[Text Processing]
Sentiment Analysis: Naive Bayes

Loïc Barrault

l.barrault@sheffield.ac.uk
Department of Computer Science

Text Processing: Overview

- Introduction [RG]
- Text encoding [RG]
- Information Retrieval [RG]
- Text Compression [RG]

Text Processing: Overview

- Introduction [RG]
- Text encoding [RG]
- Information Retrieval [RG]
- Text Compression [RG]

- Sentiment Analysis [LB]
- Information Extraction [LB]
- Introduction to Deep Learning for Text Processing [LB]

[Sentiment Analysis]
Corpus-based / Machine Learning

Sentiment Analysis: 2 main approaches

- Lexicon based
  - Binary
  - Gradable
- **Corpus based**
  - **Naive Bayes**
  - Deep Learning

Bayes classifier

Principle

Assign the **sentiment** or **class** having the highest **posterior probability**.
Namely, determine the sentiment $\mathbf{s}^*$ of text T such that:

$$\mathbf{s}^* = \underset{s_i}{\text{argmax}}\, p(s_i|T) \text{ for } s_i \in \{\textbf{negative}, \textbf{positive}, \textbf{neutral}\}$$

$p(s_i|T)$ cannot be directly estimated correctly $\rightarrow$ use the Bayes rule:

$$p(s_i|T) = \frac{p(T|s_i)p(s_i)}{p(T)}$$

Bayes classifier

Bayes rule:

$$\mathbf{s}^* = \underset{s_i}{\operatorname{argmax}} \frac{p(T|s_i)p(s_i)}{p(T)}$$

Since **evidence** $p(T)$ is independent of s_i, we can ignore it

$$\mathbf{s}^* = \underset{s_i}{\operatorname{argmax}}\, p(T|s_i)p(s_i)$$

- $p(s_i|T)$ is the **posterior probability**
- $p(T|s_i)$ is the **likelihood**
- $p(s_i)$ is the **prior probability**

Naive Bayes classifier

How to compute the likelihood?

Assume that T is described by a number of **features** or attributes $t_1, t_2, \ldots, t_N$

Naive assumption: **features** are **independent**

$$p(T|s_i) = p(t_1, t_2, \ldots, t_N|s_i) \approx \prod_{j=1}^{N} p(t_j|s_i)$$

$\Rightarrow$ product of probabilities of each **feature** value of text occurring with class s_i

Naive Bayes classifier

How to compute the prior probability?

$\rightarrow$ corresponds to the safest decision when no other information is given $\sim$ majority voting

Requires an **annotated corpus** (text along with their sentiment)

Compute **prior probability** by simple relative frequency

$$p(s_i) = \frac{count(s_i)}{\sum_{j=0}^{J} count(s_j)}$$

with J the number of different classes and $count(.)$ is the counting function

Digression: corpus based machine learning

Corpora:

- **training** set $\rightarrow$ used to **estimate probabilities**
  - input data along with the ground truth (correct labels)
- **development** set, also called **validation** set $\rightarrow$ used to **design** the model
  - e.g. feature selection, set meta-parameters (could be some weights)
  - ground truth (correct labels) available
  - used to select the best model
- **test** set $\rightarrow$ used to **evaluate generalisation** power
  - unseen examples
  - best case: no access to the ground truth

Naive Bayes classifier

Final decision

$$\mathbf{s}^* = \underset{s_i}{\operatorname{argmax}}\, p(s_i) \prod_{j=1}^{N} p(t_j|s_i)$$

1. Compute prior probability of each class
2. For each class:
   - Compute likelihood of each feature
3. Calculate the posterior probability by product of previous components
4. Select sentiment having maximum posterior probability

$\rightarrow$ **negative**, **positive** or **neutral**

Naive Bayes for Sentiment Analysis - Example

Consider the following dummy **training** corpus of 7 movie reviews:

