Lecture 14: Naïve Bayes Classifier, Evaluation Methods

LING 1330/2330: Introduction to Computational Linguistics

Na-Rae Han
Overview

- Text classification; Naïve Bayes classifier
  - Language and Computers: Ch.5 Classifying documents
  - NLTK book: Ch.6 Learning to classify text

- Evaluating the performance of a system
  - Language and Computers:
    - Ch.5.4 Measuring success, 5.4.1 Base rates
  - NLTK book: Ch.6.3 Evaluation
  - Cross-validation
  - Accuracy vs. precision vs. recall
  - F-measure
Bayes' Theorem, recap

\[ P(B \mid A) = \frac{P(B, A)}{P(A)} = \frac{P(A \mid B) \cdot P(B)}{P(A)} \]

- B: Pitt closing, A: snowing
- \( P(B \mid A) \): probability of Pitt closing, given snowy weather
- \( P(B, A) \): probability of Pitt closing and snowing

1: The probability of Pitt closing given it's snowing is equal to the probability of Pitt closing and snowing, divided by the probability of snowing.

2: The probability of Pitt closing AND it's snowing is equal to the probability of Pitt closing (=prior) multiplied by the probability of snowing given that Pitt is closed.

\( \leftrightarrow \) Corollary of 1! You get this by swapping A and B and solving for \( P(B, A) \)
Naïve Bayes Assumption

- Given a label, a set of features $f_1, f_2, \ldots, f_n$ are generated with different probabilities.
- The features are independent of each other; $f_x$ occurring does not affect $f_y$ occurring, etc.

$\Rightarrow$ Naïve Bayes Assumption

- This feature independence assumption simplifies combining contributions of features; you just multiply their probabilities:

$$P(f_1, f_2, \ldots, f_n \mid L) = P(f_1 \mid L) \cdot P(f_2 \mid L) \cdot \ldots \cdot P(f_n \mid L)$$

$\Leftarrow$ "Naïve" because features are often inter-dependent.

$\Leftarrow f_1$: 'contains-linguistics:YES' and $f_2$: 'contains-syntax:YES' are not independent.
Bayes' Theorem & spam likelihood

1. \[ P(\text{SPAM} \mid D) = \frac{P(\text{SPAM}, D)}{P(D)} = \frac{P(\text{SPAM}, D)}{P(\text{SPAM}, D) + P(\text{HAM}, D)} \]

2. \[ P(\text{SPAM}, D) = P(D \mid \text{SPAM}) \times P(\text{SPAM}) = P(\text{SPAM}) \times P(D \mid \text{SPAM}) = P(\text{SPAM}) \times P(f_1, f_2, \ldots, f_n \mid \text{SPAM}) = P(\text{SPAM}) \times P(f_1 \mid \text{SPAM}) \times \ldots \times P(f_n \mid \text{SPAM}) \]

- **SPAM**: document is spam, D: a specific document occurs
- **P(\text{SPAM} \mid D)**: probability of document being SPAM, given a particular document
- **P(\text{SPAM}, D)**: probability of D occurring and it being SPAM
- **Which means**: we can calculate **P(\text{SPAM} \mid D)** from
  \[ P(\text{SPAM}, D) \text{ and } P(\text{HAM}, D), \text{ which are calculated by 2.}. \]
Jane Austen or Herman Melville?

- *I never met with a disposition more truly amiable.*
- *But Queequeg, do you see, was a creature in the transition stage -- neither caterpillar nor butterfly.*
- *Oh, my sweet cardinals!*

Task: build a Naïve Bayes classifier and explore it

http://www.pitt.edu/~naraehan/ling1330/hw10.html

What did you all think?
**whosaid: a Naïve Bayes classifier**

- **How did the classifier do?**
  - 0.951 accuracy on shared model.

- **Training set:** 15,152 sentences
  - 6,672 are Austen
    - \( P(\text{austen}) = 0.44 \)
    - **Austen prior**
  - 8,480 are Melville
    - \( P(\text{melville}) = 0.56 \)
    - **Melville prior**
  - Sentences have a higher chance of being Melville out of the gate!

