Text Classification: Naive Bayes

The Task of Text Classification

Help Hours Today: 4-5:30

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.

Who wrote which Federalist papers?

- Anonymous essays try to convince New York to ratify U.S Constitution written by Jay, Madison, Hamilton.
- Authorship of 12 of the letters in dispute
- Solved by Mosteller and Wallace (1963) using Bayesian methods

Image: Tyler Feder (https://www.allfreepapercrafts.com/Free-Printables/Hamilton-Paper-Dolls)

What is the subject of this research 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

…

Positive or negative movie review?

- \+ *...zany characters and richly applied satire, and some great plot twists*
- – *It was pathetic. The worst part about it was the boxing scenes...*
- \+ *...awesome caramel sauce and sweet toasty almonds. I love this place!*
- – *...awful pizza and ridiculously overpriced...*

Positive or negative movie review?

+ *...zany characters and richly applied satire, and some great plot twists*
– *It was pathetic. The worst part about it was the boxing scenes...*
+ *...awesome caramel sauce and sweet toasty almonds. I love this place!*
– *...awful pizza and ridiculously overpriced...*

Why sentiment analysis?

Movie: is this review positive or negative?

Products: what do people think about the new iPhone?

Public sentiment: how is consumer confidence?

Politics: what do people think about this candidate or issue?

Prediction: predict election outcomes or market trends from sentiment

Scherer Typology of Affective States

Emotion: brief organically synchronized … evaluation of a major event

- *angry, sad, joyful, fearful, ashamed, proud, elated*

Mood: diffuse non-caused low-intensity long-duration change in subjective feeling

- *cheerful, gloomy, irritable, listless, depressed, buoyant*

Interpersonal stances: affective stance toward another person in a specific interaction

- *friendly, flirtatious, distant, cold, warm, supportive, contemptuous*

Attitudes: enduring, affectively colored beliefs, dispositions towards objects or persons

- *liking, loving, hating, valuing, desiring*

Personality traits: stable personality dispositions and typical behavior tendencies

- *nervous, anxious, reckless, morose, hostile, jealous*

Scherer Typology of Affective States

Emotion: brief organically synchronized … evaluation of a major event
- *angry, sad, joyful, fearful, ashamed, proud, elated*

Mood: diffuse non-caused low-intensity long-duration change in subjective feeling
- *cheerful, gloomy, irritable, listless, depressed, buoyant*

Interpersonal stances: affective stance toward another person in a specific interaction
- *friendly, flirtatious, distant, cold, warm, supportive, contemptuous*

Attitudes: enduring, affectively colored beliefs, dispositions towards objects or persons
- *liking, loving, hating, valuing, desiring*

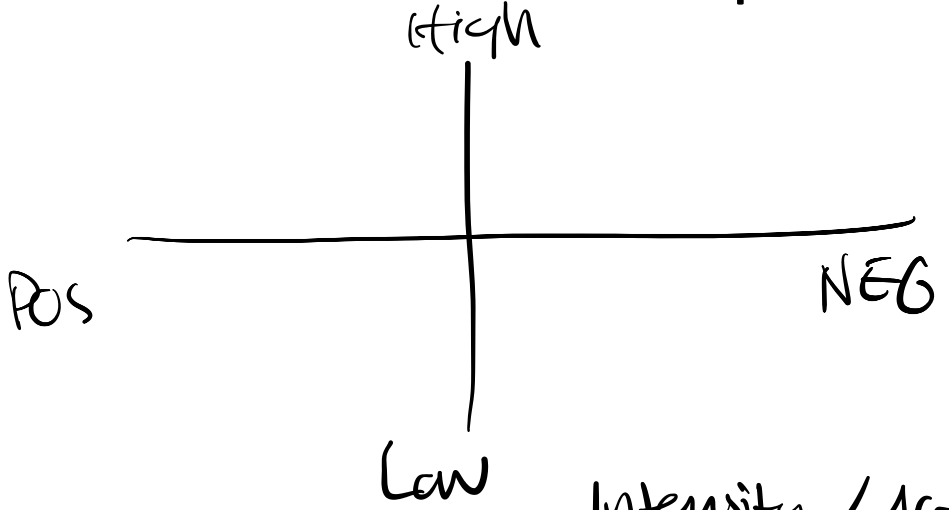
Personality traits: stable personality dispositions and typical behavior tendencies
- *nervous, anxious, reckless, morose, hostile, jealous*

Basic Sentiment Classification

Sentiment analysis is the detection of **attitudes**

Today we will simply ask:

- Is the attitude of this text positive or negative?



