Naïve Bayes

CMSC 473/673
UMBC

Some slides adapted from 3SLP
Outline

Terminology: bag-of-words

“Naïve” assumption

Training & performance

NB as a language model
Outline

Terminology: bag-of-words

“Naïve” assumption

Training & performance

NB as a language model
Bag-of-words

Based on *some* tokenization, turn an input document into an array (or dictionary or set) of its unique vocab items.

Position/ordering in the document is *not* important.

Bag-of-words go hand-in-hand with one-hot representations, but they *can* be extended to handle dense representations.

Short-hand: **BOW**
I love this movie! It's sweet, but with satirical humor. The dialogue is great and the adventure scenes are fun... It manages to be whimsical and romantic while laughing at the conventions of the fairy tale genre. I would recommend it to just about anyone. I've seen it several times, and I'm always happy to see it again whenever I have a friend who hasn't seen it yet!
I love this movie! It's sweet, but with satirical humor. The dialogue is great and the adventure scenes are fun... It manages to be whimsical and romantic while laughing at the conventions of the fairy tale genre. I would recommend it to just about anyone. I've seen it several times, and I'm always happy to see it again whenever I have a friend who hasn't seen it yet!
I love this movie! It's sweet, but with satirical humor. The dialogue is great and the adventure scenes are fun... It manages to be whimsical and romantic while laughing at the conventions of the fairy tale genre. I would recommend it to just about anyone. I've seen it several times, and I'm always happy to see it again whenever I have a friend who hasn't seen it yet!
The Bag of Words Representation

I love this movie! It's sweet, but with satirical humor. The dialogue is great and the adventure scenes are fun... It manages to be whimsical and romantic while laughing at the conventions of the fairy tale genre. I would recommend it to just about anyone. I've seen it several times, and I'm always happy to see it again whenever I have a friend who hasn't seen it yet!
Bag-of-words as a Function

Based on some tokenization, turn an input document into an array (or dictionary or set) of its unique vocab items

Think of getting a BOW rep. as a function $f$

input: Document

output: Container of size $E$, indexable by each vocab type $v$
## Some Bag-of-words Functions

<table>
<thead>
<tr>
<th>Kind</th>
<th>Type of $f_v$</th>
<th>Interpretation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Binary</td>
<td>0, 1</td>
<td>Did $v$ appear in the document?</td>
</tr>
<tr>
<td>Count-based</td>
<td>Natural number (int $\geq 0$)</td>
<td>How often did $v$ occur in the document?</td>
</tr>
<tr>
<td>Averaged</td>
<td>Real number ($\geq 0$, $\leq 1$)</td>
<td>How often did $v$ occur in the document, normalized by doc length?</td>
</tr>
<tr>
<td>TF-IDF (term frequency, inverse document frequency)</td>
<td>Real number ($\geq 0$)</td>
<td>How frequent is a word, tempered by how prevalent it is across the corpus (to be covered later!)</td>
</tr>
</tbody>
</table>

...
## Some Bag-of-words Functions

<table>
<thead>
<tr>
<th>Kind</th>
<th>Type of $f_v$</th>
<th>Interpretation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Binary</td>
<td>0, 1</td>
<td>Did $v$ appear in the document?</td>
</tr>
<tr>
<td>Count-based</td>
<td>Natural number (int &gt;= 0)</td>
<td>How often did $v$ occur in the document?</td>
</tr>
<tr>
<td>Averaged</td>
<td>Real number ($\geq 0$, $\leq 1$)</td>
<td>How often did $v$ occur in the document, normalized by doc length?</td>
</tr>
<tr>
<td>TF-IDF (term frequency, inverse document frequency)</td>
<td>Real number ($\geq 0$)</td>
<td>How frequent is a word, tempered by how prevalent it is across the corpus (to be covered later!)</td>
</tr>
</tbody>
</table>

Q: Is this a reasonable representation?

Q: What are some tradeoffs (benefits vs. costs)?
Outline

Terminology: bag-of-words

“Naïve” assumption

Training & performance

NB as a language model
### Bag of Words Classifier

```
<table>
<thead>
<tr>
<th>Word</th>
<th>Count</th>
</tr>
</thead>
<tbody>
<tr>
<td>seen</td>
<td>2</td>
</tr>
<tr>
<td>sweet</td>
<td>1</td>
</tr>
<tr>
<td>whimsical</td>
<td>1</td>
</tr>
<tr>
<td>recommend</td>
<td>1</td>
</tr>
<tr>
<td>happy</td>
<td>1</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>
```

\[ \gamma( ) = c \]
Naïve Bayes (NB) Classifier

\[ \text{argmax}_Y p(X \mid Y) \times p(Y) \]

Q: Are we doing discriminative training or generative training?
Naïve Bayes (NB) Classifier

