CS 6120/CS4120: Natural Language Processing

Instructor: Prof. Lu Wang
College of Computer and Information Science
Northeastern University
Webpage: www.ccs.neu.edu/home/luwang
Outline

• Text Categorization/Classification
• Naïve Bayes
• Evaluation
Positive or negative movie review?

- unbelievably disappointing
- Full of zany characters and richly applied satire, and some great plot twists
- this is the greatest screwball comedy ever filmed
- It was pathetic. The worst part about it was the boxing scenes.
Male or female author?

1. By 1925 present-day Vietnam was divided into three parts under French colonial rule. The southern region embracing Saigon and the Mekong delta was the colony of Cochin-China; the central area with its imperial capital at Hue was the protectorate of Annam...

2. Clara never failed to be astonished by the extraordinary felicity of her own name. She found it hard to trust herself to the mercy of fate, which had managed over the years to convert her greatest shame into one of her greatest assets...

Text Classification

• Assigning subject categories, topics, or genres
• Spam detection
• Authorship identification
• Age/gender identification
• Language Identification
• Sentiment analysis
• ...
Text Classification: definition

• **Input:**
  - a document $d$
  - a fixed set of classes $C = \{c_1, c_2, \ldots, c_J\}$

• **Output:** a predicted class $c \in C$
Classification Methods: Hand-coded rules

- Rules based on combinations of words or other features
  - spam: black-list-address OR (“dollars” AND “have been selected”)
- Accuracy can be high
  - If rules carefully refined by expert
- But building and maintaining these rules is expensive
Classification Methods:
Supervised Machine Learning

• **Input:**
  • a document \( d \)
  • a fixed set of classes \( C = \{c_1, c_2, ..., c_J\} \)
  • A training set of \( m \) hand-labeled documents \((d_1, y_1), ..., (d_m, y_m), y_i \text{ is in } C\)

• **Output:**
  • a learned classifier \( \gamma: d \rightarrow c \)
Classification Methods: Supervised Machine Learning

- Any kind of classifier
  - Naïve Bayes
  - Logistic regression
  - Support-vector machines
  - k-Nearest Neighbors
  - Neural networks
  - ...
Outline

• Text Categorization/Classification
• Naïve Bayes
• Evaluation
Naïve Bayes Classifier
Naïve Bayes Intuition

• Simple (“naïve”) classification method based on Bayes rule
• Relies on very simple representation of document
  • Bag of words
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

$$\gamma() = c$$

<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>
Bayes’ Rule Applied to Documents and Classes

- For a document \(d\) and a class \(c\)

\[
P(c \mid d) = \frac{P(d \mid c)P(c)}{P(d)}
\]
Naïve Bayes Classifier (I)

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

\[ = \arg\max_{c \in C} \frac{P(d \mid c)P(c)}{P(d)} \]

\[ = \arg\max_{c \in C} P(d \mid c)P(c) \]

MAP is “maximum a posteriori” = most likely class

Bayes Rule

Dropping the denominator
Naïve Bayes Classifier (I)

\[ c_{MAP} = \underset{c \in C}{\text{argmax}} \ P(c \mid d) \]

\[ = \underset{c \in C}{\text{argmax}} \ \frac{P(d \mid c)P(c)}{P(d)} \]

\[ = \underset{c \in C}{\text{argmax}} \ P(d \mid c)P(c) \]

MAP is “maximum a posteriori” = most likely class

Bayes Rule

Dropping the denominator

Why we can do this?
Naïve Bayes Classifier (II)

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

\[ = \arg \max_{c \in C} P(x_1, x_2, \ldots, x_n \mid c)P(c) \]

Document \( d \) represented as features \( x_1..x_n \)
Naïve Bayes Classifier (IV)

\[ c_{\text{MAP}} = \arg\max_{c \in C} P(x_1, x_2, \ldots, x_n \mid c) P(c) \]

\( O(|X|^n \cdot |C|) \) parameters

\(|X|\) represents the maximum number of possible values for \(x_i\)
\[ P(x_1, x_2, \ldots, x_n \mid c) \]

