Naive Bayes and text classification

ÉCOLE D'INGÉNIEURS EN INFORMATIQUE

04/03 - Gustave Cortal

Is this spam?

Subject: **Important notice!**
From: Stanford University <newsforum@stanford.edu>
Date: October 28, 2011 12:34:16 PM PDT
To: undisclosed-recipients:;

Greats News!

You can now access the latest news by using the link below to login to Stanford University News Forum.

http://www.123contactform.com/contact-form-StanfordNew1-236335.html

Click on the above link to login for more information about this new exciting forum. You can also copy the above link to your browser bar and login for more information
about the new services.

© Stanford University. All Rights Reserved.

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.

What is the subject of this article?

MEDLINE Article


Brain and Cognition

Syntactic frame and verb bias in aphasia: Plausibility judgments of undergoer-subject sentences

?

MeSH Subject Category Hierarchy

- Antogonists and Inhibitors
- Blood Supply
- Chemistry
- Drug Therapy
- Embryology
- Epidemiology
- …

Text classification

- Assigning subject categories, topics, or genres
- Spam detection
- Authorship identification
- Age identification
- Language identification
- Sentiment analysis
- …

Text classification: definition

- Input:
  - a document $\boldsymbol{d}$
  - a fixed set of classes $\boldsymbol{C = \{c_1, c_2, \ldots, c_J\}}$
- Output: a predicted class $\boldsymbol{c \in C}$

Classification methods: supervised machine learning

- Input:
  - a document **d**
  - a fixed set of classes $\mathbf{C = \{c_1, c_2, \ldots, c_J\}}$
  - A training set of **m** hand-labeled documents $\mathbf{(d_1,c_1),\ldots,(d_m,c_m)}$
- Output:
  - a learned classifier $\boldsymbol{\gamma}\mathbf{:d} \rightarrow \mathbf{c}$

Classification methods: supervised machine learning

- Any kind of classifier
  - Naïve Bayes
  - Logistic Regression
  - Support-Vector Machines
  - k-Nearest Neighbors
  - …

Naive Bayes classifier

Naive Bayes intuition

Simple classification method based on Bayes rule

Relies on very simple representation of document : Bag of Words

The Bag of Words representation

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!

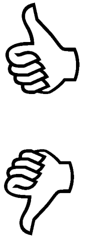
it	6
I	5
the	4
to	3
and	3
seen	2
yet	1
would	1
whimsical	1
times	1
sweet	1
satirical	1
adventure	1
genre	1
fairy	1
humor	1
have	1
great	1
...	...

The Bag of Words representation

$\gamma($

seen	2
sweet	1
whimsical	1
recommend	1
happy	1
...	...

$)=c$



Bayes' rule applied to documents

- For a document **d** and a class **c**

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

Naive Bayes classifier

$$c_{MAP} = \underset{c \in C}{\operatorname{argmax}}\, P(c \mid d)$$

MAP is "maximum a posteriori" = most likely class

$$= \underset{c \in C}{\operatorname{argmax}} \frac{P(d \mid c)P(c)}{P(d)}$$

Bayes Rule

$$= \underset{c \in C}{\operatorname{argmax}}\, P(d \mid c)P(c)$$

Dropping the denominator

Naive Bayes classifier

"Likelihood" "Prior"

$$c_{MAP} = \underset{c \in C}{\operatorname{argmax}} P(d \mid c) P(c)$$

$$= \underset{c \in C}{\operatorname{argmax}} P(x_1, x_2, \ldots, x_n \mid c) P(c)$$

Document *d* represented as features *x1..xn*

Independence assumptions

$$P(x_1, x_2, \ldots, x_n \mid c)$$

Bag of Words assumption: Assume position doesn't matter

Conditional independence: Assume the feature probabilities $P(x_i|c_j)$ are independent given the class *c.*

$$P(x_1, \ldots, x_n \mid c) = P(x_1 \mid c) \bullet P(x_2 \mid c) \bullet P(x_3 \mid c) \bullet \ldots \bullet P(x_n \mid c)$$

Naive Bayes classifier

$$c_{MAP} = \underset{c \in C}{\operatorname{argmax}} P(x_1, x_2, \ldots, x_n \mid c) P(c)$$

$$c_{NB} = \underset{c \in C}{\operatorname{argmax}} P(c_j) \prod_{x \in X} P(x \mid c)$$

Applying Naive Bayes classifiers to text classification

positions $\leftarrow$ all word positions in test document

$$c_{NB} = \underset{c_j \in C}{\operatorname{argmax}} P(c_j) \prod_{i \in positions} P(x_i \mid c_j)$$

Problems with multiplying lots of probs

$$c_{NB} = \underset{c_j \in C}{\operatorname{argmax}} P(c_j) \prod_{i \in positions} P(x_i \mid c_j)$$

Multiplying lots of probabilities can result in floating-point underflow

.0006 * .0007 * .0009 * .01 * .5 * .000008….