Start with Bayes Rule

$$\arg\max_Y p(X \mid Y) \ast p(Y)$$

Start with Bayes Rule

**Q:** Are we doing discriminative training or generative training?

**A:** generative training
Naïve Bayes (NB) Classifier

$$\arg\max_Y \prod_{t} p(X_t|Y) \times p(Y)$$

Label each word

Iterate through types

Adopt naïve bag of words representation $X_t$
Naïve Bayes (NB) Classifier

\[
\text{argmax}_Y \prod_{t} p(X_t | Y) \ast p(Y)
\]

Adopt naïve bag of words representation \(X_t\)

Assume position doesn’t matter
Outline

Terminology: bag-of-words

“Naïve” assumption

Training & performance

NB as a language model
Learning for a Naïve Bayes Classifier

Assuming V vocab types $w_1, \ldots, w_V$ and L classes $u_1, \ldots, u_L$ (and appropriate corpora)
Learning for a Naïve Bayes Classifier

Assuming $V$ vocab types $w_1, \ldots, w_V$ and $L$ classes $u_1, \ldots, u_L$ (and appropriate corpora)

Q: What parameters (values/weights) must be learned?
Learning for a Naïve Bayes Classifier

Assuming V vocab types $w_1, \ldots, w_V$ and L classes $u_1, \ldots, u_L$ (and appropriate corpora)

Q: What parameters (values/weights) must be learned?

A: $p(w_v | u_i), p(u_i)$
Learning for a Naïve Bayes Classifier

Assuming V vocab types $w_1, \ldots, w_V$ and L classes $u_1, \ldots, u_L$ (and appropriate corpora)

Q: What parameters (values/weights) must be learned?
A: $p(w_v | u_l), p(u_l)$

Q: How many parameters must be learned?
Learning for a Naïve Bayes Classifier

Assuming V vocab types $w_1, \ldots, w_V$ and L classes $u_1, \ldots, u_L$ (and appropriate corpora)

Q: What parameters (values/weights) must be learned?
A: $p(w_v | u_l), p(u_l)$

Q: How many parameters must be learned?
A: $LK + L$
Learning for a Naïve Bayes Classifier

Assuming V vocab types $w_1, \ldots, w_V$ and L classes $u_1, \ldots, u_L$ (and appropriate corpora)

Q: What parameters (values/weights) must be learned?  
A: $p(w_v|u_l), p(u_l)$

Q: How many parameters must be learned?  
A: $LK + L$

Q: What distributions need to sum to 1?
Learning for a Naïve Bayes Classifier

Assuming $V$ vocab types $w_1, \ldots, w_V$ and $L$ classes $u_1, \ldots, u_L$ (and appropriate corpora)

Q: What parameters (values/weights) must be learned?
A: $p(w_v|u_l), p(u_l)$

Q: How many parameters must be learned?
A: $LK + L$

Q: What distributions need to sum to 1?
A: Each $p(\cdot | u_l)$, and the prior
Learning for a Naïve Bayes Classifier

Assuming V vocab types $w_1, \ldots, w_V$ and L classes $u_1, \ldots, u_L$ (and appropriate corpora)

If you’re going to compute perplexity from $p(\cdot | u_l)$, all class specific language models $p(\cdot | u_l)$ MUST share a common vocab (otherwise it’s not a fair comparison!!!)
Learning for a Naïve Bayes Classifier

Assuming V vocab types $w_1, \ldots, w_V$ and L classes $u_1, \ldots, u_L$ (and appropriate corpora)

If you’re going to compute perplexity from $p(\cdot | u_l)$, all class specific language models $p(\cdot | u_l)$ MUST share a common vocab (otherwise it’s not a fair comparison!!!)

Q: Should OOV and UNK be included?
Learning for a Naïve Bayes Classifier

Assuming $V$ vocab types $w_1, \ldots, w_V$ and $L$ classes $u_1, \ldots, u_L$ (and appropriate corpora)

If you’re going to compute perplexity from $p(\cdot | u_l)$, all class specific language models $p(\cdot | u_l)$ MUST share a common vocab (otherwise it’s not a fair comparison!!!)

Q: Should OOV and UNK be included?

Q: Should EOS be included?
Learning for a Naïve Bayes Classifier

Assuming $V$ vocab types $w_1, \ldots, w_V$ and $L$ classes $u_1, \ldots, u_L$ (and appropriate corpora)

If you’re going to compute perplexity from $p(\cdot | u_l)$, all class specific language models $p(\cdot | u_l)$ **MUST** share a common vocab (otherwise it’s not a fair comparison!!!)

