Assignment 1 will be out later today — due Monday, Feb 22, 1:30pm (in 2 weeks)
Assignment 1 will be out later today — due Monday, Feb 22, 1:30pm (in 2 weeks)
Why classify?

Spam detection

Sentiment analysis
Why classify?

- Authorship attribution
- Language detection
- News categorization
- ...
Classification: The Task

Movie was terrible -> Classify -> Negative

Amazing acting -> Classify -> Positive
Classification: The Task

- Inputs:
  - Movie was terrible
  - Amazing acting
Classification: The Task

• Inputs:
  • A document $d$

Movie was terrible -> Classify -> Negative
Amazing acting -> Classify -> Positive
Classification: The Task

• Inputs:
  • A document $d$
  • A set of classes $C = \{c_1, c_2, c_3, \ldots, c_m\}$
Classification: The Task

- Inputs:
  - A document $d$
  - A set of classes $C = \{c_1, c_2, c_3, \ldots, c_m\}$

- Output:
Classification: The Task

- Inputs:
  - A document \( d \)
  - A set of classes \( C = \{ c_1, c_2, c_3, \ldots, c_m \} \)

- Output:
  - Predicted class \( c \) for document \( d \)
Rule-based classification
Rule-based classification

- Combinations of features on words in document, meta-data

**IF** there exists word $w$ in document $d$ such that $w$ in \{good, great, extra-ordinary, \ldots\},
**THEN** output Positive

**IF** email address ends in \{ithelpdesk.com, makemoney.com, spinthewheel.com, \ldots\}
**THEN** output SPAM
Rule-based classification

- Combinations of features on words in document, meta-data

  IF there exists word w in document d such that w in [good, great, extra-ordinary, …],
  THEN output Positive

  IF email address ends in [ithelpdesk.com, makemoney.com, spinthewheel.com, …]
  THEN output SPAM

- Can be very accurate
Rule-based classification

- Combinations of features on words in document, meta-data

  IF there exists word w in document d such that w in [good, great, extra-ordinary, …],
  THEN output Positive

  IF email address ends in [ithelpdesk.com, makemoney.com, spinthewheel.com, …]
  THEN output SPAM

- Can be very accurate

- Rules may be hard to define (and some even unknown to us!)
Rule-based classification

- Combinations of features on words in document, meta-data

  IF there exists word w in document d such that w in [good, great, extra-ordinary, ...],
  THEN output Positive

  IF email address ends in [ithelpdesk.com, makemoney.com, spinthewheel.com, ...]
  THEN output SPAM

- Can be very accurate

- Rules may be hard to define (and some even unknown to us!)

- Expensive
Rule-based classification

- Combinations of features on words in document, meta-data

  IF there exists word w in document d such that w in [good, great, extra-ordinary, …],
  THEN output Positive

  IF email address ends in [ithelpdesk.com, makemoney.com, spinthewheel.com, …]
  THEN output SPAM

- Can be very accurate
- Rules may be hard to define (and some even unknown to us!)
- Expensive
- Not easily generalizable
Rule-based classification

- Combinations of features on words in document, meta-data

  IF there exists word \( w \) in document \( d \) such that \( w \) in \[good, great, extra-ordinary, \ldots\],
  THEN output Positive

  IF email address ends in \[i\text{hel}p\text{desk.com, m}a\text{kemoney.com, s}p\text{inthewheel.com, \ldots}\]
  THEN output SPAM

- Can be very accurate
- Rules may be hard to define (and some even unknown to us!)
- Expensive
- Not easily generalizable
Supervised Learning: Let’s use statistics!
Supervised Learning: Let’s use statistics!

- Data-driven approach
Supervised Learning: Let’s use statistics!

• Data-driven approach

• Let the machine figure out the best patterns to use
Supervised Learning: Let’s use statistics!

- Data-driven approach
- Let the machine figure out the best patterns to use
- Inputs:
Supervised Learning: Let’s use statistics!

• Data-driven approach

• Let the machine figure out the best patterns to use

• Inputs:
  
  • Set of $m$ classes $C = \{c_1, c_2, \ldots, c_m\}$
Supervised Learning: Let’s use statistics!

• Data-driven approach

• Let the machine figure out the best patterns to use

• Inputs:

  • Set of \( m \) classes \( C = \{c_1, c_2, \ldots, c_m\} \)

  • Set of \( n \) ‘labeled’ documents: \( \{(d_1, c_1), (d_2, c_2), \ldots, (d_n, c_n)\} \)
Supervised Learning: Let’s use statistics!

• Data-driven approach

• Let the machine figure out the best patterns to use

• Inputs:
  
  • Set of $m$ classes $C = \{c_1, c_2, \ldots, c_m\}$
  
  • Set of $n$ ‘labeled’ documents:  $\{(d_1, c_1), (d_2, c_2), \ldots, (d_n, c_n)\}$

• Output:
Supervised Learning: Let’s use statistics!

- Data-driven approach
- Let the machine figure out the best patterns to use

**Inputs:**
- Set of $m$ classes $C = \{c_1, c_2, \ldots, c_m\}$
- Set of $n$ ‘labeled’ documents: $\{(d_1, c_1), (d_2, c_2), \ldots, (d_n, c_n)\}$

**Output:**
- Trained classifier, $F : d \rightarrow c$
Supervised Learning: Let’s use statistics!

- Data-driven approach
- Let the machine figure out the best patterns to use
- Inputs:
  - Set of $m$ classes $C = \{c_1, c_2, \ldots, c_m\}$
  - Set of $n$ ‘labeled’ documents: $\{(d_1, c_1), (d_2, c_2), \ldots, (d_n, c_n)\}$
- Output:
  - Trained classifier, $F : d \rightarrow c$

Key questions:

a) What is the form of $F$?

b) How do we learn $F$?
Types of supervised classifiers

- Naive Bayes
- Logistic regression
- Support vector machines
- k-nearest neighbors
Multinomial Naive Bayes
Multinomial Naive Bayes

- Simple classification model making use of Bayes rule
Multinomial Naive Bayes

- Simple classification model making use of Bayes rule
- Bayes Rule:
Multinomial Naive Bayes

• Simple classification model making use of Bayes rule

• Bayes Rule:

\[
P(c \mid d) = \frac{P(c) \ P(d \mid c)}{P(d)}
\]

\begin{align*}
\text{d - document} \\
\text{c - class}
\end{align*}
Multinomial Naive Bayes

