - Each conditional parameter $\log \hat{P}(t_k|s)$ is a weight that indicates how good an indicator t_k is for s.

Naive Bayes classifier

- Classification rule:

$$c_{\text{map}} = \arg\max_{s \in \mathbb{C}} \left[\log \hat{P}(s) + \sum_{1 \leq k \leq n_d} \log \hat{P}(t_k|s) \right]$$

- Simple interpretation:
  - Each conditional parameter $\log \hat{P}(t_k|s)$ is a weight that indicates how good an indicator t_k is for s.
  - The prior $\log \hat{P}(s)$ is a weight that indicates the relative frequency of s.

Naive Bayes classifier

- Classification rule:

$$c_{\text{map}} = \arg\max_{s \in \mathbb{C}} \left[\log \hat{P}(s) + \sum_{1 \le k \le n_d} \log \hat{P}(t_k|s) \right]$$

- Simple interpretation:
  - Each conditional parameter $\log \hat{P}(t_k|s)$ is a weight that indicates how good an indicator t_k is for s.
  - The prior $\log \hat{P}(s)$ is a weight that indicates the relative frequency of s.
  - The sum of log prior and term weights is then a measure of how much evidence there is for the document being in the class.

Naive Bayes classifier

- Classification rule:

$$c_{\text{map}} = \arg\max_{s \in \mathbb{C}} \left[\log \hat{P}(s) + \sum_{1 \le k \le n_d} \log \hat{P}(t_k|s)\right]$$

- Simple interpretation:
  - Each conditional parameter $\log \hat{P}(t_k|s)$ is a weight that indicates how good an indicator t_k is for s.
  - The prior $\log \hat{P}(s)$ is a weight that indicates the relative frequency of s.
  - The sum of log prior and term weights is then a measure of how much evidence there is for the document being in the class.
  - We select the class with the most evidence.

Parameter estimation take 1: Maximum likelihood

Parameter estimation take 1: Maximum likelihood

- Estimate parameters $\hat{P}(s)$ and $\hat{P}(t_k|s)$ from train data: How?

Parameter estimation take 1: Maximum likelihood

- Estimate parameters $\hat{P}(s)$ and $\hat{P}(t_k|s)$ from train data: How?
- Prior:

$$\hat{P}(s) = \frac{N_s}{N}$$

Parameter estimation take 1: Maximum likelihood

- Estimate parameters $\hat{P}(s)$ and $\hat{P}(t_k|s)$ from train data: How?
- Prior:

$$\hat{P}(s) = \frac{N_s}{N}$$

- N_s: number of tokens of word w using sense s; N: total number of tokens of word w.

Parameter estimation take 1: Maximum likelihood

- Estimate parameters $\hat{P}(s)$ and $\hat{P}(t_k|s)$ from train data: How?
- Prior:

$$\hat{P}(s) = \frac{N_s}{N}$$

- N_s: number of tokens of word w using sense s; N: total number of tokens of word w.
- Conditional probabilities:

$$\hat{P}(t|s) = \frac{T_{st}}{\sum_{t' \in V} T_{st'}}$$

Parameter estimation take 1: Maximum likelihood

- Estimate parameters $\hat{P}(s)$ and $\hat{P}(t_k|s)$ from train data: How?
- Prior:

$$\hat{P}(s) = \frac{N_s}{N}$$

- N_s: number of tokens of word w using sense s; N: total number of tokens of word w.
- Conditional probabilities:

$$\hat{P}(t|s) = \frac{T_{st}}{\sum_{t' \in V} T_{st'}}$$

- T_{st} is the number of tokens of t in training data with sense s (includes multiple occurrences)

Parameter estimation take 1: Maximum likelihood

- Estimate parameters $\hat{P}(s)$ and $\hat{P}(t_k|s)$ from train data: How?
- Prior:

$$\hat{P}(s) = \frac{N_s}{N}$$

- N_s: number of tokens of word w using sense s; N: total number of tokens of word w.
- Conditional probabilities:

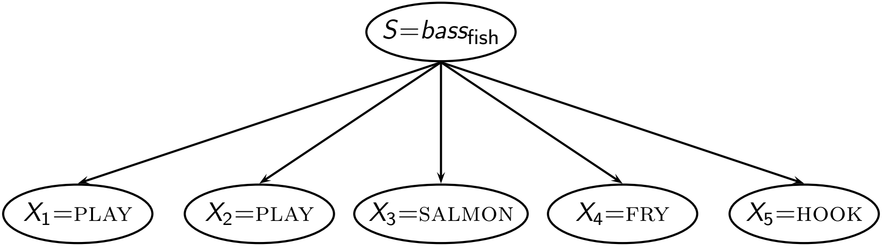
$$\hat{P}(t|s) = \frac{T_{st}}{\sum_{t' \in V} T_{st'}}$$