Doc	Words	Class
1	great movie, excellent plot, renowned actors	**positive**
2	I had not seen a fantastic plot like this in good 5 years. amazing!!!	**positive**
3	lovely plot, amazing cast, somehow I am in love with the bad guy	**positive**
4	bad movie with great cast, but very poor plot and unimaginative ending	**negative**
5	I hate this film, it has nothing original	**negative**
6	great movie, but not...	**negative**
7	very bad movie, I have no words to express how I dislike it	**negative**

Naive Bayes for Sentiment Analysis - Example

Compute prior probability of each class by relative frequency

$$p(\textbf{positive}) = \frac{count(\textbf{positive})}{\sum_{s \in \{\textbf{positive},\textbf{negative}\}} count(s)} = \frac{3}{7} = 0.43$$

$$p(\textbf{negative}) = \frac{count(\textbf{negative})}{\sum_{s \in \{\textbf{positive},\textbf{negative}\}} count(s)} = \frac{4}{7} = 0.57$$

Naive Bayes for Sentiment Analysis - Example

What **features** should we consider?

- could use **all** words
  - but some might not be relevant $\rightarrow$ we are interested in the **emotion words**
  - use the **development** corpus to decide!
- in this example: focus on **adjectives** (**bag-of-word** representation)

Doc	Words	Class
1	**great** movie, **excellent** plot, **renowned** actors	**positive**
2	I had not seen a **fantastic** plot like this in **good** 5 years. **amazing !!!**	**positive**
3	**Lovely** plot, **amazing** cast, somehow I am in love with the **bad** guy	**positive**
4	**bad** movie with **great** cast, but very **poor** plot and **unimaginative** ending	**negative**
5	I hate this film, it has nothing **original**	**negative**
6	**great** movie, but **bad** casting...	**negative**
7	Very **bad** movie, I have no words to express how I dislike it	**negative**

Naive Bayes for Sentiment Analysis - Example

What **features** should we consider?

- could use **all** words
  - but some might not be relevant → we are interested in the **emotion words**
  - use the **development** corpus to decide!
- in this example: focus on **adjectives** (**bag-of-word** representation)

Doc	Words	Class
1	**great excellent renowned**	**positive**
2	**fantastic good amazing !!!**	**positive**
3	**lovely amazing bad**	**positive**
4	**bad great poor unimaginative**	**negative**
5	**original**	**negative**
6	**great bad**	**negative**
7	**bad**	**negative**

Naive Bayes for Sentiment Analysis - Example

Compute the likelihoods for all features and given each class

Important

Assume standard pre-processing: tokenisation, lowercasing, punctuation removal (but keep special punctuation, e.g. "!!!")

Examples:

- GOOD = GooD = Good = good
- I'll = I will (though not relevant here)
- aren't = are not

Naive Bayes for Sentiment Analysis - Example

Compute the likelihoods for all features and given each class

$$p(t_j|s_i) = \frac{count(t_j, s_i)}{\sum_{j,k} count(t_j, s_k)} \qquad \rightarrow \textbf{relative frequency}$$

p(amazing|**positive**)	= 2/10	p(amazing|**negative**)	= 0/8
p(bad|**positive**)	= 1/10	p(bad|**negative**)	= 3/8
p(excellent|**positive**)	= 1/10	p(excellent|**negative**)	= 0/8
p(fantastic|**positive**)	= 1/10	p(fantastic|**negative**)	= 0/8
p(good|**positive**)	= 1/10	p(good|**negative**)	= 0/8
p(great|**positive**)	= 1/10	p(great|**negative**)	= 2/8
p(lovely|**positive**)	= 1/10	p(lovely|**negative**)	= 0/8
p(original|**positive**)	= 0/10	p(original|**negative**)	= 1/8
p(poor|**positive**)	= 0/10	p(poor|**negative**)	= 1/8
p(renowned|**positive**)	= 1/10	p(renowned|**negative**)	= 0/8
p(unimaginative|**positive**)	= 0/10	p(unimaginative|**negative**)	= 1/8
p(!!!|**positive**)	= 1/10	p(!!!|**negative**)	= 0/8