• Simple classification model making use of Bayes rule

• Bayes Rule:

\[
P(c \mid d) = \frac{P(c) \ P(d \mid c)}{P(d)}
\]

- \(d\) – document
- \(c\) – class
Multinomial Naive Bayes

- Simple classification model making use of Bayes rule

- Bayes Rule:

\[
P(c \mid d) = \frac{P(c) P(d \mid c)}{P(d)}
\]
Multinomial Naive Bayes

• Simple classification model making use of Bayes rule

• Bayes Rule:

\[ P(c \mid d) = \frac{P(c) \cdot P(d \mid c)}{P(d)} \]

• Makes strong (‘naive’) independence assumptions
Predicting a class

- Best class, \( c_{MAP} = \arg\max_{c \in C} p(c | d) \)
Predicting a class

- Best class, \( \mathcal{C}_{MAP} = \arg\max_{c \in \mathcal{C}} p(c|d) \)
  
  \[
  = \arg\max_c \frac{p(c) p(d|c)}{p(d)}
  \]
Predicting a class

• Best class, $c_{\text{MAP}} = \arg\max_{c \in C} p(c | d)$

  $= \arg\max_c \frac{p(c) p(d | c)}{p(d)}$

  $= \arg\max_c p(c) p(d | c)$
Predicting a class

- Best class, \( C_{MAP} = \arg\max_{c \in C} p(c | d) \)
  \[
  = \arg\max_{c} \frac{p(c) \cdot p(d | c)}{p(d)}
  = \arg\max_{c} p(c) \cdot p(d | c)
  \]

\( p(c) \rightarrow \text{prior probability of class } c \)
Predicting a class

• Best class,

\[ c_{MAP} = \arg\max_{c \in C} p(c | d) \]

\[ = \arg\max_c \frac{p(c) p(d | c)}{p(d)} \]

\[ = \arg\max_c p(c) p(d | c) \]

\[ p(c) \rightarrow \text{Prior probability of class } c \]

\[ p(d | c) \rightarrow \text{Conditional probability of generating document } d \text{ from class } c \]
Predicting a class

\[ C_{\text{MAP}} = \arg \max_{c \in C} p(c \mid d) \]

\[ = \arg \max_{c} \frac{p(c) p(d \mid c)}{p(d)} \]

\[ = \arg \max_{c} p(c) p(d \mid c) \]

- \( p(c) \rightarrow \) Prior probability of class \( c \)
- \( p(d \mid c) \rightarrow \) Conditional probability of generating document \( d \) from class \( c \).
How to represent $P(d \mid c)$?
How to represent $P(d \mid c)$?

- **Option 1**: represent the entire sequence of words
How to represent $P(d \mid c)$?

- **Option 1**: represent the entire sequence of words
  
  - $P(w_1, w_2, \ldots, w_K \mid c)$  
    
    *(too many sequences!)*
How to represent $P(d \mid c)$?

• **Option 1**: represent the entire sequence of words

  \[ P(w_1, w_2, \ldots, w_K \mid c) \]  
  *(too many sequences!)*

• **Option 2**: Bag of words
How to represent $P(d \mid c)$?

- **Option 1**: represent the entire sequence of words
  
  - $P(w_1, w_2, \ldots, w_K \mid c)$
    
    (too many sequences!)

- **Option 2**: Bag of words
How to represent $P(d \mid c)$?

- **Option 1**: represent the entire sequence of words
  
  $$P(w_1, w_2, \ldots, w_K \mid c)$$  
  (too many sequences!)

- **Option 2**: Bag of words

  Assume position of each word is irrelevant  
  (both absolute and relative)
How to represent $P(d \mid c)$?

- **Option 1**: represent the entire sequence of words
  \[ P(w_1, w_2, \ldots, w_K \mid c) \]  
  (too many sequences!)

- **Option 2**: Bag of words
  
  Assume position of each word is irrelevant  
  (both absolute and relative)
  
  \[ P(w_1, w_2, \ldots, w_K \mid c) = P(w_1 \mid c)P(w_2 \mid c) \ldots P(w_K \mid c) \]
How to represent $P(d \mid c)$?

- **Option 1**: represent the entire sequence of words

  $$P(w_1, w_2, \ldots, w_K \mid c)$$  \textit{(too many sequences!)}

- **Option 2**: Bag of words

  - Assume position of each word is irrelevant  
    (both absolute and relative)

  $$P(w_1, w_2, \ldots, w_K \mid c) = P(w_1 \mid c)P(w_2 \mid c) \ldots P(w_k \mid c)$$

  - Probability of each word is \textit{conditionally independent}  
    of the other words given class $c$
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!
Predicting with Naive Bayes

• We now have:

$$C_{MAP} = \arg\max_c P(d \mid c) P(c)$$
Predicting with Naive Bayes

- We now have:

\[ c_{\text{MAP}} = \arg\max_c P(d \mid c) P(c) \]

\[ = \arg\max_c P(w_1, w_2, \ldots, w_k \mid c) P(c) \]
Predicting with Naive Bayes

- We now have:

$$C_{MAP} = \arg \max_c P(d \mid c) P(c)$$

$$= \arg \max_c P(w_1, w_2, \ldots, w_k \mid c) P(c)$$

$$= \arg \max_c P(c) \prod_{i=1}^{k} P(w_i \mid c)$$

(using BOW assumption)
Naive Bayes as a generative model

\[ P(c) \]

\[ d_1 \]

\[ \ldots \]

\[ \ldots \]
Naive Bayes as a generative model
Naive Bayes as a generative model
Naive Bayes as a generative model

\[ c = \text{Science} \]
\[ w_1 = \text{Scientists} \]
\[ w_2 = \text{have} \]
\[ w_3 = \text{discovered} \]

\[ c = \text{Environment} \]
\[ w_1 = \text{Global} \]
\[ w_2 = \text{warming} \]
\[ w_3 = \text{has} \]

Generate the entire data set one document at a time
Estimating probabilities

Maximum likelihood estimates:

\[
\hat{P}(c_j) = \frac{\text{count}(\text{class} = c_j)}{\sum_c \text{count}(\text{class} = c)}
\]
Estimating probabilities

Maximum likelihood estimates:

\[
\hat{P}(c_j) = \frac{\text{count (class = } c_j)}{\sum_{c} \text{count (class = } c)}
\]

\[
\hat{P}(w_i | c_j) = \frac{\text{count (} w_i, c_j \text{)}}{\sum_{w} \text{count (} w, c_j \text{)}}
\]

\[
\arg\max_{c} P(c) \prod_{i=1}^{k} P(w_i | c)
\]
Data sparsity
Data sparsity

- What if \( \text{count('amazing', positive)} = 0? \)
Data sparsity

• What if count('amazing', positive) = 0?