- T_{st} is the number of tokens of t in training data with sense s (includes multiple occurrences)
- We've made a Naive Bayes independence assumption here:

$$\hat{P}(t_j, t_k|s) \;\; = \;\; \hat{P}(t_j|s)\hat{P}(t_k|s, t_j) \;\; = \;\; \hat{P}(t_j|s)\hat{P}(t_k|s)$$

The problem with maximum likelihood estimates: Zeros

The problem with maximum likelihood estimates: Zeros

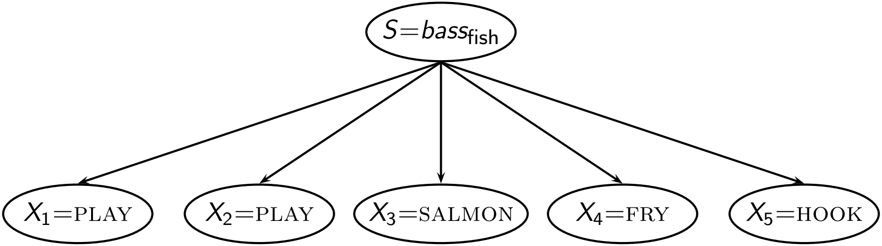


$$
\begin{aligned}
P(bass_{\text{fish}}|d) \quad \propto \quad & P(bass_{\text{fish}}) \cdot P(\text{PLAY}|bass_{\text{fish}}) \cdot P(\text{PLAY}|bass_{\text{fish}}) \\
& \cdot P(\text{SALMON}|bass_{\text{fish}}) \cdot P(\text{FRY}|bass_{\text{fish}}) \cdot P(\text{HOOK}|bass_{\text{fish}})
\end{aligned}
$$

- If HOOK never occurs with sense $bass_{\text{fish}}$ in the training set:

$$\hat{P}(\text{HOOK}|bass_{\text{fish}}) = \frac{T_{bass_{\text{fish}},\text{HOOK}}}{\sum_{t' \in V} T_{bass_{\text{fish}},t'}} = \frac{0}{\sum_{t' \in V} T_{bass_{\text{fish}},t'}} = 0$$

The problem with maximum likelihood estimates: Zeros



$$P(bass_{\text{fish}}|d) \propto P(bass_{\text{fish}}) \cdot P(\text{PLAY}|bass_{\text{fish}}) \cdot P(\text{PLAY}|bass_{\text{fish}}) \cdot P(\text{SALMON}|bass_{\text{fish}}) \cdot P(\text{FRY}|bass_{\text{fish}}) \cdot P(\text{HOOK}|bass_{\text{fish}})$$

- If HOOK never occurs with sense $bass_{\text{fish}}$ in the training set:

$$\hat{P}(\text{HOOK}|bass_{\text{fish}}) = \frac{T_{bass_{\text{fish}},\text{HOOK}}}{\sum_{t' \in V} T_{bass_{\text{fish}},t'}} = \frac{0}{\sum_{t' \in V} T_{bass_{\text{fish}},t'}} = 0$$

The problem with maximum likelihood estimates: Zeros (cont)

- If there were no occurrences of HOOK in documents in class $bass_{\text{fish}}$, we'd get a zero estimate:

$$\hat{P}(\text{HOOK}|bass_{\text{fish}}) = \frac{T_{bass_{\text{fish}},\text{HOOK}}}{\sum_{t' \in V} T_{bass_{\text{fish}},t'}} = 0$$

- $\rightarrow$ We will get $P(bass_{\text{fish}}|w) = 0$ for any document that contains hook! No matter how much positive evidence there is for one of the senses.
- Zero probabilities cannot be conditioned away.

To avoid zeros: Add-one smoothing

To avoid zeros: Add-one smoothing

- Before:

$$\hat{P}(t|s) = \frac{T_{st}}{\sum_{t' \in V} T_{st'}}$$

To avoid zeros: Add-one smoothing

- Before:

$$\hat{P}(t|s) = \frac{T_{st}}{\sum_{t' \in V} T_{st'}}$$

- Now: Add one to each count to avoid zeros:

$$\hat{P}_{sm}(t|s) = \frac{T_{st} + 1}{\sum_{t' \in V_w} (T_{ct'} + 1)} = \frac{T_{st} + 1}{(\sum_{t' \in V} T_{st'}) + |V_w|}$$

To avoid zeros: Add-one smoothing

- Before:

$$\hat{P}(t|s) = \frac{T_{st}}{\sum_{t' \in V} T_{st'}}$$

- Now: Add one to each count to avoid zeros:

$$\hat{P}_{sm}(t|s) = \frac{T_{st} + 1}{\sum_{t' \in V_w} (T_{ct'} + 1)} = \frac{T_{st} + 1}{(\sum_{t' \in V} T_{st'}) + |V_w|}$$

- V_w is the set of context features for w (*bass*, the word we are disambiguating), and $|V_w|$ is the number of such features.