A binary/count-based NB classifier is also called a **Multinomial Naïve Bayes** classifier
Multinomial Naïve Bayes: Learning

From training corpus, extract *Vocabulary*
Multinomial Naïve Bayes: Learning

From training corpus, extract *Vocabulary*

**Calculate $P(c_j)$ terms**

For each $c_j$ in $C$ do

$docs_j = \text{all docs with class } = c_j$

$$p(c_j) = \frac{|docs_j|}{\# \text{ docs}}$$
Multinomial Naïve Bayes: Learning

From training corpus, extract *Vocabulary*

**Calculate $P(c_j)$ terms**

For each $c_j$ in $C$ do

$docs_j = \text{all docs with class }=c_j$

$$p(c_j) = \frac{|docs_j|}{\# \text{ docs}}$$

**Calculate $P(w_k \mid c_j)$ terms**

$Text_j = \text{single doc containing all docs}_j$

For each word $w_k$ in *Vocabulary*

$$n_k = \# \text{ of occurrences of } w_k \text{ in } Text_j$$

$$p(w_k \mid c_j) = \text{class (unigram) LM}$$
Brill and Banko (2001)

With enough data, the classifier may not matter
Outline

Terminology: bag-of-words

“Naïve” assumption

Training & performance

NB as a language model
# Naïve Bayes as a Language Model

<table>
<thead>
<tr>
<th>Positive Model</th>
<th>Negative Model</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.1</td>
<td>0.2</td>
</tr>
<tr>
<td>love</td>
<td>love</td>
</tr>
<tr>
<td>this</td>
<td>this</td>
</tr>
<tr>
<td>fun</td>
<td>fun</td>
</tr>
<tr>
<td>film</td>
<td>film</td>
</tr>
<tr>
<td>0.1</td>
<td>0.1</td>
</tr>
</tbody>
</table>
Naïve Bayes as a Language Model

Which class assigns the higher probability to \( s \)?

<table>
<thead>
<tr>
<th>Positive Model</th>
<th>Negative Model</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.1 l</td>
<td>0.2 l</td>
</tr>
<tr>
<td>0.1 love</td>
<td>0.001 love</td>
</tr>
<tr>
<td>0.01 this</td>
<td>0.01 this</td>
</tr>
<tr>
<td>0.05 fun</td>
<td>0.005 fun</td>
</tr>
<tr>
<td>0.1 film</td>
<td>0.1 film</td>
</tr>
</tbody>
</table>

Sect. 3.2.1
Naïve Bayes as a Language Model

Which class assigns the higher probability to $s$?

<table>
<thead>
<tr>
<th>Positive Model</th>
<th>Negative Model</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.1 I</td>
<td>0.2 I</td>
</tr>
<tr>
<td>0.1 love</td>
<td>0.001 love</td>
</tr>
<tr>
<td>0.01 this</td>
<td>0.01 this</td>
</tr>
<tr>
<td>0.05 fun</td>
<td>0.005 fun</td>
</tr>
<tr>
<td>0.1 film</td>
<td>0.1 film</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>I</th>
<th>love</th>
<th>this</th>
<th>fun</th>
<th>film</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.1</td>
<td>0.1</td>
<td>0.01</td>
<td>0.05</td>
<td>0.1</td>
</tr>
<tr>
<td>0.2</td>
<td>0.001</td>
<td>0.01</td>
<td>0.005</td>
<td>0.1</td>
</tr>
</tbody>
</table>
Naïve Bayes as a Language Model

Which class assigns the higher probability to $s$?

<table>
<thead>
<tr>
<th>Positive Model</th>
<th>Negative Model</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.1 l</td>
<td>0.2 l</td>
</tr>
<tr>
<td>0.1 love</td>
<td>0.001 love</td>
</tr>
<tr>
<td>0.01 this</td>
<td>0.01 this</td>
</tr>
<tr>
<td>0.05 fun</td>
<td>0.005 fun</td>
</tr>
<tr>
<td>0.1 film</td>
<td>0.1 film</td>
</tr>
</tbody>
</table>

$5 \times 10^{-7} \approx P(s|\text{pos}) > P(s|\text{neg}) \approx 1 \times 10^{-9}$
Naïve Bayes To Try

http://csee.umbc.edu/courses/undergraduate/473/f19/nb

- Toy problem: classify whether a tweet will be retweeted
- Toy problem: OOV and EOS are not included
- Laplace smoothing is used for $p(\text{word} \mid \text{label})$
Summary: Naïve Bayes is Not So Naïve

Very Fast, low storage requirements

Robust to Irrelevant Features

Very good in domains with many equally important features

Optimal if the independence assumptions hold

Dependable baseline for text classification (but often not the best)
But: Naïve Bayes Isn’t Without Issue

Model the posterior in one go?

Are the features really uncorrelated?

Are plain counts always appropriate?

Are there “better” ways of handling missing/noisy data? (automated, more principled)
Outline

Terminology: bag-of-words

“Naïve” assumption

Training & performance

NB as a language model