⇒ Implies P('amazing' | positive) = 0
Data sparsity

• What if $\text{count('amazing', positive)} = 0$?

  $\Rightarrow$ Implies $P('amazing' \mid \text{positive}) = 0$

• Given a review document, $d = "\ldots \text{most amazing movie ever} \ldots"$
Data sparsity

• What if \( \text{count('amazing', positive)} = 0? \)

  \( \Rightarrow \) Implies \( \text{P('amazing' | positive)} = 0 \)

• Given a review document, \( d = "\ldots \text{most amazing movie ever} \ldots" \)

\[
\begin{align*}
C_{\text{MAP}} &= \arg\max_C \hat{p}(c) \prod_{i=1}^{K} p(\omega_i | c) \\
&= \arg\max_C \hat{p}(c) \cdot 0 = 0
\end{align*}
\]
Data sparsity

- What if count(‘amazing’, positive) = 0?

  ➡ Implies P(‘amazing’ | positive) = 0

- Given a review document, d = “…. most amazing movie ever …”

\[
C_{MAP} = \arg\max_C \hat{p}(c) \frac{1}{K} \prod_{i=1}^{K} p(\omega_i | c)
\]

\[
= \arg\max_c \hat{p}(c) \cdot 0 = 0
\]

This sounds familiar…
Solution: Smoothing!
Solution: Smoothing!

- Laplace smoothing:
Solution: Smoothing!

• Laplace smoothing:

\[
\hat{P}(w_i | c) = \frac{\text{count}(w_i, c) + \alpha}{\left[ \sum_{w} \text{count}(w, c) \right] + \alpha |V|}
\]
Solution: Smoothing!

- Laplace smoothing:

\[
\hat{P}(w_i | c) = \frac{\text{count}(w_i, c) + \alpha}{\left[ \sum_w \text{count}(w, c) \right] + \alpha |V|}
\]

*Vocabulary size*
Solution: Smoothing!

- Laplace smoothing:

\[
\hat{p}(w_i | c) = \frac{\text{count}(w_i, c) + \alpha}{\left[ \sum_w \text{count}(w, c) \right] + \alpha |V|}
\]
Solution: Smoothing!

• Laplace smoothing:

\[
\hat{P}(w_i | c) = \frac{\text{count}(w_i, c) + \alpha}{\left[ \sum_w \text{count}(w, c) \right] + \alpha |V|}
\]

• Simple, easy to use
Solution: Smoothing!

• Laplace smoothing:

\[ \hat{P}(w_i | c) = \frac{\text{count}(w_i, c) + \alpha}{\sum_w \text{count}(w, c) + \alpha |V|} \]

• Simple, easy to use

• Effective in practice
Overall process
Overall process

**Input:** Set of annotated documents \( \{(d_i, c_i)\}_{i=1}^n \)
Overall process

**Input:** Set of annotated documents $\{(d_i, c_i)\}_{i=1}^n$

A. Compute vocabulary $V$ of all words
Overall process

Input: Set of annotated documents \( \{(d_i, c_i)\}_{i=1}^n \)

A. Compute vocabulary \( V \) of all words

B. Calculate \( \hat{P}(c_j) = \frac{\text{Count}(c_j)}{n} \)
Overall process

Input: Set of annotated documents \(\{(d_i, c_i)\}_{i=1}^{n}\)

A. Compute vocabulary \(V\) of all words

B. Calculate \(\hat{P}(c_j) = \frac{\text{Count}(c_j)}{n}\)

C. Calculate \(\hat{P}(w_i | c_j) = \frac{\text{Count}(w_i, c_j) + \alpha}{\sum_{w \in V} [\text{Count}(w, c_j) + \alpha]}\)
Overall process

**Input:** Set of annotated documents \( \{(d_i, c_i)\}_{i=1}^n \)

A. Compute vocabulary \( V \) of all words

B. Calculate \( \hat{P}(c_j) = \frac{\text{Count}(c_j)}{n} \)

C. Calculate \( \hat{P}(w_i | c_j) = \frac{\text{Count}(w_i, c_j) + \alpha}{\sum_{w \in V} [\text{Count}(w, c_j) + \alpha]} \)

D. (Prediction) Given document \( d = (w_1, w_2, \ldots, w_k) \)

\[ c_{MAP} = \arg \max_c \hat{P}(c) \prod_{i=1}^K \hat{P}(w_i | c) \]
Overall process

**Input:** Set of annotated documents \( \{(d_i, c_i)\}_{i=1}^n \)

A. Compute vocabulary \( V \) of all words

B. Calculate \( \hat{P}(c_j) = \frac{\text{Count}(c_j)}{n} \)

C. Calculate \( \hat{P}(w_i | c_j) = \frac{\text{Count}(w_i, c_j) + \alpha}{\sum_{w \in V} [\text{Count}(w, c_j) + \alpha]} \)

D. (Prediction) Given document \( d = (w_1, w_2, \ldots, w_k) \)

\[ c_{MAP} = \arg \max_c [\hat{P}(c)]^{\prod_{i=1}^K \hat{P}(w_i | c)} \]

prior
Naive Bayes Example

\[ \hat{P}(w \mid c) = \frac{\text{count}(w, c) + 1}{\text{count}(c) + |V|} \]

\[ \hat{P}(c) = \frac{N_c}{N} \]