Naive Bayes: Summary

Naive Bayes: Summary

- Estimate parameters from the training corpus using add-one smoothing

Naive Bayes: Summary

- Estimate parameters from the training corpus using add-one smoothing
- For a new document, for each class, compute sum of (i) log of prior and (ii) logs of conditional probabilities of the terms

Naive Bayes: Summary

- Estimate parameters from the training corpus using add-one smoothing
- For a new document, for each class, compute sum of (i) log of prior and (ii) logs of conditional probabilities of the terms
- Assign the document to the class with the largest score

Naive Bayes: Training

TrainMultinomialNB($\mathbb{S}, \mathbb{D}, w$)

$V \leftarrow$ ExtractFeatures($\mathbb{D}, w$)
$N \leftarrow$ CountOccurrences($\mathbb{D}, w$)
for each $s \in \mathbb{S}$
do $N_s \leftarrow$ CountOccurrencesOfSense($\mathbb{D}, s$)
    $prior[s] \leftarrow N_s / N$
    $text_s \leftarrow$ ConcatTextOfAllContextsWithSense($\mathbb{D}, s$)
    for each $t \in V$
    do $T_{st} \leftarrow$ CountTokensOfTerm($text_s, t$)
    for each $t \in V$
    do $condprob[t][s] \leftarrow \frac{T_{st}+1}{\sum_{t'}(T_{st'}+1)}$
return $V, prior, condprob$

Naive Bayes: Testing

ApplyMultinomialNB($\mathbb{S}, V, prior, condprob, d$)

$W \leftarrow$ ExtractFeatureTokensFromDoc(V, d)
for each $s \in \mathbb{S}$
do $score[c] \leftarrow \log prior[s]$
    for each $t \in W$
    do $score[s] += \log condprob[t][s]$
return $\arg\max_{s \in \mathbb{S}} score[s]$

Exercise

For our feature set, we often choose the n most frequent content words in our document set, retaining duplicates, leaving out w (*bass*). Here we choose *fry, play, clarinet, salmon, hook*, and *guitar*. Note that *guitar* occurs in neither the test set nor training set. Assume the context window for training

	docID	words in context	in $s = bass_{\text{fish}}$?
training set	1	fry fry	yes
	2	play play clarinet	no
	3	salmon fry	yes
	4	play	yes
test set	5	play play fry hook play play play play play	?

- Estimate parameters of Naive Bayes classifier
- Classify test document

Unsmoothed Example: Parameter estimates

Priors: $\hat{P}(s) = 3/4$ and $\hat{P}(\overline{s}) = 1/4$
Conditional probabilities:

$$
\begin{aligned}
\hat{P}(\textsc{fry}|s) &= 3/5 \\
\hat{P}(\textsc{salmon}|s) = \hat{P}(\textsc{play}|s) &= 1/5 \\
\hat{P}(\textsc{hook}|s) &= 0/5 = 0 \\
\hat{P}(\textsc{clarinet}|\overline{s}) &= 1/3 \\
\hat{P}(\textsc{play}|\overline{s}) &= 2/3
\end{aligned}
$$

The denominators are 5 and 3 because the lengths of $text_s$ and $text_{\overline{s}}$ are 5 and 3, respectively.

Smoothing Example: Parameter estimates

Priors: $\hat{P}(s) = 3/4$ and $\hat{P}(\overline{s}) = 1/4$
Conditional probabilities:

$$\begin{aligned}
\hat{P}_{sm}(\text{FRY}|s) &= (3+1)/(5+6) = 4/11 \\
\hat{P}_{sm}(\text{SALMON}|s) = \hat{P}_{sm}(\text{PLAY}|s) &= (1+1)/(5+6) = 2/11 \\
\hat{P}_{sm}(\text{HOOK}|s) &= (0+1)/(5+6) = 1/11 \\
\hat{P}_{sm}(\text{CLARINET}|\overline{s}) &= (1+1)/(3+6) = 2/9 \\
\hat{P}_{sm}(\text{PLAY}|\overline{s}) &= (2+1)/(3+6) = 3/9 = 1/3
\end{aligned}$$

The denominators are $(5+6)$ and $(3+6)$ because the lengths of $text_s$ and $text_{\overline{s}}$ are 5 and 3, respectively, and because the constant $|V_w|$ is 6 as the feature set consists of six terms.

