Natural Language Processing
Text classification

Sofia Serrano
sofias6@cs.washington.edu

Credit to Yulia Tsvetkov and Noah Smith for slides
Announcements

- Academic Integrity Form is due back on Canvas today
- Quiz 1: Wednesday (1/18)
  - 6 multiple-choice questions
  - Released on Canvas once lecture ends on 1/18, open for 12 hours
  - 10-min time limit once you start the quiz
  - Materials from weeks 1 and 2 (anything we talk about up through the end of class today)
    - Introduction to NLP, introduction to text classification
    - Instructions for HW 1
- No class on Monday (MLK Day)

https://courses.cs.washington.edu/courses/cse447/23wi/
We’ll consider alternative models for classification

- Supervised text classification
  - Rule-based
  - Probabilistic
    - Generative models
      - Naïve Bayes
    - Discriminative models
      - Linear models
        - Multinomial logistic regression (aka MaxEnt)
      - Non-linear models
        - Multilayer perceptron
Generative and discriminative models

- **Generative model**: a model that calculates the probability of the input data itself
  \[ P(X, Y) \] 
  joint

- **Discriminative model**: a model that calculates the probability of a latent trait given the data
  \[ P(Y | X) \] 
  conditional
Generative and discriminative models

- **Generative text classification:** Learn a model of the joint $P(X, y)$, and find

  $$\hat{y} = \arg\max_{\tilde{y}} P(X, \tilde{y})$$

- **Discriminative text classification:** Learn a model of the conditional $P(y \mid X)$, and find

  $$\hat{y} = \arg\max_{\tilde{y}} P(\tilde{y} \mid X)$$
We’ll consider alternative models for classification

- Supervised text classification
  - Rule-based
  - Probabilistic
    - Generative models
    - Discriminative models
      - Naïve Bayes
      - Linear models
        - Multinomial logistic regression (aka MaxEnt)
      - Non-linear models
        - Multilayer perceptron
Generative text classification: naïve Bayes

- Simple (naïve) classification method
  - based on the Bayes rule
- Relies on very simple representation of a documents
  - bag-of-words, no relative order
- A good baseline for more sophisticated models

Naïve Bayes

Sentiment analysis: movie reviews

- Given a document $d$ (e.g., a movie review)
- Decide which class $c$ it belongs to: positive, negative, neutral
- Compute $P(c \mid d)$ for each $c$
  - $P(\text{positive} \mid d), P(\text{negative} \mid d), P(\text{neutral} \mid d)$
  - Select the one with max $P$
Naïve Bayes

- Given a document $d$ and a class $c$, Bayes’ rule:

$$P(c|d) = \frac{P(d|c)P(c)}{P(d)}$$
Naïve Bayes

- Given a document $d$ and a class $c$, Bayes’ rule:

$$ P(c|d) = \frac{P(d|c)P(c)}{P(d)} $$

$$ P(‘positive’|d) \propto P(d|‘positive’)P(‘positive’) $$

- likelihood
- prior
Naïve Bayes

- Given a document $d$ and a class $c$, Bayes’ rule:

$$P(c|d) = \frac{P(d|c)P(c)}{P(d)}$$

$$P(\text{‘positive’}|d) \propto P(d|\text{‘positive’})P(\text{‘positive’})$$
Naïve Bayes

- Given a document $d$ and a class $c$, Bayes’ rule:

$$P(c|d) = \frac{P(d|c)P(c)}{P(d)}$$

- $P(\text{‘positive’}|d) \propto P(d|\text{‘positive’})P(\text{‘positive’})$

likelihood
Naïve Bayes independence assumptions

\[ P(w_1, w_2, \ldots, w_n | c) \]

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

\[ P(w_1, w_2, \ldots, w_n | c) = P(w_1 | c) \times P(w_2 | c) \times P(w_3 | c) \times \ldots \times P(w_n | c) \]
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!
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!

\[ P(d|c) = P(w_1, w_2, \ldots, w_n|c) = \prod_i P(w_i|c) \]
Bag-of-Words (BOW)

- Given a document $d$ (e.g., a movie review) – how to represent $d$?
Generative text classification: Naïve Bayes

\[ C_{NB} = \arg \max_c P(c|d) = \arg \max_c \frac{P(d|c)P(c)}{P(d)} \propto \]
\[ \arg \max_c P(d|c)P(c) = \]
\[ \arg \max_c P(w_1, w_2, \ldots, w_n | c)P(c) = \]
\[ \arg \max_{c_j} P(c_j) \prod_i P(w_i | c) \]
Underflow prevention: log space

- Multiplying lots of probabilities can result in floating-point underflow
- 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

\[
C_{NB} = \arg\max_{c_j} P(c_j) \prod_i P(w_i|c)
\]

\[
C_{NB} = \arg\max_{c_j} \log(P(c_j)) + \sum_i \log(P(w_i|c))
\]