<table>
<thead>
<tr>
<th>Doc</th>
<th>Words</th>
<th>Class</th>
</tr>
</thead>
<tbody>
<tr>
<td>Training</td>
<td>1 Chinese Beijing Chinese</td>
<td>c</td>
</tr>
<tr>
<td></td>
<td>2 Chinese Chinese Shanghai</td>
<td>c</td>
</tr>
<tr>
<td></td>
<td>3 Chinese Macao</td>
<td>c</td>
</tr>
<tr>
<td></td>
<td>4 Tokyo Japan Chinese</td>
<td>j</td>
</tr>
<tr>
<td>Test</td>
<td>5 Chinese Chinese Chinese Tokyo Japan</td>
<td>?</td>
</tr>
</tbody>
</table>

\[ \frac{3}{4} \times \frac{3}{7} \approx 0.0003 \]
# Naive Bayes Example

\[
\hat{P}(c) = \frac{N_c}{N}
\]

\[
\hat{P}(w | c) = \frac{\text{count}(w, c) + 1}{\text{count}(c) + |V|}
\]

**Priors:**

\[
P(c) = \frac{3}{4}
\]

\[
P(j) = \frac{1}{4}
\]

<table>
<thead>
<tr>
<th>Doc</th>
<th>Words</th>
<th>Class</th>
</tr>
</thead>
<tbody>
<tr>
<td>Training 1</td>
<td>Chinese Beijing Chinese</td>
<td>c</td>
</tr>
<tr>
<td>2</td>
<td>Chinese Chinese Shanghai</td>
<td>c</td>
</tr>
<tr>
<td>3</td>
<td>Chinese Macao</td>
<td>c</td>
</tr>
<tr>
<td>4</td>
<td>Tokyo Japan Chinese</td>
<td>j</td>
</tr>
<tr>
<td>Test 5</td>
<td>Chinese Chinese Chinese Tokyo Japan</td>
<td>?</td>
</tr>
</tbody>
</table>

\[
\approx 0.0001
\]

\[
\approx 0.0003
\]
Naive Bayes Example

\[
\hat{P}(c) = \frac{N_c}{N}
\]

\[
\hat{P}(w \mid c) = \frac{\text{count}(w, c) + 1}{\text{count}(c) + |V|}
\]

**Priors:**

\[
P(c) = \frac{3}{4}, \quad P(j) = \frac{1}{4}
\]

**Conditional Probabilities:**

\[
P(\text{Chinese} \mid c) = \frac{(5+1)}{(8+6)} = \frac{6}{14} = \frac{3}{7}
\]

\[
P(\text{Tokyo} \mid c) = \frac{(0+1)}{(8+6)} = \frac{1}{14}
\]

\[
P(\text{Japan} \mid c) = \frac{(0+1)}{(8+6)} = \frac{1}{14}
\]

\[
P(\text{Chinese} \mid j) = \frac{(1+1)}{(3+6)} = \frac{2}{9}
\]

\[
P(\text{Tokyo} \mid j) = \frac{(1+1)}{(3+6)} = \frac{2}{9}
\]

\[
P(\text{Japan} \mid j) = \frac{(1+1)}{(3+6)} = \frac{2}{9}
\]

<table>
<thead>
<tr>
<th>Doc</th>
<th>Words</th>
<th>Class</th>
</tr>
</thead>
<tbody>
<tr>
<td>Training 1</td>
<td>Chinese Beijing Chinese</td>
<td>c</td>
</tr>
<tr>
<td>2</td>
<td>Chinese Chinese Shanghai</td>
<td>c</td>
</tr>
<tr>
<td>3</td>
<td>Chinese Macao</td>
<td>c</td>
</tr>
<tr>
<td>4</td>
<td>Tokyo Japan Chinese</td>
<td>j</td>
</tr>
<tr>
<td>Test 5</td>
<td>Chinese Chinese Chinese Tokyo Japan</td>
<td>?</td>
</tr>
</tbody>
</table>
**Naive Bayes Example**

\[
\hat{P}(c) = \frac{N_c}{N}
\]

\[
\hat{P}(w \mid c) = \frac{\text{count}(w, c) + 1}{\text{count}(c) + |V|}
\]

**Priors:**

\[
P(c) = \frac{3}{4} \quad \frac{1}{4}
\]

\[
P(j) = \frac{3}{4} \quad \frac{1}{4}
\]

**Conditional Probabilities:**

\[
P(\text{Chinese} \mid c) = \frac{5+1}{8+6} = \frac{6}{14} = \frac{3}{7}
\]

\[
P(\text{Tokyo} \mid c) = \frac{0+1}{8+6} = \frac{1}{14}
\]

\[
P(\text{Japan} \mid c) = \frac{0+1}{8+6} = \frac{1}{14}
\]

\[
P(\text{Chinese} \mid j) = \frac{1+1}{3+6} = \frac{2}{9}
\]

\[
P(\text{Tokyo} \mid j) = \frac{1+1}{3+6} = \frac{2}{9}
\]

\[
P(\text{Japan} \mid j) = \frac{1+1}{3+6} = \frac{2}{9}
\]

**Choosing a class:**

\[
P(c \mid d5) \propto \frac{3}{4} \cdot \left(\frac{3}{7}\right)^3 \cdot \frac{1}{14} \cdot \frac{1}{14} 
\approx 0.0003
\]

\[
P(j \mid d5) \propto \frac{1}{4} \cdot \left(\frac{2}{9}\right)^3 \cdot \frac{2}{9} \cdot \frac{2}{9}
\approx 0.0001
\]

---

**Table:**

<table>
<thead>
<tr>
<th></th>
<th>Doc</th>
<th>Words</th>
<th>Class</th>
</tr>
</thead>
<tbody>
<tr>
<td>Training</td>
<td>1</td>
<td>Chinese Beijing Chinese</td>
<td>c</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>Chinese Chinese Shanghai</td>
<td>c</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>Chinese Macao</td>
<td>c</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>Tokyo Japan Chinese</td>
<td>j</td>
</tr>
<tr>
<td>Test</td>
<td>5</td>
<td>Chinese Chinese Chinese Tokyo Japan</td>
<td>?</td>
</tr>
</tbody>
</table>
# Features