Example: Classification

$$
\begin{aligned}
\hat{P}_{sm}(s|d_5) &\propto 3/4 \cdot (2/11)^7 \cdot 4/11 \cdot 1/11 \approx 1.643 * 10^{-7} \\
\hat{P}_{sm}(\overline{s}|d_5) &\propto 1/4 \cdot (1/3)^7 \cdot 1/9 \cdot 1/9 \approx 2.389 * 10^{-7}
\end{aligned}
$$

Thus, the classifier assigns the test document to $s = \overline{bass_{\text{fish}}}$.
The reason for this classification decision is that the seven occurrences of the negative indicator PLAY in d_5 outweigh the occurrence the positive indicator FRY.
Note: the terms to the right of $\propto$ are what we were referring to as **joint probabilities** in first few slides.

Normalizing Joint probabilities

To get from joint probability $\hat{P}(s_1, t_{1,n_d})$ to the conditional probability $\hat{P}(s_1 \mid t_{1,n_d})$ we must normalize:

$$
\begin{array}{ll}
\text{(a)} & \hat{P}(s_1, t_{1,n_d}) = \hat{P}(s) \prod_{1 \leq k \leq n_d} \hat{P}(t_k|s_1) = 0.00099 \\
\text{(b)} & \hat{P}(s_1|t_{1,n_d}) = \hat{P}(s_1, t_{1,n_d}) / \hat{P}(d) \\
\text{(c)} & \hat{P}(s_2, t_{1,n_d}) = \hat{P}(s) \prod_{1 \leq k \leq n_d} \hat{P}(t_k|s_1) = 0.00001 \\
\text{(d)} & \hat{P}(s_2|t_{1,n_d}) = \hat{P}(s_2, t_{1,n_d}) / \hat{P}(d) \\
\text{(e)} & \hat{P}(s_1|t_{1,n_d}) + \hat{P}(s_2|t_{1,n_d}) = \frac{\hat{P}(s_1, t_{1,n_d}) + \hat{P}(s_2, t_{1,n_d})}{\hat{P}(d)} = 1.0 \\
\text{(f)} & \hat{P}(s_1, t_{1,n_d}) + \hat{P}(s_2, t_{1,n_d}) = \hat{P}(d) = 0.0001 \\
\text{(g)} & \hat{P}(s_1|t_{1,n_d}) = 0.99, \hat{P}(s_2|t_{1,n_d}) = 0.01.
\end{array}
$$

How to normalize

$$\hat{P}(s_1|t_{1,n_d}) = \frac{\hat{P}(s_1, t_{1,n_d})}{\sum_i \hat{P}(s_i, t_{1,n_d})}$$

In our example there are only two senses, so:

$$\hat{P}(s_1|t_{1,n_d}) = \frac{\hat{P}(s_1, t_{1,n_d})}{\hat{P}(s_1, t_{1,n_d}) + \hat{P}(s_2, t_{1,n_d})} = \frac{.00099}{.00099 + .00001} = \frac{.00099}{.001} = .99.$$

Since

$$\hat{P})(t_{1,n_d}) = \sum_{s_i} \hat{P}(s_i, t_{1,n_d}) = \hat{P}(s_1, t_{1,n_d}) + \hat{P}(s_2, t_{1,n_d})$$

this is just a rewrite of our definition of conditional probability:

$$\hat{P}(s_1|t_{1,n_d}) = \frac{\hat{P}(s_1, t_{1,n_d})}{\hat{P}(t_{1,n_d})}$$

Time complexity of Naive Bayes

Time complexity of Naive Bayes

mode	time complexity
training	$\Theta(\lvert\mathbb{D}\rvert L_{\text{ave}} + \lvert\mathbb{S}\rvert\lvert V\rvert)$
testing	$\Theta(L_{\text{a}} + \lvert\mathbb{S}\rvert M_{\text{a}}) = \Theta(\lvert\mathbb{S}\rvert M_{\text{a}})$

- L_{ave}: average length of a training doc, L_{a}: length of the test doc, M_{a}: number of distinct feature terms in the test doc, $\mathbb{D}$: training set, V: vocabulary, $\mathbb{S}$: set of senses

Time complexity of Naive Bayes

mode	time complexity
training	$\Theta(\vert\mathbb{D}\vert L_{\text{ave}} + \vert\mathbb{S}\vert\vert V\vert)$
testing	$\Theta(L_{\text{a}} + \vert\mathbb{S}\vert M_{\text{a}}) = \Theta(\vert\mathbb{S}\vert M_{\text{a}})$

- L_{ave}: average length of a training doc, L_{a}: length of the test doc, M_{a}: number of distinct feature terms in the test doc, $\mathbb{D}$: training set, V: vocabulary, $\mathbb{S}$: set of senses
- $\Theta(|\mathbb{D}|L_{\text{ave}})$ is the time it takes to compute all counts.