- Model is now just max of sum of weights
Learning the multinomial naïve Bayes

- How do we learn (train) the NB model?
Learning the multinomial naïve Bayes

- How do we learn (train) the NB model?
- We learn \( P(c) \) and \( P(w_i|c) \) from training (labeled) data

\[
C_{NB} = \arg\max_{c_j} \log(P(c_j)) + \sum_i \log(P(w_i|c))
\]
Parameter estimation

- Parameter estimation during training
- Concatenate all documents with category \( c \) into one mega-document
- Use the frequency of \( w_i \) in the mega-document to estimate the word probability

\[
C_{NB} = \arg \max_{c_j} \log(P(c_j)) + \sum_i \log(P(w_i|c))
\]

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

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

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

- fraction of times word \( w_i \) appears among all words in documents of topic \( c_j \)

- Create mega-document for topic \( j \) by concatenating all docs in this topic
  - Use frequency of \( w \) in mega-document
Problem with Maximum Likelihood

- What if we have seen no training documents with the word “fantastic” and classified in the topic positive?
Problem with Maximum Likelihood

- What if we have seen no training documents with the word “fantastic” and classified in the topic **positive**?

\[
\hat{P}(\text{“fantastic”} | c = \text{positive}) = \frac{\text{count(“fantastic”, positive)}}{\sum_{w \in V} \text{count}(w, \text{positive})} = 0
\]

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

\[
\arg\max_{c_j} P(c_j) \prod_i P(w_i | c)
\]
Laplace (add-1) smoothing for naïve Bayes

\[
\hat{P}(w_i | c_j) = \frac{\text{count}(w_i, c_j) + 1}{\sum_{w \in V}(\text{count}(w, c_j) + 1)}
\]
Laplace (add-1) smoothing for naïve Bayes

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

\[
= \frac{\text{count}(w_i, c_j) + 1}{(\sum_{w \in V}(\text{count}(w, c_j)) + |V|}
\]
Multinomial naïve Bayes: learning

- From training corpus, extract *Vocabulary*
- Calculate $P(c_j)$ terms
  - For each $c_j$ do
    - $\text{docs}_j \leftarrow$ all docs with class = $c_j$
    - $P(c_j) \leftarrow \frac{|\text{docs}_j|}{\text{total # documents}}$
Multinomial naïve Bayes: learning

- From training corpus, extract Vocabulary
- Calculate $P(c_j)$ terms
  - For each $c_j$ do
    - $\text{docs}_j \leftarrow$ all docs with class $= c_j$
    - $P(c_j) \leftarrow \frac{|\text{docs}_j|}{\text{total \# documents}}$
- Calculate $P(w_i | c_j)$ terms
  - $\text{Text}_j \leftarrow$ single doc containing all $\text{docs}_j$
  - For each word $w_i$ in Vocabulary
    - $n_i \leftarrow$ # of occurrences of $w_i$ in $\text{Text}_j$
    - $P(w_i | c_j) \leftarrow \frac{n_i + \alpha}{n + \alpha|\text{Vocabulary}|}$
## Example

<table>
<thead>
<tr>
<th>Doc</th>
<th>Words</th>
<th>Class</th>
</tr>
</thead>
<tbody>
<tr>
<td>Training</td>
<td></td>
<td></td>
</tr>
<tr>
<td>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</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>Chinese Chinese Chinese Tokyo Japan</td>
<td>?</td>
</tr>
</tbody>
</table>
Example

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

Priors:

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

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

<table>
<thead>
<tr>
<th>Doc</th>
<th>Words</th>
<th>Class</th>
</tr>
</thead>
<tbody>
<tr>
<td>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>5</td>
<td>Chinese Chinese Chinese Tokyo Japan</td>
<td>?</td>
</tr>
</tbody>
</table>
Example

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

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

Priors:

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

Conditional Probabilities:

- \( P(\text{Chinese} \mid c) = \frac{5+1}{8+6} = \frac{6}{14} = \frac{3}{7} \)
- \( P(\text{Tokyo} \mid c) = \frac{0+1}{8+6} = \frac{1}{14} \)
- \( P(\text{Japan} \mid c) = \frac{0+1}{8+6} = \frac{1}{14} \)
- \( P(\text{Chinese} \mid j) = \frac{1+1}{3+6} = \frac{2}{9} \)
- \( P(\text{Tokyo} \mid j) = \frac{1+1}{3+6} = \frac{2}{9} \)
- \( P(\text{Japan} \mid j) = \frac{1+1}{3+6} = \frac{2}{9} \)
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:
\[
\begin{align*}
P(\text{Chinese} \mid c) &= \frac{(5+1)}{(8+6)} = \frac{6}{14} = \frac{3}{7} \\
P(\text{Tokyo} \mid c) &= \frac{(0+1)}{(8+6)} = \frac{1}{14} \\
P(\text{Japan} \mid c) &= \frac{(0+1)}{(8+6)} = \frac{1}{14} \\
P(\text{Chinese} \mid j) &= \frac{(1+1)}{(3+6)} = \frac{2}{9} \\
P(\text{Tokyo} \mid j) &= \frac{(1+1)}{(3+6)} = \frac{2}{9} \\
P(\text{Japan} \mid j) &= \frac{(1+1)}{(3+6)} = \frac{2}{9}
\end{align*}
\]