- **Bag of Words assumption**: Assume position doesn’t matter
- **Conditional Independence**: Assume the feature probabilities \( P(x_i \mid c_j) \) are independent given the class \( c \).

\[ P(x_1, \ldots, x_n \mid c) = P(x_1 \mid c) \cdot P(x_2 \mid c) \cdot P(x_3 \mid c) \cdot \ldots \cdot P(x_n \mid c) \]
Multinomial Naïve Bayes Classifier

\[
c_{\text{MAP}} = \arg\max_{c \in C} P(x_1, x_2, \ldots, x_n \mid c) P(c)
\]

\[
c_{\text{NB}} = \arg\max_{c \in C} P(c) \prod_{x \in X} P(x \mid c)
\]
Applying Multinomial Naive Bayes Classifiers to Text Classification

\[ c_{NB} = \operatorname{argmax}_{c_j \in C} P(c_j) \prod_{i \in \text{positions}} P(x_i | c_j) \]
Learning for Naïve Bayes Model
Learning the Multinomial Naïve Bayes Model

• First attempt: maximum likelihood estimates
  • simply use the frequencies in the data

\[
\hat{P}(c_j) = \frac{\text{doccount}(C = c_j)}{N_{\text{doc}}}
\]

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

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

fraction of times word \( w_i \) appears among all words in documents of class \( c_j \)
Problem with Maximum Likelihood

• What if we have seen no training documents with the word *fantastic* and classified in the topic *positive (thumbs-up)*?

\[
\hat{P}(\text{"fantastic"} \mid \text{positive}) = \frac{\text{count("fantastic", positive)}}{\sum_{w \in V} \text{count}(w, \text{positive})} = 0
\]

• Zero probabilities cannot be conditioned away, no matter the other evidence!

\[
c_{MAP} = \underset{c}{\text{argmax}} \, \hat{P}(c) \prod_i \hat{P}(x_i \mid c)
\]
Laplace (add-1) smoothing for Naïve Bayes

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

\[
= \frac{\text{count}(w_i, c) + 1}{\left( \sum_{w \in V} \text{count}(w, c) \right) + |V|}
\]
Multinomial Naïve Bayes: Learning

- From training corpus, extract *Vocabulary*

- Calculate $P(c_j)$ terms
  - For each $c_j$ in $C$ do
    
    $\text{docs}_{j} \leftarrow$ all docs with class $= c_j$

    $$P(c_j) \leftarrow \frac{|\text{docs}_{j}|}{|\text{total # documents}|}$$

- Calculate $P(w_k \mid c_j)$ terms
  - $\text{Text}_j \leftarrow$ single doc containing all $\text{docs}_j$
  - For each word $w_k$ in *Vocabulary*
    
    $n_k \leftarrow$ # of occurrences of $w_k$ in $\text{Text}_j$

    $$P(w_k \mid c_j) \leftarrow \frac{n_k + \alpha}{n + \alpha |\text{Vocabulary}|}$$
Multinomial Naïve Bayes: Learning

- From training corpus, extract *Vocabulary*

- Calculate $P(c_j)$ terms
  - For each $c_j$ in $C$ do
    
    $\text{docs}_j \leftarrow$ all docs with class = $c_j$

    $P(c_j) \leftarrow \frac{|\text{docs}_j|}{|\text{total # documents}|}$

- Calculate $P(w_k \mid c_j)$ terms
  - $\text{Text}_j \leftarrow$ single doc containing all $\text{docs}_j$
  - For each word $w_k$ in *Vocabulary*
    
    $n_k \leftarrow$ # of occurrences of $w_k$ in $\text{Text}_j$

    $P(w_k \mid c_j) \leftarrow \frac{n_k + \alpha}{n + \alpha |\text{Vocabulary}|}$

A more general form: add-$\alpha$ smoothing!
Naïve Bayes and Language Modeling

• Naïve bayes classifiers can use any sort of feature
  • URL, email address, dictionaries, network features

• But if, as in the previous slides
  • We use only word features
  • we use all of the words in the text (not a subset)

• Then
  • Naïve bayes has an important similarity to language modeling.
Each class = a unigram language model