Time complexity of Naive Bayes

mode	time complexity
training	$\Theta(\lvert\mathbb{D}\rvert L_{\text{ave}} + \lvert\mathbb{S}\rvert\lvert V\rvert)$
testing	$\Theta(L_{\text{a}} + \lvert\mathbb{S}\rvert M_{\text{a}}) = \Theta(\lvert\mathbb{S}\rvert M_{\text{a}})$

- L_{ave}: average length of a training doc, L_{a}: length of the test doc, M_{a}: number of distinct feature terms in the test doc, $\mathbb{D}$: training set, V: vocabulary, $\mathbb{S}$: set of senses
- $\Theta(|\mathbb{D}|L_{\text{ave}})$ is the time it takes to compute all counts.
- $\Theta(|\mathbb{S}||V|)$ is the time it takes to compute the parameters from the counts.

Time complexity of Naive Bayes

mode	time complexity
training	$\Theta(\vert\mathbb{D}\vert L_{\mathrm{ave}} + \vert\mathbb{S}\vert\vert V\vert)$
testing	$\Theta(L_{\mathrm{a}} + \vert\mathbb{S}\vert M_{\mathrm{a}}) = \Theta(\vert\mathbb{S}\vert M_{\mathrm{a}})$

- L_{ave}: average length of a training doc, L_{a}: length of the test doc, M_{a}: number of distinct feature terms in the test doc, $\mathbb{D}$: training set, V: vocabulary, $\mathbb{S}$: set of senses
- $\Theta(|\mathbb{D}|L_{\mathrm{ave}})$ is the time it takes to compute all counts.
- $\Theta(|\mathbb{S}||V|)$ is the time it takes to compute the parameters from the counts.
- Generally: $|\mathbb{S}||V| < |\mathbb{D}|L_{\mathrm{ave}}$

Time complexity of Naive Bayes

mode	time complexity
training	$\Theta(\lvert\mathbb{D}\rvert L_{\mathrm{ave}} + \lvert\mathbb{S}\rvert\lvert V\rvert)$
testing	$\Theta(L_{\mathrm{a}} + \lvert\mathbb{S}\rvert M_{\mathrm{a}}) = \Theta(\lvert\mathbb{S}\rvert M_{\mathrm{a}})$

- L_{ave}: average length of a training doc, L_{a}: length of the test doc, M_{a}: number of distinct feature terms in the test doc, $\mathbb{D}$: training set, V: vocabulary, $\mathbb{S}$: set of senses
- $\Theta(\lvert\mathbb{D}\rvert L_{\mathrm{ave}})$ is the time it takes to compute all counts.
- $\Theta(\lvert\mathbb{S}\rvert\lvert V\rvert)$ is the time it takes to compute the parameters from the counts.
- Generally: $\lvert\mathbb{S}\rvert\lvert V\rvert < \lvert\mathbb{D}\rvert L_{\mathrm{ave}}$
- Test time is also linear (in the length of the test document).

Time complexity of Naive Bayes

mode	time complexity
training	$\Theta(\lvert\mathbb{D}\rvert L_{\mathrm{ave}} + \lvert\mathbb{S}\rvert\lvert V\rvert)$
testing	$\Theta(L_{\mathrm{a}} + \lvert\mathbb{S}\rvert M_{\mathrm{a}}) = \Theta(\lvert\mathbb{S}\rvert M_{\mathrm{a}})$

- L_{ave}: average length of a training doc, L_{a}: length of the test doc, M_{a}: number of distinct feature terms in the test doc, $\mathbb{D}$: training set, V: vocabulary, $\mathbb{S}$: set of senses
- $\Theta(\lvert\mathbb{D}\rvert L_{\mathrm{ave}})$ is the time it takes to compute all counts.
- $\Theta(\lvert\mathbb{S}\rvert\lvert V\rvert)$ is the time it takes to compute the parameters from the counts.
- Generally: $\lvert\mathbb{S}\rvert\lvert V\rvert < \lvert\mathbb{D}\rvert L_{\mathrm{ave}}$
- Test time is also linear (in the length of the test document).
- Thus: Naive Bayes is linear in the size of the training set (training) and the test document (testing). This is optimal.

Naive Bayes: Analysis

Naive Bayes: Analysis

- Now we want to gain a better understanding of the properties of Naive Bayes.

Naive Bayes: Analysis

- Now we want to gain a better understanding of the properties of Naive Bayes.
- We will formally derive the classification rule . . .