Use logs, because log(*ab*) = log(*a*) + log(*b*)

We'll sum logs of probabilities instead of multiplying probabilities

Do everything in log space

Instead of this: $$c_{NB} = \underset{c_j \in C}{\operatorname{argmax}} P(c_j) \prod_{i \in positions} P(x_i \mid c_j)$$

This: $$c_{\text{NB}} = \underset{c_j \in C}{\operatorname{argmax}} \left[\log P(c_j) + \sum_{i \in \text{positions}} \log P(x_i | c_j) \right]$$

Learning the Naive Bayes classifier

Learning the Naive Bayes classifier

Maximum Likelihood Estimate : use the frequencies in the data

$$\hat{P}(c_j) = \frac{N_{c_j}}{N_{total}}$$

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

Parameter estimation

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

fraction of times word w_i appears among all words in documents of class c_j

Create "mega-document" for class *j* by concatenating all docs in this class

- Use frequency of *w* in mega-document

Problem with Maximum Likelihood

- What if we have seen no training documents with the word *fantastic*?

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

$$c_{MAP} = \operatorname{argmax}_c \hat{P}(c) \prod_i \hat{P}(x_i \mid c)$$

Unknown words

What about unknown words that appear in our test data but not in our training data or vocabulary?

→ We remove them from the test document

Laplace (add-1) smoothing for Naive Bayes

$$\hat{P}(w_i \mid c) = \frac{count(w_i, c) + 1}{\sum_{w \in V} \left(count(w, c) + 1 \right)}$$

$$= \frac{count(w_i, c) + 1}{\left(\sum_{w \in V} count(w, c) \right) + |V|}$$

Learning the Naive Bayes classifier

From training corpus, extract **Vocabulary**

Calculate $P(c_j)$ terms

- For each c_j in C do

  $docs_j \leftarrow$ docs with class c_j

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

Calculate $P(w_k \mid c_j)$ terms

- $Mdoc_j \leftarrow$ single doc containing all $docs_j$
- For each word w_k in **Vocabulary**

  $n_k \leftarrow$ # of occurrences of w_k in $Mdoc_j$

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

Stop words

Some systems ignore **stop words**

- Stop words are very frequent words like *the* and *a*.
  - Sort the vocabulary by word frequency in training set
  - Call the top 10 or 50 words the **stopword list**.
  - Remove all stop words from both training and test sets

Naive Bayes for sentiment analysis

Example

	Cat	Documents
Training	-	just plain boring
	-	entirely predictable and lacks energy
	-	no surprises and very few laughs
	+	very powerful
	+	the most fun film of the summer
Test	?	predictable with no fun

Example

	Cat	Documents
Training	-	just plain boring
	-	entirely predictable and lacks energy
	-	no surprises and very few laughs
	+	very powerful
	+	the most fun film of the summer
Test	?	predictable with no fun

1. Calculate priors:

$$\hat{P}(c_j) = \frac{N_{c_j}}{N_{total}}$$

P(-) = 3/5
P(+) = 2/5

2. Drop *with*

3. Calculate likelihoods:

$$p(w_i|c) = \frac{count(w_i, c) + 1}{(\sum_{w \in V} count(w, c)) + |V|}$$

$$P(\text{“predictable”}|-) = \frac{1+1}{14+20} \qquad P(\text{“predictable”}|+) = \frac{0+1}{9+20}$$

$$P(\text{“no”}|-) = \frac{1+1}{14+20} \qquad P(\text{“no”}|+) = \frac{0+1}{9+20}$$

$$P(\text{“fun”}|-) = \frac{0+1}{14+20} \qquad P(\text{“fun”}|+) = \frac{1+1}{9+20}$$