- Assigning each word: \( P(\text{word} \mid c) \)
- Assigning each sentence: \( P(\text{sentence} \mid c) = \prod P(\text{word} \mid c) \)

<table>
<thead>
<tr>
<th>Class pos</th>
<th>P(sentence \mid pos)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.1</td>
<td>l</td>
</tr>
<tr>
<td>0.1</td>
<td>love</td>
</tr>
<tr>
<td>0.01</td>
<td>this</td>
</tr>
<tr>
<td>0.05</td>
<td>fun</td>
</tr>
<tr>
<td>0.1</td>
<td>film</td>
</tr>
</tbody>
</table>

\[ P(\text{sentence} \mid \text{pos}) = 0.00000005 \]
Naïve Bayes as a Language Model

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

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

$P(s|\text{pos}) > P(s|\text{neg})$
An Example
Choosing a class:

\[ P(c | d5) \propto 3/4 * (3/7)^3 * 1/14 * 1/14 \]
\[ \approx 0.0003 \]

\[ P(j | d5) \propto 1/4 * (2/9)^3 * 2/9 * 2/9 \]
\[ \approx 0.0001 \]

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

Priors:

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

Conditional Probabilities:

\[ P(\text{Chinese} | c) = \frac{5+1}{8+6} = \frac{6}{14} = \frac{3}{7} \]
\[ P(\text{Tokyo} | c) = \frac{0+1}{8+6} = \frac{1}{14} \]
\[ P(\text{Japan} | c) = \frac{0+1}{8+6} = \frac{1}{14} \]
\[ P(\text{Chinese} | j) = \frac{1+1}{3+6} = \frac{2}{9} \]
\[ P(\text{Tokyo} | j) = \frac{1+1}{3+6} = \frac{2}{9} \]
\[ P(\text{Japan} | j) = \frac{1+1}{3+6} = \frac{2}{9} \]
Summary: Naive Bayes is Not So Naive

• 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

• Optimal if the independence assumptions hold:
  • If assumed independence is correct, then it is the Bayes Optimal Classifier for problem

• A good dependable baseline for text classification
Outline

• Text Categorization/Classification
• Naïve Bayes
• Evaluation
Evaluation
The 2-by-2 contingency table (or confusion matrix)

<table>
<thead>
<tr>
<th></th>
<th>correct</th>
<th>not correct</th>
</tr>
</thead>
<tbody>
<tr>
<td>selected</td>
<td>tp (true positive)</td>
<td>fp (false positive)</td>
</tr>
<tr>
<td>not selected</td>
<td>fn (false negative)</td>
<td>tn (true negative)</td>
</tr>
</tbody>
</table>

For example,
- Which set of documents are related to the topic of NLP?
- Which set of documents are written by Shakespeare?
The 2-by-2 contingency table

<table>
<thead>
<tr>
<th></th>
<th>correct</th>
<th>not correct</th>
</tr>
</thead>
<tbody>
<tr>
<td>selected</td>
<td>tp</td>
<td>fp</td>
</tr>
<tr>
<td>not selected</td>
<td>fn</td>
<td>tn</td>
</tr>
</tbody>
</table>
Precision and recall

- **Precision**: % of selected items that are correct, $\frac{tp}{tp+fp}$
- **Recall**: % of correct items that are selected, $\frac{tp}{tp+fn}$

<table>
<thead>
<tr>
<th></th>
<th>correct</th>
<th>not correct</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>selected</strong></td>
<td>$tp$</td>
<td>$fp$</td>
</tr>
<tr>
<td><strong>not selected</strong></td>
<td>$fn$</td>
<td>$tn$</td>
</tr>
</tbody>
</table>
A combined measure: F-measure or F-score

- A combined measure that assesses the P/R tradeoff is F measure (weighted harmonic mean):

\[
F = \frac{1}{\frac{1}{\alpha \frac{1}{P}} + (1-\alpha) \frac{1}{R}}
\]

- People usually use balanced F1 measure
  - i.e., \( \alpha = \frac{1}{2} \), \( F = \frac{2PR}{P+R} \)
Text Classification Evaluation
More Than Two Classes: Sets of binary classifiers

- Dealing with any-of or multivalue classification
  - A document can belong to 0, 1, or >1 classes.