Valence / Polarity

Intensity / Arousal

Text Classification

Text Classification: definition

Input:

- a document d (a bit of text)
- 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

- negative: 🙁 OR ("didn't" AND "like")

Accuracy can be high

- If rules carefully refined by expert

But building and maintaining is expensive

- what happens when 🫨 gets added later this year?

Classification Methods: Supervised Machine Learning

Input:

- a document d
- a fixed set of classes $C = \{c_1, c_2,\ldots, c_J\}$
- A training set of m labeled documents $(d_1,c_1),\ldots,(d_m,c_m)$

Output:

- a learned classifier $\gamma: d \rightarrow c$

Classification Methods:
Supervised Machine Learning

There are many kinds of classifiers

- Naïve Bayes
- Logistic regression
- Neural networks
- k-Nearest Neighbors
- …

Text Classification and Naive Bayes

The Naive Bayes Classifier

Naive Bayes Intuition

"Naive" classification method based on Bayes rule:

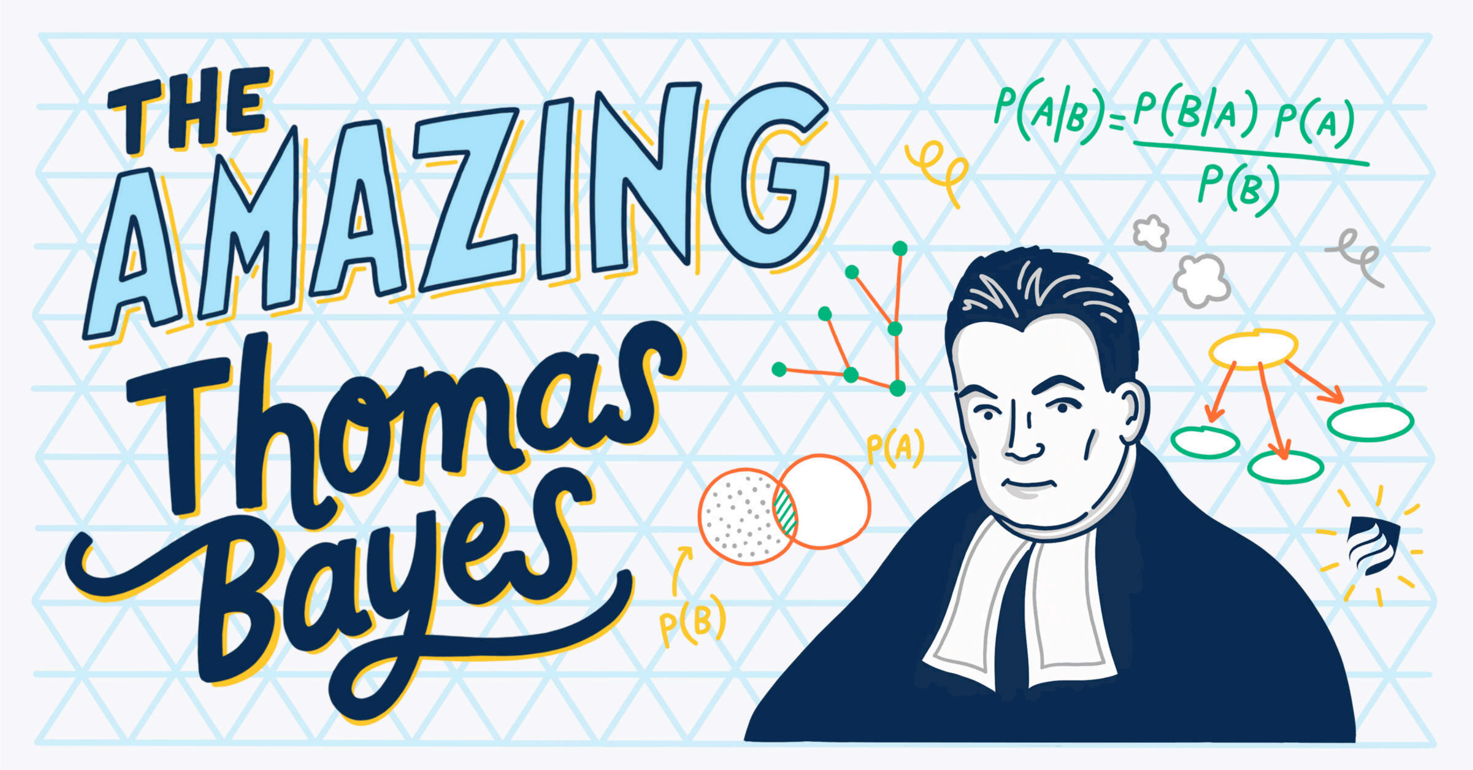


Image: Jim Kulich (https://www.elmhurst.edu/blog/thomas-bayes)

Naive Bayes Intuition

"Naive" classification method based on Bayes rule:

Naive Bayes Intuition

"Naive" classification method based on Bayes rule:

Usually used with a simplified representation of a document called a **bag of words**

Image: Khulood Nasher (https://khuloodnasher.medium.com/bag-of-words-e2969110843)

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
I
the
to
and
seen
yet
would
whimsical
times
sweet
satirical
adventure
genre
fairy
humor
have
great
…

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!

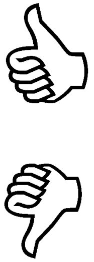
it	6
I	5
the	4
to	3
and	3
seen	2
yet	1
would	1
whimsical	1
times	1
sweet	1
satirical	1
adventure	1
genre	1
fairy	1
humor	1
have	1
great	1
…	…

The bag of words representation

$\gamma($

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

$) = c$



Bayes' Rule Applied to Documents and Classes

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

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

same for all classes

Naive Bayes Classifier

MAP is "maximum a posteriori" = most likely class

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

C = set of classes
d = document

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

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

How likely are we to produce d in class c?

How likely are we to see class c?

Naive Bayes Classifier

MAP is "maximum a posteriori" = most likely class