Choosing a class:
\[
\begin{align*}
P(c \mid d5) &\propto \frac{3}{4} \times (\frac{3}{7})^3 \times \frac{1}{14} \times \frac{1}{14} \\
&\approx 0.0003 \\
P(j \mid d5) &\propto \frac{1}{4} \times (\frac{2}{9})^3 \times \frac{2}{9} \times \frac{2}{9} \\
&\approx 0.0001
\end{align*}
\]
Summary: naïve Bayes is not so naïve

- Naïve Bayes is a probabilistic model
- Naïve because it assumes features are independent of each other for a class
- 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, then it is the Bayes Optimal Classifier for problem
- A good dependable baseline for text classification
  - But we will see other classifiers that give better accuracy
How do we evaluate our function $f$?
Classification evaluation

- **Contingency table**: model’s predictions are compared to the correct results
  - a.k.a. confusion matrix

<table>
<thead>
<tr>
<th></th>
<th>actual pos</th>
<th>actual neg</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>predicted pos</strong></td>
<td>true positive (tp)</td>
<td>false positive (fp)</td>
</tr>
<tr>
<td><strong>predicted neg</strong></td>
<td>false negative (fn)</td>
<td>true negative (tn)</td>
</tr>
</tbody>
</table>

- tn: true negative (actual neg and predicted neg)
- fp: false positive (actual pos and predicted neg)
- tp: true positive (actual pos and predicted pos)
- fn: false negative (actual neg and predicted pos)
Classification evaluation

- Borrowing from Information Retrieval, empirical NLP systems are usually evaluated using the notions of precision and recall
Classification evaluation

- **Precision (P)** is the proportion of the selected items that the system got right in the case of text categorization
  - it is the % of documents classified as “positive” by the system which are indeed “positive” documents
- Reported per class or average

\[
\text{precision} = \frac{\text{true positives}}{\text{true positives} + \text{false positives}} = \frac{tp}{tp + fp}
\]
Classification evaluation

- **Recall (R)** is the proportion of actual items that the system selected in the case of text categorization
  - it is the % of the “positive” documents which were actually classified as “positive” by the system
- Reported per class or average

\[
\text{recall} = \frac{\text{true positives}}{\text{true positives} + \text{false negatives}} = \frac{tp}{tp + fn}
\]
Classification evaluation

- We often want to trade-off precision and recall
  - typically: the higher the precision the lower the recall
  - can be plotted in a precision-recall curve
- It is convenient to combine P and R into a single measure
  - one possible way to do that is F measure

\[
F_\beta = \frac{(\beta^2+1)PR}{\beta^2P+R} \quad \text{for } \beta=1, \quad F_1 = \frac{2PR}{P+R}
\]
Classification evaluation

- Additional measures of performance: accuracy and error
  - accuracy is the proportion of items the system got right
  - error is its complement

\[
\text{accuracy} = \frac{tp + tn}{tp + fp + tn + fn}
\]
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.
  - “Weighted macro-average” weights this average by the true number of examples of each class

- **Microaveraging**
  - Collect decisions for all classes in terms of
    - True Positives
    - False Positives
  - Compute metric ONCE using that table

<table>
<thead>
<tr>
<th>Class</th>
<th>True Positives</th>
<th>False Positives</th>
</tr>
</thead>
<tbody>
<tr>
<td>Class X</td>
<td>4</td>
<td>2</td>
</tr>
<tr>
<td>Class Y</td>
<td>10</td>
<td>7</td>
</tr>
<tr>
<td>Class Z</td>
<td>9</td>
<td>3</td>
</tr>
</tbody>
</table>
Classification common practices

- Divide the training data into $k$ folds (e.g., $k=10$)
- Repeat $k$ times: train on $k-1$ folds and test on the holdout fold, cyclically
- Average over the $k$ folds’ results
K-fold cross-validation
K-fold 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**
  - Handles sampling errors from different datasets
  - Pool results over each split
  - Compute pooled dev set performance
Next class

- Supervised text classification

  - Rule-based
  - Probabilistic

    - Generative models
    - Discriminative models

      - Naïve Bayes
      - Linear models
        - Multinomial logistic regression (aka MaxEnt)
      - Non-linear models
        - Multilayer perceptron