---

**Diagram:**

- **1,000 Test**
  - Used for Evaluation: 0.951 accuracy

- **1,000 Dev-test**

- **15,152 sents Training**
  - Used for error analysis: aa, am, mm, ma
### Informative features

<table>
<thead>
<tr>
<th>Feature</th>
<th>Count</th>
<th>austen : melvil</th>
</tr>
</thead>
<tbody>
<tr>
<td>'contains-emma'</td>
<td>1</td>
<td>1864.5 : 1.0</td>
</tr>
<tr>
<td>'contains-whale'</td>
<td>1</td>
<td>1522.5 : 1.0</td>
</tr>
<tr>
<td>'contains-harriet'</td>
<td>1</td>
<td>1048.5 : 1.0</td>
</tr>
<tr>
<td>'contains-weston'</td>
<td>1</td>
<td>926.5 : 1.0</td>
</tr>
<tr>
<td>'contains-knightley'</td>
<td>1</td>
<td>840.1 : 1.0</td>
</tr>
<tr>
<td>'contains-elton'</td>
<td>1</td>
<td>771.5 : 1.0</td>
</tr>
<tr>
<td>'contains-ship'</td>
<td>1</td>
<td>696.3 : 1.0</td>
</tr>
<tr>
<td>'contains-ahab'</td>
<td>1</td>
<td>666.4 : 1.0</td>
</tr>
<tr>
<td>'contains-woodhouse'</td>
<td>1</td>
<td>652.0 : 1.0</td>
</tr>
<tr>
<td>'contains-jane'</td>
<td>1</td>
<td>613.9 : 1.0</td>
</tr>
<tr>
<td>'contains-fairfax'</td>
<td>1</td>
<td>507.1 : 1.0</td>
</tr>
<tr>
<td>'contains-churchill'</td>
<td>1</td>
<td>469.0 : 1.0</td>
</tr>
<tr>
<td>'contains-boat'</td>
<td>1</td>
<td>424.1 : 1.0</td>
</tr>
<tr>
<td>'contains-miss'</td>
<td>1</td>
<td>381.1 : 1.0</td>
</tr>
<tr>
<td>'contains-hartfield'</td>
<td>1</td>
<td>362.2 : 1.0</td>
</tr>
<tr>
<td>'contains-whales'</td>
<td>1</td>
<td>345.4 : 1.0</td>
</tr>
<tr>
<td>'contains-queequeg'</td>
<td>1</td>
<td>337.5 : 1.0</td>
</tr>
<tr>
<td>'contains-stubb'</td>
<td>1</td>
<td>325.0 : 1.0</td>
</tr>
<tr>
<td>'contains-sperm'</td>
<td>1</td>
<td>318.7 : 1.0</td>
</tr>
<tr>
<td>'contains-bates'</td>
<td>1</td>
<td>311.4 : 1.0</td>
</tr>
</tbody>
</table>
Informative features, noCharNames