## Top features for spam detection

<table>
<thead>
<tr>
<th>Rank</th>
<th>Category</th>
<th>Feature</th>
<th>Rank</th>
<th>Category</th>
<th>Feature</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Subject</td>
<td>Number of capitalized words</td>
<td>1</td>
<td>Subject</td>
<td>Min of the compression ratio for the bzip compressor</td>
</tr>
<tr>
<td>2</td>
<td>Subject</td>
<td>Sum of all the character lengths of words</td>
<td>2</td>
<td>Subject</td>
<td>Min of the compression ratio for the zlib compressor</td>
</tr>
<tr>
<td>3</td>
<td>Subject</td>
<td>Number of words containing letters and numbers</td>
<td>3</td>
<td>Subject</td>
<td>Min of character diversity of each word</td>
</tr>
<tr>
<td>4</td>
<td>Subject</td>
<td>Max of ratio of digit characters to all characters of each word</td>
<td>4</td>
<td>Subject</td>
<td>Min of the compression ratio for the lzw compressor</td>
</tr>
<tr>
<td>5</td>
<td>Header</td>
<td>Hour of day when email was sent</td>
<td>5</td>
<td>Subject</td>
<td>Max of the character lengths of words</td>
</tr>
</tbody>
</table>

### Spam URLs Features

<table>
<thead>
<tr>
<th>Rank</th>
<th>Category</th>
<th>Feature</th>
<th>Rank</th>
<th>Category</th>
<th>Feature</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>URL</td>
<td>The number of all URLs in an email</td>
<td>1</td>
<td>Header</td>
<td>Day of week when email was sent</td>
</tr>
<tr>
<td>2</td>
<td>URL</td>
<td>The number of unique URLs in an email</td>
<td>2</td>
<td>Payload</td>
<td>Number of characters</td>
</tr>
<tr>
<td>3</td>
<td>Payload</td>
<td>Number of words containing letters and numbers</td>
<td>3</td>
<td>Payload</td>
<td>Sum of all the character lengths of words</td>
</tr>
<tr>
<td>4</td>
<td>Payload</td>
<td>Min of the compression ratio for the bzip compressor</td>
<td>4</td>
<td>Header</td>
<td>Minute of hour when email was sent</td>
</tr>
<tr>
<td>5</td>
<td>Payload</td>
<td>Number of words containing only letters</td>
<td>5</td>
<td>Header</td>
<td>Hour of day when email was sent</td>
</tr>
</tbody>
</table>
Features

<table>
<thead>
<tr>
<th>Rank</th>
<th>Category</th>
<th>Feature</th>
<th>Rank</th>
<th>Category</th>
<th>Feature</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Subject</td>
<td>Number of capitalized words</td>
<td>1</td>
<td>Subject</td>
<td>Min of the compression ratio for the bzip2 compressor</td>
</tr>
<tr>
<td>2</td>
<td>Subject</td>
<td>Sum of all the character lengths of words</td>
<td>2</td>
<td>Subject</td>
<td>Min of the compression ratio for the zlib compressor</td>
</tr>
<tr>
<td>3</td>
<td>Subject</td>
<td>Number of words containing letters and numbers</td>
<td>3</td>
<td>Subject</td>
<td>Min of character diversity of each word</td>
</tr>
<tr>
<td>4</td>
<td>Subject</td>
<td>Max of ratio of digit characters to all characters of each word</td>
<td>4</td>
<td>Subject</td>
<td>Min of the compression ratio for the lzw compressor</td>
</tr>
<tr>
<td>5</td>
<td>Header</td>
<td>Hour of day when email was sent</td>
<td>5</td>
<td>Subject</td>
<td>Max of the character lengths of words</td>
</tr>
</tbody>
</table>

Spam URLs Features

<table>
<thead>
<tr>
<th>Rank</th>
<th>Category</th>
<th>Feature</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>URL</td>
<td>The number of all URLs in an email</td>
</tr>
<tr>
<td>2</td>
<td>URL</td>
<td>The number of unique URLs in an email</td>
</tr>
<tr>
<td>3</td>
<td>Payload</td>
<td>Number of words containing letters and numbers</td>
</tr>
<tr>
<td>4</td>
<td>Payload</td>
<td>Min of the compression ratio for the bzip2 compressor</td>
</tr>
<tr>
<td>5</td>
<td>Payload</td>
<td>Number of words containing only letters</td>
</tr>
</tbody>
</table>

Top features for spam detection

- In general, Naive Bayes can use any set of features, not just words:
- URLs, email addresses, Capitalization, ...
- Domain knowledge crucial to performance
Naive Bayes and Language Models
Naive Bayes and Language Models

• If features = bag of words, each class is a unigram language model!
Naive Bayes and Language Models

• If features = bag of words, each class is a unigram language model!

• For class $c$, assigning each word: $P(w | c)$
  
  assigning sentence: $P(S | c) = \prod_{w \in S} P(w | c)$
Naive Bayes and Language Models

- If features = bag of words, each class is a unigram language model!

- For class \( c \), assigning each word: \( P(w | c) \)

  assigning sentence: \( P(S | c) = \prod_{w \in S} P(w | c) \)

<table>
<thead>
<tr>
<th>Class pos</th>
<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>0.1</td>
<td>.05</td>
<td>0.01</td>
<td>0.1</td>
</tr>
</tbody>
</table>

\( P(s \mid \text{pos}) = 0.0000005 \)
Naive Bayes as a language model

- Which class assigns the higher probability to s?

<table>
<thead>
<tr>
<th>Model pos</th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>0.1</td>
<td>I</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.1</td>
<td>love</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.01</td>
<td>this</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.05</td>
<td>fun</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.1</td>
<td>film</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

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

\[
P(s|\text{pos}) \ ? \ P(s|\text{neg})
\]
Naive Bayes as a language model

- Which class assigns the higher probability to $s$?

$$P(s|\text{pos}) > P(s|\text{neg})$$
Evaluation
Evaluation

• Consider binary classification
Evaluation

- Consider binary classification

- Table of predictions
Evaluation

• Consider binary classification

• Table of predictions

<table>
<thead>
<tr>
<th></th>
<th>Positive</th>
<th>Negative</th>
</tr>
</thead>
<tbody>
<tr>
<td>Truth</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Positive</td>
<td>100</td>
<td>5</td>
</tr>
<tr>
<td>Negative</td>
<td>45</td>
<td>100</td>
</tr>
</tbody>
</table>
Evaluation

• Consider binary classification

• Table of predictions

<table>
<thead>
<tr>
<th>Predicted</th>
<th>Truth</th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Positive</td>
<td>Negative</td>
</tr>
<tr>
<td>Positive</td>
<td>100</td>
<td>5</td>
<td></td>
</tr>
<tr>
<td>Negative</td>
<td>45</td>
<td>100</td>
<td></td>
</tr>
</tbody>
</table>