Naive Bayes: Analysis

- Now we want to gain a better understanding of the properties of Naive Bayes.
- We will formally derive the classification rule . . .
- . . . and state the assumptions we make in that derivation explicitly.

Derivation of Naive Bayes rule

Derivation of Naive Bayes rule

We want to find the class that is most likely given the document:

$$c_{\text{map}} = \underset{s \in \mathbb{C}}{\arg\max}\ P(s|d)$$

Apply Bayes rule $P(A|B) = \frac{P(B|A)P(A)}{P(B)}$:

$$c_{\text{map}} = \underset{s \in \mathbb{C}}{\arg\max}\ \frac{P(d|s)P(s)}{P(d)}$$

Drop denominator since $P(d)$ is the same for all classes:

$$c_{\text{map}} = \underset{s \in \mathbb{C}}{\arg\max}\ P(d|s)P(s)$$

Too many parameters / sparseness

$$
\begin{aligned}
c_{\text{map}} &= \underset{s \in \mathbb{C}}{\arg\max}\; P(d|s)P(s) \\
&= \underset{s \in \mathbb{C}}{\arg\max}\, P(\langle t_1, \ldots, t_k, \ldots, t_{n_d} \rangle | s)P(s)
\end{aligned}
$$

Too many parameters / sparseness

$$
\begin{aligned}
c_{\text{map}} &= \underset{s \in \mathbb{C}}{\arg\max}\ P(d|s)P(s) \\
&= \underset{s \in \mathbb{C}}{\arg\max}\, P(\langle t_1, \ldots, t_k, \ldots, t_{n_d} \rangle | s)P(s)
\end{aligned}
$$

- There are too many parameters $P(\langle t_1, \ldots, t_k, \ldots, t_{n_d} \rangle | s)$, one for each unique combination of a class and a sequence of words.

Too many parameters / sparseness

$$\begin{aligned} c_{\text{map}} &= \underset{s \in \mathbb{C}}{\arg\max}\ P(d|s)P(s) \\ &= \underset{s \in \mathbb{C}}{\arg\max}\, P(\langle t_1, \ldots, t_k, \ldots, t_{n_d} \rangle | s)P(s) \end{aligned}$$

- There are too many parameters $P(\langle t_1, \ldots, t_k, \ldots, t_{n_d} \rangle | s)$, one for each unique combination of a class and a sequence of words.
- We would need a very, very large number of training examples to estimate that many parameters.

Too many parameters / sparseness

$$
\begin{aligned}
c_{\text{map}} &= \arg\max_{s \in \mathbb{C}} \; P(d|s)P(s) \\
&= \arg\max_{s \in \mathbb{C}} P(\langle t_1, \ldots, t_k, \ldots, t_{n_d} \rangle | s) P(s)
\end{aligned}
$$

- There are too many parameters $P(\langle t_1, \ldots, t_k, \ldots, t_{n_d} \rangle | s)$, one for each unique combination of a class and a sequence of words.
- We would need a very, very large number of training examples to estimate that many parameters.
- This is the problem of data sparseness.

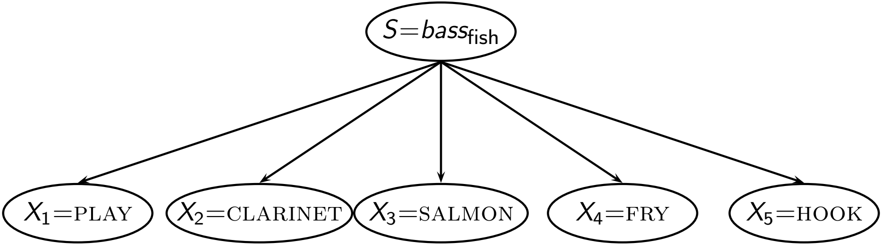
Naive Bayes conditional independence assumption

To reduce the number of parameters to a manageable size, we make the Naive Bayes conditional independence assumption:

$$P(d|s) = P(\langle t_1, \ldots, t_{n_d} \rangle | s) = \prod_{1 \leq k \leq n_d} P(X_k = t_k | s)$$