Naive Bayes for Sentiment Analysis - Example

Relative frequencies for prior and likelihoods make the model in a Naive Bayes classifier
$\rightarrow$ features are supposed independent (no covariance taken into account)
$\rightarrow$ this is an approximation of course

What is the model?
$\rightarrow$ the set of all prior probabilities and likelihoods

At **test** time, this model is used to find the most likely class (sentiment) for the unknown text

$$\mathbf{s}^* = \underset{s_i}{\operatorname{argmax}}\, p(s_i) \prod_{j=1}^{N} p(t_j|s_i)$$

Naive Bayes for Sentiment Analysis - Text Ex. 1

Consider the following test segment to classify:

Doc	Words	Class
8	This was a fantastic story, great, lovely	?

Naive Bayes for Sentiment Analysis - Text Ex. 1

Consider the following test segment to classify:

1. Extract features

Doc	Words	Class
8	This was a **fantastic** story, **great**, **lovely**	?

Naive Bayes for Sentiment Analysis - Text Ex. 1

Consider the following test segment to classify:

1. Extract features
2. Build representation

Doc	Words	Class
8	**fantastic great lovely**	?

Naive Bayes for Sentiment Analysis - Text Ex. 1

Consider the following test segment to classify:

1. Extract features
2. Build representation

Doc	Words	Class
8	**fantastic great lovely**	?

3. Get likelihoods

$$\begin{aligned} p(\text{fantastic}|\textbf{positive}) &= 1/10 \\ p(\text{great}|\textbf{positive}) &= 1/10 \\ p(\text{lovely}|\textbf{positive}) &= 1/10 \end{aligned} \quad \Bigg| \quad \begin{aligned} p(\text{fantastic}|\textbf{negative}) &= 0/8 \\ p(\text{great}|\textbf{negative}) &= 2/8 \\ p(\text{lovely}|\textbf{negative}) &= 0/8 \end{aligned}$$

Naive Bayes for Sentiment Analysis - Text Ex. 1

$$\begin{aligned} p(\textbf{positive}|text) &= p(\textbf{positive}) * p(fantastic|\textbf{positive}) * p(great|\textbf{positive}) * p(lovely|\textbf{positive}) \\ &= 3/7 * 1/10 * 1/10 * 1/10 \\ &= \mathbf{0.00043} \end{aligned}$$

$$\begin{aligned} p(\textbf{negative}|text) &= p(\textbf{negative}) * p(fantastic|\textbf{negative}) * p(great|\textbf{negative}) * p(lovely|\textbf{negative}) \\ &= 4/7 * 0/8 * 2/8 * 0/8 \\ &= \mathbf{0} \end{aligned}$$

Final decision: sentiment is **positive**

Naive Bayes for Sentiment Analysis - Text Ex. 2

Consider the following test segment to classify:

Doc	Words	Class
9	Great plot, great cast, great everything	?

Naive Bayes for Sentiment Analysis - Text Ex. 2

Consider the following test segment to classify:

1. Extract features
2. Build representation

Doc	Words	Class
9	**Great great great**	?

Naive Bayes for Sentiment Analysis - Text Ex. 2

Consider the following test segment to classify:

1. Extract features
2. Build representation

Doc	Words	Class
9	**Great great great**	?

3. Get likelihoods: p(great|**positive**) $= 1/10$ | p(great|**negative**) $= 2/8$

Naive Bayes for Sentiment Analysis - Text Ex. 2

Consider the following test segment to classify:

1. Extract features
2. Build representation

Doc	Words	Class
9	**Great great great**	?

3. Get likelihoods: p(great|**positive**) $= 1/10$ | p(great|**negative**) $= 2/8$
4. Compute posteriors

$$p(\textbf{positive}|text) = 3/7 * 1/10 * 1/10 * 1/10 = \mathbf{0.00043}$$
$$p(\textbf{negative}|text) = 4/7 * 2/8 * 2/8 * 2/8 = \mathbf{0.00893}$$

Final decision: sentiment is **negative**

Naive Bayes for Sentiment Analysis - Text Ex. 2

Consider the following test segment to classify:

1. Extract features
2. Build representation

Doc	Words	Class
9	**Great great great**	?