Confusion Matrix
Evaluation

- Consider binary classification

- Table of predictions

<table>
<thead>
<tr>
<th>Truth</th>
<th>Positive</th>
<th>Negative</th>
</tr>
</thead>
<tbody>
<tr>
<td>Positive</td>
<td>100</td>
<td>5</td>
</tr>
<tr>
<td>Negative</td>
<td>45</td>
<td>100</td>
</tr>
</tbody>
</table>

Confusion Matrix
Evaluation

• Consider binary classification

• Table of predictions

<table>
<thead>
<tr>
<th>Predicted</th>
<th>Positive</th>
<th>Negative</th>
</tr>
</thead>
<tbody>
<tr>
<td>Positive</td>
<td>100</td>
<td>5</td>
</tr>
<tr>
<td>Negative</td>
<td>45</td>
<td>100</td>
</tr>
</tbody>
</table>

• Ideally, we want:
Evaluation

• Consider binary classification

• Table of predictions

<table>
<thead>
<tr>
<th>Predicted</th>
<th>Positive</th>
<th>Negative</th>
</tr>
</thead>
<tbody>
<tr>
<td>Positive</td>
<td>100</td>
<td>5</td>
</tr>
<tr>
<td>Negative</td>
<td>45</td>
<td>100</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Truth</th>
<th>Positive</th>
<th>Negative</th>
</tr>
</thead>
<tbody>
<tr>
<td>Positive</td>
<td>145</td>
<td>0</td>
</tr>
<tr>
<td>Negative</td>
<td>0</td>
<td>105</td>
</tr>
</tbody>
</table>

• Ideally, we want:
## Evaluation Metrics

<table>
<thead>
<tr>
<th>Predicted</th>
<th>Truth</th>
<th>Positive</th>
<th>Negative</th>
</tr>
</thead>
<tbody>
<tr>
<td>Positive</td>
<td>Truth</td>
<td>100</td>
<td>5</td>
</tr>
<tr>
<td>Negative</td>
<td>Truth</td>
<td>45</td>
<td>100</td>
</tr>
</tbody>
</table>
## Evaluation Metrics

- **True positive:** Predicted + and actual +

<table>
<thead>
<tr>
<th>Predicted</th>
<th>Truth</th>
<th>Positive</th>
<th>Negative</th>
</tr>
</thead>
<tbody>
<tr>
<td>Positive</td>
<td>100</td>
<td>5</td>
<td></td>
</tr>
<tr>
<td>Negative</td>
<td>45</td>
<td>100</td>
<td></td>
</tr>
</tbody>
</table>

• True positive: Predicted + and actual +
Evaluation Metrics

- True positive: Predicted + and actual +
- True negative: Predicted - and actual -
Evaluation Metrics

<table>
<thead>
<tr>
<th>Predicted</th>
<th>Truth</th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Positive</td>
<td>Negative</td>
<td></td>
</tr>
<tr>
<td>Positive</td>
<td>100</td>
<td>5</td>
<td></td>
</tr>
<tr>
<td>Negative</td>
<td>45</td>
<td>100</td>
<td></td>
</tr>
</tbody>
</table>

- True positive: Predicted + and actual +
- True negative: Predicted - and actual -
- False positive: Predicted + and actual -
## Evaluation Metrics

<table>
<thead>
<tr>
<th>Predicted</th>
<th>Truth</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Positive</td>
<td>Negative</td>
</tr>
<tr>
<td>Positive</td>
<td>100</td>
<td>5</td>
</tr>
<tr>
<td>Negative</td>
<td>45</td>
<td>100</td>
</tr>
</tbody>
</table>

- True positive: Predicted + and actual +
- True negative: Predicted - and actual -
- False positive: Predicted + and actual -
- False negative: Predicted - and actual +
Evaluation Metrics

<table>
<thead>
<tr>
<th>Predicted</th>
<th>Positive</th>
<th>Negative</th>
</tr>
</thead>
<tbody>
<tr>
<td>Positive</td>
<td>100</td>
<td>5</td>
</tr>
<tr>
<td>Negative</td>
<td>45</td>
<td>100</td>
</tr>
</tbody>
</table>

Truth

\[
\text{Accuracy} = \frac{TP + TN}{Total} = \frac{200}{250} = 80\%
\]

- True positive: Predicted + and actual +
- True negative: Predicted - and actual -
- False positive: Predicted + and actual -
- False negative: Predicted - and actual +
Evaluation Metrics

- **True positive**: Predicted + and actual +
- **True negative**: Predicted - and actual -
- **False positive**: Predicted + and actual -
- **False negative**: Predicted - and actual +

### Accuracy

\[
\text{Accuracy} = \frac{TP + TN}{Total} = \frac{200}{250} = 80\%
\]

**Truth**

<table>
<thead>
<tr>
<th></th>
<th>Positive</th>
<th>Negative</th>
</tr>
</thead>
<tbody>
<tr>
<td>Predicted</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Positive</td>
<td>100</td>
<td>5</td>
</tr>
<tr>
<td>Negative</td>
<td>45</td>
<td>100</td>
</tr>
</tbody>
</table>

**Coarse metric**
Evaluation Metrics

Accuracy = \frac{TP + TN}{Total} = \frac{200}{250} = 80\% 

Both have same accuracy, but clearly the models are behaving very differently.

Coarse metric
Precision and Recall
Precision and Recall

- Precision: % of selected classes that are correct
Precision and Recall

• Precision: % of selected classes that are correct

\[
\text{Precision}(+) = \frac{TP}{TP + FP} \quad \text{Precision}(-) = \frac{TN}{TN + FN}
\]
Precision and Recall

- Precision: % of selected classes that are correct

\[
\text{Precision}(+) = \frac{TP}{TP + FP} \quad \text{Precision}(-) = \frac{TN}{TN + FN}
\]
Precision and Recall

- Precision: % of selected classes that are correct

\[
\text{Precision}(+) = \frac{TP}{TP + FP} \quad \text{Precision}(-) = \frac{TN}{TN + FN}
\]
Precision and Recall

• Precision: % of selected classes that are correct

\[
\text{Precision( + )} = \frac{TP}{TP + FP} \quad \text{Precision( - )} = \frac{TN}{TN + FN}
\]