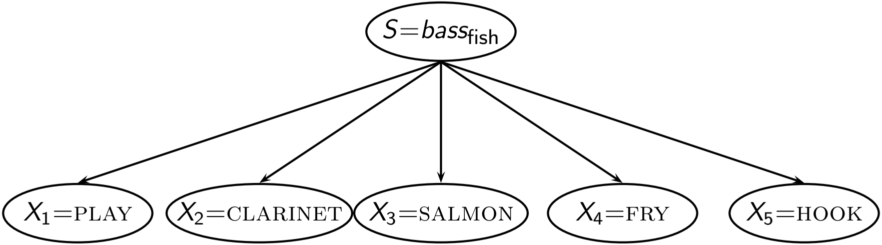
We assume that the probability of observing the conjunction of attributes is equal to the product of the individual probabilities $P(X_k = t_k|s)$. Recall from earlier the estimates for these priors and conditional probabilities: $\hat{P}(s) = \frac{N_c}{N}$ and $\hat{P}(t|c) = \frac{T_{ct}+1}{(\sum_{t' \in V} T_{ct'})+B}$

Generative Model



$P(s|d) \propto P(s) \prod_{1 \leq k \leq |n_d|} P(t_k|c)$

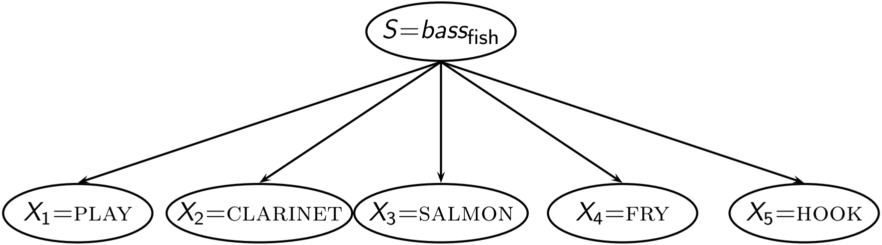
Generative Model



$P(s|d) \propto P(s) \prod_{1 \leq k \leq |n_d|} P(t_k|c)$

- Generate a sense with probability $P(s)$

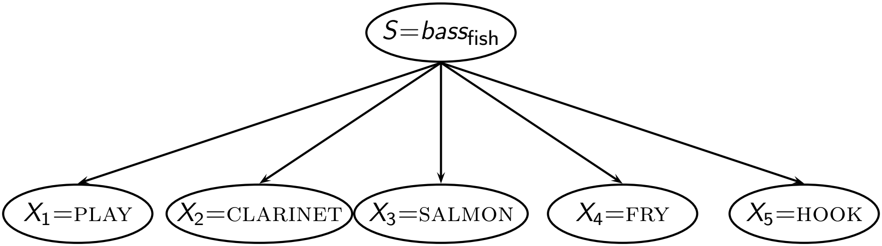
Generative Model



$P(s|d) \propto P(s) \prod_{1 \leq k \leq |n_d|} P(t_k|c)$

- Generate a sense with probability $P(s)$
- Generate each of the context words, conditional on the sense, but independent of each other, with probability $P(t_k|s)$

Generative Model



$P(s|d) \propto P(s) \prod_{1 \leq k \leq |n_d|} P(t_k|c)$

- Generate a sense with probability $P(s)$
- Generate each of the context words, conditional on the sense, but independent of each other, with probability $P(t_k|s)$
- To classify docs, we "simulate" the generative process and find the sense that is most likely to have generated the doc.

Second independence assumption

Second independence assumption

- $\hat{P}(t_{k_1}|c) = \hat{P}(t_{k_2}|c)$

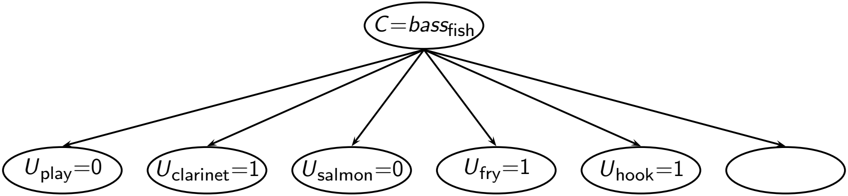
Second independence assumption

- $\hat{P}(t_{k_1}|c) = \hat{P}(t_{k_2}|c)$
- For example, for a token of *bass* using the sense $bass_{\text{fish}}$, the probability of generating FRY in the first position in the document is the same as generating it in the last position.

Second independence assumption

- $\hat{P}(t_{k_1}|c) = \hat{P}(t_{k_2}|c)$
- For example, for a token of *bass* using the sense $bass_{\text{fish}}$, the probability of generating FRY in the first position in the document is the same as generating it in the last position.
- The two independence assumptions amount to the bag of words model (information retrieval: order of words in documents does not matter).

A different Naive Bayes model: Bernoulli model



multinomial $\quad c_{map} = \arg\max_{s \in \mathbb{C}} \left[\log \hat{P}(s) + \sum_{1 \leq k \leq |n_d|} \log \hat{P}(t_k|s)\right]$

bernoulli $\quad c_{map} = \arg\max_{s \in \mathbb{C}} \left[\log \hat{P}(s) + \sum_{1 \leq k \leq |V_w|} \log \hat{P}(t_k|s)\right]$

Violation of Naive Bayes independence assumptions

Violation of Naive Bayes independence assumptions

- The independence assumptions do not really hold of documents written in natural language.