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

Bayes Rule

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

Dropping the denominator

$$= \operatorname*{argmax}_{c \in C} P(d \mid c)P(c)$$

Naive Bayes Classifier

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

Likelihood Prior

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

Naive Bayes Classifier

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

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

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

Naïve Bayes Classifier

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

$O(|X|^n \bullet |C|)$ parameters

How often does this class occur?

Could only be estimated if a very, very large number of training examples was available.

We can just count the relative frequencies in a corpus

Multinomial Naive Bayes: Independence Assumptions

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

Bag of words assumption — position doesn't matter

Conditional Independence: assume that the feature probabilities $P(x_i \mid c_j)$ are independent given the class.

$$P(x_1 = w_1, x_2 = w_2 \ldots x_n = w_n \mid c) \approx P(x_1 \mid c) \cdot P(x_2 \mid c) \cdot \ldots P(x_n \mid c)$$

Multinomial Naive Bayes: Independence Assumptions

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

Bag of Words assumption: Assume position doesn't matter

Multinomial Naive Bayes: 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.*

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

Multinomial Naive Bayes Classifier

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

Prior

Likelihood

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

Multinomial 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 Multinomial Naive Bayes Classifiers to Text Classification

$\text{positions} \leftarrow$ 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

There's a problem with this:

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

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

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

Problems with multiplying lots of probs

There's a problem with this:

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

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

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

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

We'll sum logs of probabilities instead of multiplying probabilities!