• Recall: % of correct items selected
Precision and Recall

• Precision: % of selected classes that are correct

\[
\text{Precision}(+) = \frac{TP}{TP + FP} \quad \text{Precision}(-) = \frac{TN}{TN + FN}
\]

• Recall: % of correct items selected

\[
\text{Recall}(+) = \frac{TP}{TP + FN} \quad \text{Recall}(-) = \frac{TN}{TN + FP}
\]
F-Score
F-Score

• Combined measure using precision and recall
F-Score

- Combined measure using precision and recall
- Harmonic mean of Precision and Recall
F-Score

• Combined measure using precision and recall

• Harmonic mean of Precision and Recall

\[ F_1 = \frac{2 \cdot \text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}} \]
F-Score

- Combined measure using precision and recall
- Harmonic mean of Precision and Recall

\[ F_1 = \frac{2 \cdot \text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}} \]
F-Score

• Combined measure using precision and recall

• Harmonic mean of Precision and Recall

\[ F_1 = \frac{2 \cdot \text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}} \]

• Or more generally,
F-Score

• Combined measure using precision and recall

• Harmonic mean of Precision and Recall

\[ F_1 = \frac{2 \cdot \text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}} \]

• Or more generally,

\[ F_\beta = \frac{(1 + \beta^2) \cdot \text{Precision} \cdot \text{Recall}}{\beta^2 \cdot \text{Precision} + \text{Recall}} \]
Choosing Beta

Which value of Beta maximizes $F_{\beta}$ for the positive class?

A. $\beta = 0.5$

B. $\beta = 1$

C. $\beta = 2$
Aggregating scores

• We now have Precision, Recall, F1 for each class

• Can we combine them for an overall score?

  • Macro-average: Compute for each class, then average

  • Micro-average: Collect predictions for all classes and jointly evaluate
Macro vs Micro average

Class 1

<table>
<thead>
<tr>
<th>Classifier: yes</th>
<th>Truth: yes</th>
<th>Truth: no</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>10</td>
<td></td>
</tr>
<tr>
<td>Classifier: no</td>
<td>10</td>
<td>970</td>
</tr>
</tbody>
</table>

Class 2

<table>
<thead>
<tr>
<th>Classifier: yes</th>
<th>Truth: yes</th>
<th>Truth: no</th>
</tr>
</thead>
<tbody>
<tr>
<td>90</td>
<td>10</td>
<td></td>
</tr>
<tr>
<td>Classifier: no</td>
<td>10</td>
<td>890</td>
</tr>
</tbody>
</table>

Micro Ave. Table

<table>
<thead>
<tr>
<th>Classifier: yes</th>
<th>Truth: yes</th>
<th>Truth: no</th>
</tr>
</thead>
<tbody>
<tr>
<td>100</td>
<td>20</td>
<td></td>
</tr>
<tr>
<td>Classifier: no</td>
<td>20</td>
<td>1860</td>
</tr>
</tbody>
</table>

• Macroaveraged precision: \((0.5 + 0.9)/2 = 0.7\)
• Microaveraged precision: \(100/120 = .83\)
• Microaveraged score is dominated by score on common classes
Validation

Train

Validation

Test
• Choose a metric: Precision/Recall/F1
Validation

- Choose a metric: Precision/Recall/F1
- Optimize for metric on Validation (aka Development) set
Validation

- Choose a metric: Precision/Recall/F1
- Optimize for metric on Validation (aka Development) set
- Finally evaluate on ‘unseen’ test set
Validation

- Choose a metric: Precision/Recall/F1
- Optimize for metric on Validation (aka Development) set
- Finally evaluate on ‘unseen’ test set
- Choice of data splits may affect your evaluation
Validation

- Choose a metric: Precision/Recall/F1
- Optimize for metric on Validation (aka Development) set
- Finally evaluate on ‘unseen’ test set
- Choice of data splits may affect your evaluation
- Cross-validation:
Validation

• Choose a metric: Precision/Recall/F1

• Optimize for metric on Validation (aka Development) set

• Finally evaluate on ‘unseen’ test set

• Choice of data splits may affect your evaluation

• Cross-validation:
  • Repeatedly sample several train-val splits
Validation

- Choose a metric: Precision/Recall/F1

- Optimize for metric on Validation (aka Development) set

- Finally evaluate on ‘unseen’ test set

- Choice of data splits may affect your evaluation

- Cross-validation:
  - Repeatedly sample several train-val splits
Validation

- Choose a metric: Precision/Recall/F1
- Optimize for metric on Validation (aka Development) set
- Finally evaluate on ‘unseen’ test set
- Choice of data splits may affect your evaluation
- Cross-validation:
  - Repeatedly sample several train-val splits
  - Reduces bias due to sampling errors
Advantages of Naive Bayes
Advantages of Naive Bayes

• Very fast, low storage requirements
Advantages of Naive Bayes

- Very fast, low storage requirements
- Robust to irrelevant features
  - Irrelevant features cancel each other without affecting results
Advantages of Naive Bayes

• Very fast, low storage requirements

• Robust to irrelevant features
  Irrelevant features cancel each other without affecting results

• Very good in domains with many equally important features
  Decision trees suffer from fragmentation in such cases — especially if little data
Advantages of Naive Bayes

• Very fast, low storage requirements

• Robust to irrelevant features
  Irrelevant features cancel each other without affecting results

• Very good in domains with many equally important features
  Decision trees suffer from fragmentation in such cases — especially if little data

• Optimal if the independence assumptions hold
  If assumed independence is correct, this is the ‘Bayes optimal’ classifier
Advantages of Naive Bayes

- Very fast, low storage requirements
- Robust to irrelevant features
  Irrelevant features cancel each other without affecting results
- Very good in domains with many equally important features
  Decision trees suffer from fragmentation in such cases — especially if little data
- Optimal if the independence assumptions hold
  If assumed independence is correct, this is the ‘Bayes optimal’ classifier
- A good dependable baseline for text classification
  However, other classifiers can give better accuracy
Practical Naive Bayes
Practical Naive Bayes

- Small data sizes:
  - Naive Bayes is great! (high bias)
  - Rule-based classifiers might work well too
Practical Naive Bayes

- Small data sizes:
  - Naive Bayes is great! (high bias)
  - Rule-based classifiers might work well too

- Medium size datasets:
  - More advanced classifiers might perform better (e.g. SVM, logistic regression)
Practical Naive Bayes