- For each class $c \in C$
  - Build a classifier $\gamma_c$ to distinguish $c$ from all other classes $c' \in C$

- Given test doc $d$,
  - Evaluate it for membership in each class using each $\gamma_c$
  - $d$ belongs to any class for which $\gamma_c$ returns true
More Than Two Classes: Sets of binary classifiers

• **One-of** or multinomial classification
  • Classes are mutually exclusive: each document in exactly one class

• For each class \( c \in C \)
  • Build a classifier \( \gamma_c \) to distinguish \( c \) from all other classes \( c' \in C \)

• Given test doc \( d \),
  • Evaluate it for membership in each class using each \( \gamma_c \)
  • \( d \) belongs to the **one** class with maximum score
Confusion matrix \( c \)

- For each pair of classes \(<c_1, c_2>\) how many documents from \( c_1 \) were incorrectly assigned to \( c_2 \)?
  - \( c_{3,2} \): 90 wheat documents incorrectly assigned to poultry

<table>
<thead>
<tr>
<th>Docs in test set</th>
<th>Assigned UK</th>
<th>Assigned poultry</th>
<th>Assigned wheat</th>
<th>Assigned coffee</th>
<th>Assigned interest</th>
<th>Assigned trade</th>
</tr>
</thead>
<tbody>
<tr>
<td>True UK</td>
<td>95</td>
<td>1</td>
<td>13</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>True poultry</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>True wheat</td>
<td>10</td>
<td>90</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>True coffee</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>34</td>
<td>3</td>
<td>7</td>
</tr>
<tr>
<td>True interest</td>
<td>0</td>
<td>1</td>
<td>2</td>
<td>13</td>
<td>26</td>
<td>5</td>
</tr>
<tr>
<td>True trade</td>
<td>0</td>
<td>0</td>
<td>2</td>
<td>14</td>
<td>5</td>
<td>10</td>
</tr>
</tbody>
</table>
Per class evaluation measures

Recall:
Fraction of docs in class $i$ classified correctly:

\[
\frac{\sum c_{ii}}{\sum c_{ij}}
\]

Precision:
Fraction of docs assigned class $i$ that are actually about class $i$:

\[
\frac{c_{ii}}{\sum c_{ji}}
\]

Accuracy: (1 - error rate)
Fraction of docs classified correctly:

\[
\frac{\sum c_{ii}}{\sum \sum c_{ij}}
\]
Micro- vs. Macro-Averaging

• If we have more than one class, how do we combine multiple performance measures into one quantity?

• **Macroaveraging**: Compute performance for each class, then average.

• **Microaveraging**: Collect decisions for all classes, compute contingency table, evaluate.
# Micro- vs. Macro-Averaging: Example

## Class 1

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

## Class 2

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

## Micro Ave. Table

<table>
<thead>
<tr>
<th></th>
<th>Truth: yes</th>
<th>Truth: no</th>
</tr>
</thead>
<tbody>
<tr>
<td>Classifier: yes</td>
<td>100</td>
<td>20</td>
</tr>
<tr>
<td>Classifier: no</td>
<td>20</td>
<td>1860</td>
</tr>
</tbody>
</table>
Micro- vs. Macro-Averaging: Example

<table>
<thead>
<tr>
<th>Class 1</th>
<th>Class 2</th>
<th>Micro Ave. Table</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Truth: yes</td>
<td>Truth: no</td>
<td>Truth: yes</td>
</tr>
<tr>
<td>Classifier: yes</td>
<td>10</td>
<td>10</td>
</tr>
<tr>
<td>Classifier: no</td>
<td>10</td>
<td>970</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Classifier: no</td>
</tr>
</tbody>
</table>

- Macroaveraged precision: \((0.5 + 0.9)/2 = 0.7\)
- Microaveraged precision: \(100/120 = .83\)
Development Test Sets and Cross-validation

Metric: P/R/F1 or Accuracy

Unseen test set
• avoid overfitting (‘tuning to the test set’)
• more conservative estimate of performance

Cross-validation over multiple splits
• Handle sampling errors from different datasets
• Pool results over each split
• Compute pooled dev set performance