3. Get likelihoods: $p(\text{great}|\textbf{positive}) = 1/10$ | p ... e) $= 2/8$
4. Compute posteriors

Training data should be representative!

$$p(\textbf{positive}|text) = 3/7 * 1/10 * 1/10 * 1/10 = \mathbf{0.00043}$$
$$p(\textbf{negative}|text) = 4/7 * 2/8 * 2/8 * 2/8 = \mathbf{0.00893}$$

Final decision: sentiment is **negative**

Naive Bayes for Sentiment Analysis - Text Ex. 3

Consider the following test segment to classify:

Doc	Words	Class
10	Boring movie, annoying plot, unimaginative ending	?

Naive Bayes for Sentiment Analysis - Text Ex. 3

Consider the following test segment to classify:

1. Extract features
2. Build representation

Doc	Words	Class
9	**Boring annoying unimaginative**	?

Naive Bayes for Sentiment Analysis - Text Ex. 3

Consider the following test segment to classify:

1. Extract features
2. Build representation

Doc	Words	Class
9	**Boring annoying unimaginative**	?

3. Get likelihoods:

$p(\text{unimaginative}|\textbf{positive}) \quad = 0/10 \;\Big|\; p(\text{unimaginative}|\textbf{negative}) \quad = 1/8$

Naive Bayes for Sentiment Analysis - Text Ex. 3

Consider the following test segment to classify:

1. Extract features
2. Build representation

Doc	Words	Class
9	**Boring annoying unimaginative**	?

3. Get likelihoods:
   p(unimaginative|**positive**) = 0/10 | p(unimaginative|**negative**) = 1/8
4. Compute posteriors

$$p(\textbf{positive}|text) = 3/7 * 0/10 * 0/10 * 0/10 = \textbf{0.0}$$
$$p(\textbf{negative}|text) = 4/7 * 0/8 * 0/8 * 1/8 = \textbf{0.0}$$

Final decision: sentiment is **???**

Naive Bayes for Sentiment Analysis - Text Ex. 3

Consider the following test segment to classify:

1. Extract features
2. Build representation

Doc	Words	Class
9	**Boring annoying unimaginative**	?

Training data should be large!

3. Get likelihoods:

p(unimaginative|**positive**) $= 0/10$ | p(unimaginative|**negative**) $= 1/8$

4. Compute posteriors

$$p(\textbf{positive}|text) = 3/7 * 0/10 * 0/10 * 0/10 = \mathbf{0.0}$$
$$p(\textbf{negative}|text) = 4/7 * 0/8 * 0/8 * 1/8 = \mathbf{0.0}$$

Final decision: sentiment is **???**

Naive Bayes for Sentiment Analysis

Cannot ensure that all possible word appear in the training corpus
$\rightarrow$ apply a **smoothing** technique

$$p(t_j|s_i) = \frac{count(t_j, s_i) + 1}{count(s_i) + |V|}$$

$\rightarrow$ **Laplace** smoothing, also called **add-1** smoothing

where $|V|$ is the number of distinct features (also called **vocabulary**) $\rightarrow$ 12 in our example

Naive Bayes for Sentiment Analysis - Text Ex. 3

Consider the following test segment to classify:

1. Extract features
2. Build representation

Doc	Words	Class
9	**Boring annoying unimaginative**	?

Naive Bayes for Sentiment Analysis - Text Ex. 3

Consider the following test segment to classify:

1. Extract features
2. Build representation

Doc	Words	Class
9	**Boring annoying unimaginative**	?