We actually do everything in log space

Instead of this:

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

$$C_{NB} = \underset{c_j \in C}{\operatorname{argmax}} \left[\log P(c_j) + \sum_{i \in position} \log P(x_i \mid c_j) \right]$$

We actually 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:

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

Notes:

1) Taking log doesn't change the ranking of classes!

   The class with highest probability also has highest log probability!

2) It's a linear model:

   Just a max of a sum of weights: a **linear** function of the inputs

   So naive bayes is a **linear classifier**

Naive Bayes: Learning

Learning A Multinomial Naive Bayes Model

First attempt: maximum likelihood estimates

- simply use the frequencies in the data

$$\hat{P}(c_j) = \frac{N_{c_j}}{N_{total}} = \frac{N \text{ fiction docs}}{N \text{ total docs}}$$

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

Create a mega-doc for topic c_j by concatenating all c_j docs.

Learning A Multinomial Naive Bayes Model

First attempt: maximum likelihood estimates

- simply 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

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

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 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** (***thumbs-up)***?

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

Problem with Maximum Likelihood

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

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

If we naively multiply, we will lose *all* probability for this class!

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

Solution: Smoothing!

Laplace (add-1) smoothing for Naïve Bayes:

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

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

Solution: Smoothing!

Laplace (add-1) smoothing for Naïve Bayes:

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

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

Multinomial Naïve Bayes: Learning

- From training corpus, extract vocabulary V

Calculate *priors:*

For each c_j in C:

docs$_j$ = n docs in class c

$$p(c_j) = \frac{|\text{doc}_j|}{|\text{total \# of docs}|}$$

Calculate *likelihoods:*

- Text$_j$ = single doc containing all docs in class j.
- n = # of words in Text$_j$
- For each word w_k in V:

n_k = # of w_k in Text$_j$

$$p(w_k \mid c_j) = \frac{n_k + \alpha}{n + \alpha[\text{Vocab}]}$$

$$= \frac{n_k + 1}{n + V}$$

Multinomial Naïve Bayes: Learning

- From training corpus, extract vocabulary V

Calculate *priors:*

- For each c_j in C:

  $docs_j = n$ docs in class c

  $$p(c_j) = \frac{|docs_j|}{|\text{total \# documents}|}$$

Calculate *likelihoods:*

- $Text_j$ = single doc containing all $docs_j$
- For each word w_k in V:

  $n_k = \#$ of w_k in $Text_j$

  $$p(w_k|c_j) = \frac{n_k + \alpha}{n + \alpha|Vocabulary|}$$

$$p(w_k | c_j) = \frac{n_k + 1}{n + V}$$

$$p(\ddot{\smile} | \text{Fiction}) = \frac{1}{n_{\text{Fiction}} + V} = \frac{1}{n_{\text{Fiction}}}$$

$$p(\ddot{\vee} | \text{Nonfiction}) = \frac{1}{n_{NF} + V} = \frac{1}{N_{NF}}$$

Unknown words

What about unknown words

- that appear in our test data
- but not in our training data or vocabulary?

We **ignore** them

- Remove them from the test document!
- Pretend they weren't there!
- Don't include any probability for them at all!

Why don't we build an unknown word model?

- It doesn't help: knowing which class has more unknown words is not generally helpful!

Text Classification and Naive Bayes

Sentiment and Binary Naive Bayes

Let's do a worked sentiment 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

A worked sentiment example with add-1 smoothing

	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. Priors from training:

$$P(-) = \frac{3}{5} \qquad P(+) = \frac{2}{5}$$

2. Drop "with"

3. Likelihoods from training:

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

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

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

4. Scoring the test set:

$$P(-)P(S|-) = \frac{3}{5} \cdot \prod (w|-) = \frac{3}{5} \cdot \frac{1}{16} \cdot \frac{1}{16} \cdot \frac{1}{34} = 6.1 \times 10^{-5}$$

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

A worked sentiment example with add-1 smoothing

	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. Priors from training:

$$P(-) = \frac{3}{5} \qquad P(+) = \frac{2}{5}$$

2. Drop "with"

3. Likelihoods from training:

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

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

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

4. Scoring the test set:

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

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

Optimizing for sentiment analysis

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

- The occurrence of the word *fantastic* tells us a lot
- The fact that it occurs 5 times may not tell us much more.