Violation of Naive Bayes independence assumptions

- The independence assumptions do not really hold of documents written in natural language.
- Conditional independence:

$$P(\langle t_1, \ldots, t_{n_d} \rangle | s) = \prod_{1 \le k \le n_d} P(X_k = t_k | s)$$

Violation of Naive Bayes independence assumptions

- The independence assumptions do not really hold of documents written in natural language.
- Conditional independence:

$$P(\langle t_1, \ldots, t_{n_d} \rangle | s) = \prod_{1 \leq k \leq n_d} P(X_k = t_k | s)$$

- Positional independence: $\hat{P}(t_{k_1}|c) = \hat{P}(t_{k_2}|c)$

Violation of Naive Bayes independence assumptions

- The independence assumptions do not really hold of documents written in natural language.
- Conditional independence:

$$P(\langle t_1, \ldots, t_{n_d} \rangle | s) = \prod_{1 \leq k \leq n_d} P(X_k = t_k | s)$$

- Positional independence: $\hat{P}(t_{k_1}|c) = \hat{P}(t_{k_2}|c)$
- Exercise

Violation of Naive Bayes independence assumptions

- The independence assumptions do not really hold of documents written in natural language.
- Conditional independence:

$$P(\langle t_1, \ldots, t_{n_d} \rangle | s) = \prod_{1 \leq k \leq n_d} P(X_k = t_k | s)$$

- Positional independence: $\hat{P}(t_{k_1}|c) = \hat{P}(t_{k_2}|c)$
- Exercise
  - Examples for why conditional independence assumption is not really true?

Violation of Naive Bayes independence assumptions

- The independence assumptions do not really hold of documents written in natural language.
- Conditional independence:

$$P(\langle t_1, \dots, t_{n_d} \rangle | s) = \prod_{1 \leq k \leq n_d} P(X_k = t_k | s)$$

- Positional independence: $\hat{P}(t_{k_1}|c) = \hat{P}(t_{k_2}|c)$
- Exercise
  - Examples for why conditional independence assumption is not really true?
  - Examples for why positional independence assumption is not really true?

Violation of Naive Bayes independence assumptions

- The independence assumptions do not really hold of documents written in natural language.
- Conditional independence:

$$P(\langle t_1, \ldots, t_{n_d} \rangle | s) = \prod_{1 \le k \le n_d} P(X_k = t_k | s)$$

- Positional independence: $\hat{P}(t_{k_1}|c) = \hat{P}(t_{k_2}|c)$
- Exercise
  - Examples for why conditional independence assumption is not really true?
  - Examples for why positional independence assumption is not really true?
- How can Naive Bayes work if it makes such inappropriate assumptions?

Why does Naive Bayes work?

- Naive Bayes can work well even though conditional independence assumptions are badly violated.
- Example:

	s_1	s_2	sense selected
true probability $P(s\|d)$	0.6	0.4	s_1
$\hat{P}(s)\prod_{1\leq k\leq n_d}\hat{P}(t_k\|s)$	0.00099	0.00001	
NB estimate $\hat{P}(s\|d)$	0.99	0.01	s_1

Illustration

- Suppose the featues are NOT independent, causing, say, underestimation $\mathrm{P}(s_2 \mid d) = 0.01$).

  $$\mathrm{P}(s_2|t_k, t_j) \quad > \quad \hat{P}(s_1|t_k)\hat{P}(s_2|t_j)$$

  Then $\mathrm{P}(s_1|t_k, t_j)$ will be overestimated (0.99):

  $$\mathrm{P}(s_1|t_k, t_j) \quad < \quad \hat{P}(s_1|t_k)\hat{P}(s_1|t_j)$$

  As long as NB overestimates the larger prob, it will still make correct classification decisions. Even if NB overestimates the smaller prob, the decision *might* still be right.
- Classification is about predicting the correct class and not about accurately estimating probabilities.
- Correct estimation $\Rightarrow$ accurate prediction.
- But not vice versa!

Naive Bayes is not so naive

- Naive Bayes has won some bakeoffs (e.g., KDD-CUP 97)
- More robust to nonrelevant features than some more complex learning methods
- More robust to concept drift (changing of definition of class over time) than some more complex learning methods
- Better than methods like decision trees when we have many equally important features
- A good dependable baseline for text classification (but not the best)
- Optimal if independence assumptions hold (never true for text, but true for some domains)
- Very fast
- Low storage requirements