3. **Get new likelihoods:**

$$
\begin{aligned}
p(\text{boring}|\textbf{positive}) &= (0+1)/(10+12) = 1/22 \\
p(\text{annoying}|\textbf{positive}) &= (0+1)/(10+12) = 1/22 \\
p(\text{unimaginative}|\textbf{positive}) &= (0+1)/(10+12) = 1/22
\end{aligned}
$$

$$
\begin{aligned}
p(\text{boring}|\textbf{negative}) &= (0+1)/(8+12) = 1/20 \\
p(\text{annoying}|\textbf{negative}) &= (0+1)/(8+12) = 1/20 \\
p(\text{unimaginative}|\textbf{negative}) &= (1+1)/(8+12) = 2/20
\end{aligned}
$$

Naive Bayes for Sentiment Analysis - Text Ex. 3

Consider the following test segment to classify:

1. Extract features
2. Build representation

Doc	Words	Class
9	**Boring annoying unimaginative**	?

3. **Get new likelihoods:**

$$\begin{aligned} p(\text{boring}|\textbf{positive}) &= (0+1)/(10+12) = 1/22 \\ p(\text{annoying}|\textbf{positive}) &= (0+1)/(10+12) = 1/22 \\ p(\text{unimaginative}|\textbf{positive}) &= (0+1)/(10+12) = 1/22 \end{aligned}$$

$$\begin{aligned} p(\text{boring}|\textbf{negative}) &= (0+1)/(8+12) = 1/20 \\ p(\text{annoying}|\textbf{negative}) &= (0+1)/(8+12) = 1/20 \\ p(\text{unimaginative}|\textbf{negative}) &= (1+1)/(8+12) = 2/20 \end{aligned}$$

4. Compute posteriors

$$\begin{aligned} p(\textbf{positive}|text) = 3/7 * 1/22 * 1/22 * 1/22 &= \mathbf{0.000040} \\ p(\textbf{negative}|text) = 4/7 * 1/20 * 1/20 * 2/20 &= \mathbf{0.000143} \end{aligned}$$

Final decision: sentiment is **negative**

Naive Bayes for Sentiment Analysis

How can be used a trained classifier?

- Different level of granularity:
- document-level:
  - direct classification
  - by aggregation of sentence classification results
- sentence or phrase-level:
  - select sentences focusing on some aspect (features)
  - $\rightarrow$ provides a specific sentiment analysis for this feature

Naive Bayes for Sentiment Analysis - Questions

Is this a good solution?

- Simple solution. Works well if data is not sparse.

Is it robust?

- Problem: what about new words?

What is the role of the prior?

- reminder: safest decision **when no other information is given** $\sim$ majority voting
- important especially on biased cases

Can we extend to a non-binary classification?

- Naive Bayes can be easily extended by considering more than 2 classes
- ... but beware of the **curse of dimensionality** $\rightarrow$ sparsity

Naive Bayes for Sentiment Analysis - Questions

How can we improve this solution?

1. Consider other **features**?
   - using all words in Naive Bayes works well for some tasks
   - subsets of words may help → use the **development** set to that end
   - previous examples consider only adjectives, this is limitating
     - **verbs**: hate, dislike
     - **intensifiers**: very, much, a lot
     - **negation**: not ← **very important!**
     - **nouns**: love, creativity

   → possibly people tend to mostly talk of those **nouns** in a **positive** or **negative** way
   - pre-built polarity lexicons can be helpful
2. Consider other **algorithms**?
   - Maximum Entropy (MaxEnt), Support Vector Machines (SVM), neural networks

   → no assumption of statistical independence among features

   → more complex but tend to do better

Extra reading

Bing Liu and Lei Zhang (2012).
A survey on opinion mining and sentiment analysis.
Kluwer Academic Publishers:
http://www.cs.uic.edu/~lzhang3/paper/opinion_survey.pdf

Bing Liu (2012).
Sentiment Analysis and Opinion Mining.
Morgan and Claypool Publishers. Draft on line at:
https://www.cs.uic.edu/~liub/FBS/SentimentAnalysis-and-OpinionMining.pdf