- **Small data sizes:**
  - Naive Bayes is great! (high bias)
  - Rule-based classifiers might work well too

- **Medium size datasets:**
  - More advanced classifiers might perform better (e.g. SVM, logistic regression)

- **Large datasets:**
  - Naive Bayes becomes competitive again (although most classifiers work well)
Failings of Naive Bayes (I)
Failings of Naive Bayes (I)

Independence assumptions are too strong
Failings of Naive Bayes (1)

Independence assumptions are too strong

<table>
<thead>
<tr>
<th>x1</th>
<th>x2</th>
<th>Class: $x_1 \text{ XOR } x_2$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>
Failings of Naive Bayes (1)

Independence assumptions are too strong

<table>
<thead>
<tr>
<th>x1</th>
<th>x2</th>
<th>Class: x₁ XOR x₂</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>
Failings of Naive Bayes (1)

Independence assumptions are too strong

<table>
<thead>
<tr>
<th>x1</th>
<th>x2</th>
<th>Class: x1 XOR x2</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>
Failings of Naive Bayes (1)

Independence assumptions are too strong

<table>
<thead>
<tr>
<th>x1</th>
<th>x2</th>
<th>Class: x₁ XOR x₂</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

• XOR problem: Naive Bayes cannot learn a decision boundary
Failings of Naive Bayes (1)

Independence assumptions are too strong

<table>
<thead>
<tr>
<th>x1</th>
<th>x2</th>
<th>Class: x₁ XOR x₂</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

- XOR problem: Naive Bayes cannot learn a decision boundary
- Both variables are jointly required to predict class
Failings of Naive Bayes (2)
Failings of Naive Bayes (2)

Class imbalance
Failings of Naive Bayes (2)

Class imbalance

- One or more classes have more instances than others
Failings of Naive Bayes (2)

Class imbalance

- One or more classes have more instances than others
- Data skew causes NB to prefer one class over the other
Failings of Naive Bayes (2)

Class imbalance

• One or more classes have more instances than others

• Data skew causes NB to prefer one class over the other

• Potential solution: Complement Naive Bayes (Rennie et al., 2003)
Failings of Naive Bayes (2)

Class imbalance

• One or more classes have more instances than others

• Data skew causes NB to prefer one class over the other

• Potential solution: Complement Naive Bayes (Rennie et al., 2003)

\[
\hat{p}(w_i | \tilde{c}_j) = \frac{\sum_{c \neq c_j} \text{Count}(w_i, c)}{\sum_{c \neq c_j} \sum_{w} \text{Count}(w, c)} \rightarrow \text{Count # times word occurs in classes other than } c
\]
Failings of Naive Bayes (3)

(assuming $\epsilon$ added for smoothing)
Failings of Naive Bayes (3)

Weight magnitude errors

(assuming $\epsilon$ added for smoothing)
Failings of Naive Bayes (3)

Weight magnitude errors

- Classes with larger weights are preferred

(assuming $\epsilon$ added for smoothing)
Failings of Naive Bayes (3)

Weight magnitude errors

- Classes with larger weights are preferred

- 10 documents with class=MA and "Boston" occurring once each

(assuming $\epsilon$ added for smoothing)
Failings of Naive Bayes (3)

Weight magnitude errors

• Classes with larger weights are preferred

• 10 documents with class=MA and “Boston” occurring once each

• 10 documents with class=CA and “San Francisco” occurring once each

(assuming $\epsilon$ added for smoothing)
Failings of Naive Bayes (3)

Weight magnitude errors

- Classes with larger weights are preferred
- 10 documents with class=MA and "Boston" occurring once each
- 10 documents with class=CA and "San Francisco" occurring once each
- New document: "Boston Boston Boston San Francisco San Francisco"

(assuming $\epsilon$ added for smoothing)
Failings of Naive Bayes (3)

Weight magnitude errors

- Classes with larger weights are preferred
- 10 documents with class=MA and "Boston" occurring once each
- 10 documents with class=CA and "San Francisco" occurring once each
- New document: "Boston Boston Boston San Francisco San Francisco"

(assuming $\epsilon$ added for smoothing)
Failings of Naive Bayes (3)

Weight magnitude errors

- Classes with larger weights are preferred
- 10 documents with class=MA and “Boston” occurring once each
- 10 documents with class=CA and “San Francisco” occurring once each
- New document: “Boston Boston Boston San Francisco San Francisco”

\[
P(\text{class} = CA \mid \text{document}) \ ? \ P(\text{class} = MA \mid \text{document})
\]  
(assuming \(\epsilon\) added for smoothing)
Practical text classification
Practical text classification

- Domain knowledge is crucial to selecting good features
Practical text classification

• Domain knowledge is crucial to selecting good features

• Handle class imbalance by re-weighting classes
Practical text classification

• Domain knowledge is crucial to selecting good features

• Handle class imbalance by re-weighting classes

• Use log scale operations instead of multiplying probabilities
Practical text classification

• Domain knowledge is crucial to selecting good features

• Handle class imbalance by re-weighting classes

• Use log scale operations instead of multiplying probabilities

• Since $\log(xy) = \log(x) + \log(y)$
  Better to sum logs of probabilities instead of multiplying probabilities
Practical text classification

- Domain knowledge is crucial to selecting good features
- Handle class imbalance by re-weighting classes
- Use log scale operations instead of multiplying probabilities
  - Since \( \log(xy) = \log(x) + \log(y) \)
    - Better to sum logs of probabilities instead of multiplying probabilities
- Class with highest un-normalized log probability score is still most probable
  \[
  C_{NB} = \arg \max_{c_j \in C} \log P(c_j) + \sum_{i \in \text{positions}} \log P(x_i \mid c_j)
  \]
Practical text classification

- Domain knowledge is crucial to selecting good features
- Handle class imbalance by re-weighting classes
- Use log scale operations instead of multiplying probabilities
  - Since \( \log(xy) = \log(x) + \log(y) \)
  - Better to sum logs of probabilities instead of multiplying probabilities
- Class with highest un-normalized log probability score is still most probable
  \[
  C_{NB} = \arg \max_{c_j \in C} \log P(c_j) + \sum_{i \in \text{positions}} \log P(x_i | c_j)
  \]
- Model is now just max of sum of weights