4. Scoring the test set:

$$P(-)P(S|-) = \frac{3}{5} \times \frac{2 \times 2 \times 1}{34^3} = 6.1 \times 10^{-5}$$

$$P(+)P(S|+) = \frac{2}{5} \times \frac{1 \times 1 \times 2}{29^3} = 3.2 \times 10^{-5}$$

Binary Naive Bayes

For tasks like sentiment, word **occurrence** seems to be more important than word **frequency**

Binary Naive Bayes: remove duplicate words in each document

Each class is a unigram language model

- Assigning each word: P(word | c)
- Assigning each sentence: P(s|c)=Π P(word|c)

Class *pos*

0.1 I

0.1 love

0.01this

0.05fun

0.1 film

I	love	this	fun	film
0.1	0.1	.05	0.01	0.1

P(s | pos) = 0.0000005

Naive Bayes as a Language Model

- Which class assigns the higher probability to s?

Model pos	
0.1	I
0.1	love
0.01	this
0.05	fun
0.1	film

Model neg	
0.2	I
0.001	love
0.01	this
0.005	fun
0.1	film

I	love	this	fun	film
0.1	0.1	0.01	0.05	0.1
0.2	0.001	0.01	0.005	0.1

$P(s|pos) > P(s|neg)$

Evaluation (Precision, Recall, F1)

How good is our classifier ? The confusion matrix

		gold standard labels	
		gold positive	gold negative
system output labels	system positive	**true positive**	**false positive**
	system negative	**false negative**	**true negative**

Accuracy

gold standard labels

		gold positive	gold negative
system output labels	system positive	**true positive**	**false positive**
	system negative	**false negative**	**true negative**

$$\textbf{accuracy} = \frac{\text{tp+tn}}{\text{tp+fp+tn+fn}}$$

Problems with accuracy

Accuracy doesn't work well when we're dealing with imbalanced classes

Suppose we want to find EPITA tweets in a corpus of 1 million tweets

- 100 are about EPITA
- 999 900 are tweets about something unrelated

Imagine the following simple classifier: **every tweet is "not about EPITA"**

Problems with accuracy

Accuracy of our "not about EPITA" classifier:

999 900 true negatives and 100 false negatives

Accuracy is 999 900/1 000 000 = **99.99%**

→ but useless at finding EPITA tweets

Precision and Recall

system output labels		*gold standard labels*		
		gold positive	gold negative	
	system positive	**true positive**	**false positive**	**precision** $= \frac{\text{tp}}{\text{tp+fp}}$
	system negative	**false negative**	**true negative**	
		recall $= \frac{\text{tp}}{\text{tp+fn}}$		**accuracy** $= \frac{\text{tp+tn}}{\text{tp+fp+tn+fn}}$

F1 score

- F1 is a combination of precision and recall

$$F_1 = \frac{2PR}{P+R}$$

Evaluation for more than two classes: microaverage

		gold labels		
		urgent	normal	spam
system output	urgent	8	10	1
	normal	5	60	50
	spam	3	30	200

$$\textbf{precision}_{u} = \frac{8}{8+10+1}$$

$$\textbf{precision}_{n} = \frac{60}{5+60+50}$$

$$\textbf{precision}_{s} = \frac{200}{3+30+200}$$

$$\textbf{recall}_{u} = \frac{8}{8+5+3} \quad \textbf{recall}_{n} = \frac{60}{10+60+30} \quad \textbf{recall}_{s} = \frac{200}{1+50+200}$$

Evaluation for more than two classes: macroaverage

Class 1: Urgent

	true urgent	true not
system urgent	8	11
system not	8	340

$$\text{precision} = \frac{8}{8+11} = .42$$

Class 2: Normal

	true normal	true not
system normal	60	55
system not	40	212

$$\text{precision} = \frac{60}{60+55} = .52$$

Class 3: Spam

	true spam	true not
system spam	200	33
system not	51	83

$$\text{precision} = \frac{200}{200+33} = .86$$

Pooled

	true yes	true no
system yes	268	99
system no	99	635

$$\begin{array}{c}\text{microaverage}\\ \text{precision}\end{array} = \frac{268}{268+99} = \mathbf{.73}$$

$$\begin{array}{c}\text{macroaverage}\\ \text{precision}\end{array} = \frac{.42+.52+.86}{3} = \mathbf{.60}$$

Micro or macro average?

Microaverage is dominated by the more frequent class

Macroaverage better reflects the statistics of the smaller classes, and so is more appropriate when performance on all the classes is equally important