Binary multinominal naive bayes, or **binary NB**

- Clip our word counts at 1

Text Classification and Naive Bayes

More on Sentiment Classification

Sentiment Classification: Dealing with Negation

I really like this movie

I really **don't** like this movie

Negation changes the meaning of "like" to negative.

Negation can also change negative to positive-ish

- **Don't** dismiss this film
- **Doesn't** let us get bored

Sentiment Classification: Dealing with Negation

Das, Sanjiv and Mike Chen. 2001. Yahoo! for Amazon: Extracting market sentiment from stock message boards. In Proceedings of the Asia Pacific Finance Association Annual Conference (APFA).

Bo Pang, Lillian Lee, and Shivakumar Vaithyanathan. 2002. Thumbs up? Sentiment Classification using Machine Learning Techniques. EMNLP-2002, 79—86.

Simple baseline method:

Add NOT_ to every word between negation and following punctuation:

```
didn't like this movie , but I
```

↓

```
didn't NOT_like NOT_this NOT_movie but I
```

Sentiment Classification: Lexicons

Problem: sometimes, we don't have labeled data

Solution: use a pre-defined **lexicon** of words that are good predictors

MPQA Subjectivity Cues Lexicon

Theresa Wilson, Janyce Wiebe, and Paul Hoffmann (2005). Recognizing Contextual Polarity in Phrase-Level Sentiment Analysis. Proc. of HLT-EMNLP-2005.

Riloff and Wiebe (2003). Learning extraction patterns for subjective expressions. EMNLP-2003.

Home page: https://mpqa.cs.pitt.edu/lexicons/subj_lexicon/

6885 words from 8221 lemmas, annotated for intensity (strong/weak)

- 2718 positive
- 4912 negative

+ : *admirable, beautiful, confident, dazzling, ecstatic, favor, glee, great*

− : *awful, bad, bias, catastrophe, cheat, deny, envious, foul, harsh, hate*

The General Inquirer

Philip J. Stone, Dexter C Dunphy, Marshall S. Smith, Daniel M. Ogilvie. 1966. The General Inquirer: A Computer Approach to Content Analysis. MIT Press

- Home page: http://www.wjh.harvard.edu/~inquirer
- List of Categories: http://www.wjh.harvard.edu/~inquirer/homecat.htm
- Spreadsheet: http://www.wjh.harvard.edu/~inquirer/inquirerbasic.xls

Categories:

- Positiv (1915 words) and Negativ (2291 words)
- Strong vs Weak, Active vs Passive, Overstated versus Understated
- Pleasure, Pain, Virtue, Vice, Motivation, Cognitive Orientation, etc

Free for Research Use

Using Lexicons in Sentiment Classification

Add a feature that gets a count whenever a word from the lexicon occurs

- E.g., a feature called "**this word occurs in the positive lexicon**" or "**this word occurs in the negative lexicon**"

Now all positive words (*good, great, beautiful, wonderful*) or negative words count for that feature.

Using 1-2 features isn't as good as using all the words.

- But when training data is sparse or not representative of the test set, dense lexicon features can help

Naive Bayes in Other tasks: Spam Filtering

SpamAssassin Features:

- Mentions millions of (dollar) ((dollar) NN,NNN,NNN.NN)
- From: starts with many numbers
- Subject is all capitals
- HTML has a low ratio of text to image area
- "One hundred percent guaranteed"
- Claims you can be removed from the list

Naive Bayes in Language ID

Determining what language a piece of text is written in.

Features based on character n-grams do very well

Important to train on lots of varieties of each language

(e.g., American English varieties like African-American English, or English varieties around the world like Indian English)

Summary: Naive Bayes is Not So Naive

Very Fast, low storage requirements

Work well with very small amounts of training data

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**

Slide from Chris Manning