<table>
<thead>
<tr>
<th>Feature</th>
<th>Melville : Austen</th>
<th>Ratio</th>
</tr>
</thead>
<tbody>
<tr>
<td>'contains-whale', 1</td>
<td>melvil : austen</td>
<td>1522.5: 1.0</td>
</tr>
<tr>
<td>'contains-ship', 1</td>
<td>melvil : austen</td>
<td>696.3: 1.0</td>
</tr>
<tr>
<td>'contains-boat', 1</td>
<td>melvil : austen</td>
<td>424.1: 1.0</td>
</tr>
<tr>
<td>'contains-miss', 1</td>
<td>austen : melvil</td>
<td>381.7: 1.0</td>
</tr>
<tr>
<td>'contains-whales', 1</td>
<td>melvil : austen</td>
<td>345.4: 1.0</td>
</tr>
<tr>
<td>'contains-sperm', 1</td>
<td>melvil : austen</td>
<td>318.7: 1.0</td>
</tr>
<tr>
<td>'contains-deck', 1</td>
<td>melvil : austen</td>
<td>271.5: 1.0</td>
</tr>
<tr>
<td>'contains-boats', 1</td>
<td>melvil : austen</td>
<td>195.9: 1.0</td>
</tr>
<tr>
<td>'contains-crew', 1</td>
<td>melvil : austen</td>
<td>195.9: 1.0</td>
</tr>
<tr>
<td>'contains-mast', 1</td>
<td>melvil : austen</td>
<td>175.5: 1.0</td>
</tr>
<tr>
<td>'contains-whaling', 1</td>
<td>melvil : austen</td>
<td>175.5: 1.0</td>
</tr>
<tr>
<td>('contains-`', 1)</td>
<td>austen : melvil</td>
<td>166.5: 1.0</td>
</tr>
<tr>
<td>'contains-thee', 1</td>
<td>melvil : austen</td>
<td>162.9: 1.0</td>
</tr>
<tr>
<td>('contains-ll', 1)</td>
<td>melvil : austen</td>
<td>142.4: 1.0</td>
</tr>
<tr>
<td>'contains-sail', 1</td>
<td>melvil : austen</td>
<td>137.7: 1.0</td>
</tr>
<tr>
<td>'contains-voyage', 1</td>
<td>melvil : austen</td>
<td>137.7: 1.0</td>
</tr>
<tr>
<td>'contains-flask', 1</td>
<td>melvil : austen</td>
<td>134.5: 1.0</td>
</tr>
<tr>
<td>'contains-ships', 1</td>
<td>melvil : austen</td>
<td>125.1: 1.0</td>
</tr>
<tr>
<td>'contains-leviathan', 1</td>
<td>melvil : austen</td>
<td>125.1: 1.0</td>
</tr>
<tr>
<td>'contains-cabin', 1</td>
<td>melvil : austen</td>
<td>118.8: 1.0</td>
</tr>
</tbody>
</table>
He, she, very

```python
>>> whosaid.classify(gen_feats('He knows the truth'.split()))
melville
>>> whosaid.prob_classify(gen_feats('He knows the truth'.split())).prob('austen')
0.44921141639835876
>>> whosaid.prob_classify(gen_feats('She knows the truth'.split())).prob('austen')
0.9314339848201395
>>> whosaid.feature_weights('contains-he', 1)
{'melville': 0.1554651574106827, 'austen': 0.16881462610520007}
>>> whosaid.feature_weights('contains-she', 1)
{'melville': 0.011496285815351963, 'austen': 0.2079274689045407}
>>> whosaid.feature_weights('contains-very', 1)
{'melville': 0.0321306449711119, 'austen': 0.13899295669114342}
```
Austen vs. *whale*

- Can a sentence with 'whale' ever be classified as 'austen'?

```python
>>> whosaid.prob_classify(gen_feats('the whale'.split())).prob('austen')
0.00032509324756815693
>>> whosaid.prob_classify(gen_feats('it was a whale'.split())).prob('austen')
0.0009813590453571785
>>> whosaid.prob_classify(gen_feats('it was a beautiful whale'.split())).prob('austen')
0.0034008961166336333
>>> whosaid.prob_classify(gen_feats('she married a whale'.split())).prob('austen')
0.10371709682345985
>>> whosaid.prob_classify(gen_feats('she married a beautiful whale'.split())).prob('austen')
0.28673216572155275
>>> whosaid.prob_classify(gen_feats('she married a very beautiful whale'.split())).prob('austen')
0.6349019382913935
```
Common evaluation setups

- **Training vs. testing** partitions
  1. Training data ← classifier is trained on this section
  2. Testing data ← classifier's performance is measured

- **Training, testing, + development-testing**
  + 3. Development testing data
  ← In feature engineering, researcher can error-analyze the data to improve performance
Cross validation

- But what if our training/testing split is somehow biased?
  - We could randomize
  - Or, use cross-validation.

- *n-fold cross validation method*
  - Partition the data set into equally sized *n* sets
  - Conduct *n* rounds of training-testing, each using 1 partition as testing and the rest *n-1* partitions for training
  - And then take an average of the *n* accuracy figures

<- More reliable accuracy score. Performance evaluation is less dependent on a particular training-testing split

<- We can see how widely performance varies across different training sets
Accuracy as a measure

- **Accuracy**: of all labeling decisions that a classifier made, how many of them are correct?
  - POS tagger
  - Name gender identifier
  - whosaid: Austen/Melville author classifier
  - Document topic identifier
  - Movie review classifier: positive/neg. ("sentiment classifier")
Accuracy as a measure

- **Accuracy**: of all labeling decisions that a classifier made, how many of them are *correct*?

- Interpreting accuracy numbers
  - A movie review sentiment classifier tests 90% accurate. Is this good or bad?
    - What if it turns out 80% movie reviews are positive?
    - How about 60%?
  - A document topic identifier tests 60% accurate. Good or bad?
    - What if 55% of documents are on "Politics"?
    - What if there are as many as 20 different topics, and the largest category only accounts for 10% of the data?

← These questions cannot be answered without considering base probabilities *(priors)*.
Base probabilities

- **Base probabilities (priors)**
  - The probability of a randomly drawn sample to have a label $x$
    - whosaid: 'melville' has a higher prior than 'austen'
    - POS tagger: 'Noun' may have the highest prior than other tags
    - Disease test: 'Negative' is typically much higher than 'Positive'

- **Base rate neglect**
  - A cognitive bias humans have
  - We tend to assume that base probabilities are equal

- **Base performance**
  - The "bottom line" for system performances
    = the highest base probability

  ex. POS tagger: if 20% of all words are 'Noun', then the worst-performing system can be constructed which blindly assigns 'Noun' to every word, whose accuracy is 20%.
When accuracy isn't a good measure

- A medical test for a disease is 96% accurate. Good or bad?
  - What if 95% of population is free of the disease?
- A grammatical error detector is 96% accurate. Good or bad?
  - Suppose 95% of all sentences are error-free.
    - Accuracy alone doesn't tell the whole story.
- We are interested in:
  - Of all "ungrammatical" flags the system raises, what % is correct?
    - This is the precision rate.
  - Of all actual ungrammatical sentences, what % does the system correctly capture as such?
    - This is the recall rate.
A grammatical error detector as a diagnostic test

- Positive: has grammatical error
- Negative: is error-free

<table>
<thead>
<tr>
<th>Test</th>
<th>Real</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Has grammatical error</td>
</tr>
<tr>
<td>positive</td>
<td><strong>True positives</strong></td>
</tr>
<tr>
<td>negative</td>
<td>False negatives</td>
</tr>
</tbody>
</table>

**Accuracy:**

\[
\frac{(Tp + Tn)}{(Tp + Tn + Fp + Fn)}
\]

When the data is predominantly error-free (high base rate), this is not a meaningful measure of system performance.
Outcome of a diagnostic test

- A grammatical error detector as a diagnostic test
  - Positive: has grammatical error
  - Negative: is error-free

<table>
<thead>
<tr>
<th>Test</th>
<th>Real</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Has grammatical error</td>
<td>Is error-free</td>
</tr>
<tr>
<td>positive</td>
<td>True positives</td>
<td>False positives</td>
</tr>
<tr>
<td>negative</td>
<td>False negatives</td>
<td>True negatives</td>
</tr>
</tbody>
</table>

- **Precision:**
  Rate of "True positives" out of all positive rulings
  \[ \text{Precision} = \frac{\text{Tp}}{\text{Tp} + \text{Fp}} \]
Outcome of a diagnostic test

- **A grammatical error detector as a diagnostic test**
  - Positive: has grammatical error
  - Negative: is error-free

<table>
<thead>
<tr>
<th>Test</th>
<th>Real</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Has grammatical error</td>
</tr>
<tr>
<td>positive</td>
<td><strong>2 True positives</strong></td>
</tr>
<tr>
<td>negative</td>
<td>False negatives</td>
</tr>
</tbody>
</table>

- **Recall:**
  Rate of "True positives" out of all actual positive cases (2)
  \[ \text{Recall} = \frac{\text{Tp}}{\text{Tp} + \text{Fn}} \]
Precision vs. recall

- **Precision and recall** are in a **trade-off relationship**.
  - Highly precise grammatical error detector:
    - Ignores many lower-confidence cases $\rightarrow$ drop in recall
  - High recall (captures as many errors as possible):
    - many non-errors will also be flagged $\rightarrow$ drop in precision

- In developing a real-world application, picking the right trade-off point between the two is an important usability issue.
  - A **grammar checker** for general audience (MS-Word, etc)
    - Higher precision or higher recall?
  - Same, but for English learners.
    - Higher precision or higher recall?
F-measure

- Precision and recall are in a trade-off relationship.
  - Both measures should be taken into consideration when evaluating performance

- F-measure
  - Also called F-score, $F_1$ score
  - An overall measure of a test's accuracy:
    Combines precision ($P$) and recall ($R$) into a single measure
  - Harmonic mean
  - Best value: 1, worst value: 0
  - $F_1 = \frac{2PR}{P + R}$
    - = average if $P=R$,
    - < average if $P$ and $R$ different
Wrapping up

- HW 10 final version due on Thursday
  - Need help? Office hours on Wed, 1-3pm (Reed), 3-4pm (Na-Rae)

- Next class:
  - Machine translation
  - Read Ch.21 "Machine Translation" from Jurafsky & Martin's (2008) *Speech and Language Processing*
    - 2nd edition book home page
    - 3rd edition on the way! Many draft chapters are available for preview here.