CS 333: Natural Language Processing

Fall 2025

Prof. Carolyn Anderson

Wellesley College

Reminders

- HW 3 due on Thursday
- Caroline's drop-in hour is today
- Ashley's drop-in hour shifting to 5-6pm tomorrow due to CS colloquium conflict.
- My help hours: Thursday 4-5

WELLESLEY
COMPUTER SCIENCE

CS COLLOQUIUM

INNOVATING WITH UNDERSERVED COMMUNITIES: SUCCESSES, FAILURES, AND LESSONS LEARNED

Bill Thies

Everwell Technologies,

WEDNESDAY, SEPTEMBER 24

SCI-H105

4:00 PM – 5:00 PM

SNACKS WILL BE SERVED AT 3:45 PM

???: sb129

Accessibility and Disability: accessibility@wellesley.edu

WELLESLEY

New Policy

Earn an extra late day by attending a CS research talk!

To qualify, the talk must be **in-person** with the opportunity to **ask questions**.

Send me a **brief summary** of the talk and what you learned (1-2 paragraphs), and I will note down an extra late day.

Curiosity Points on HW 1

People did very interesting things on HW 1!

- Lots of improvements to chatbot:
  - Rhyme analysis, user ratings, Jaccard similarity, repetition preferences, better error handling…
- Extra regex practice
- Literature search on regex, on chatbots, on poetry generation
- Thorough analyses, including a user study
- Exploration of sentiment in a different corpus

Text Classification Tasks

https://xkcd.com/2451/

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)

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

Sentiment Analysis

Sentiment analysis is the detection of **attitudes**

Today we will simply ask:

- Is the attitude of this text positive or negative?

Defining the Text Classification Task

Text Classification

Input:

- a document *d*
- a fixed set of classes $C = \{c_1, c_2, \ldots, c_J\}$

Output: a predicted class $c \in C$

Text Classification: Hard-Coded Rules

- Rules based on combinations of words or other features:

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

- Accuracy can be high if the rules are carefully refined by an expert

- But building and maintaining is expensive. Plus, what happens when the new fight cloud emoji gets added this year?

Text Classification: 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:

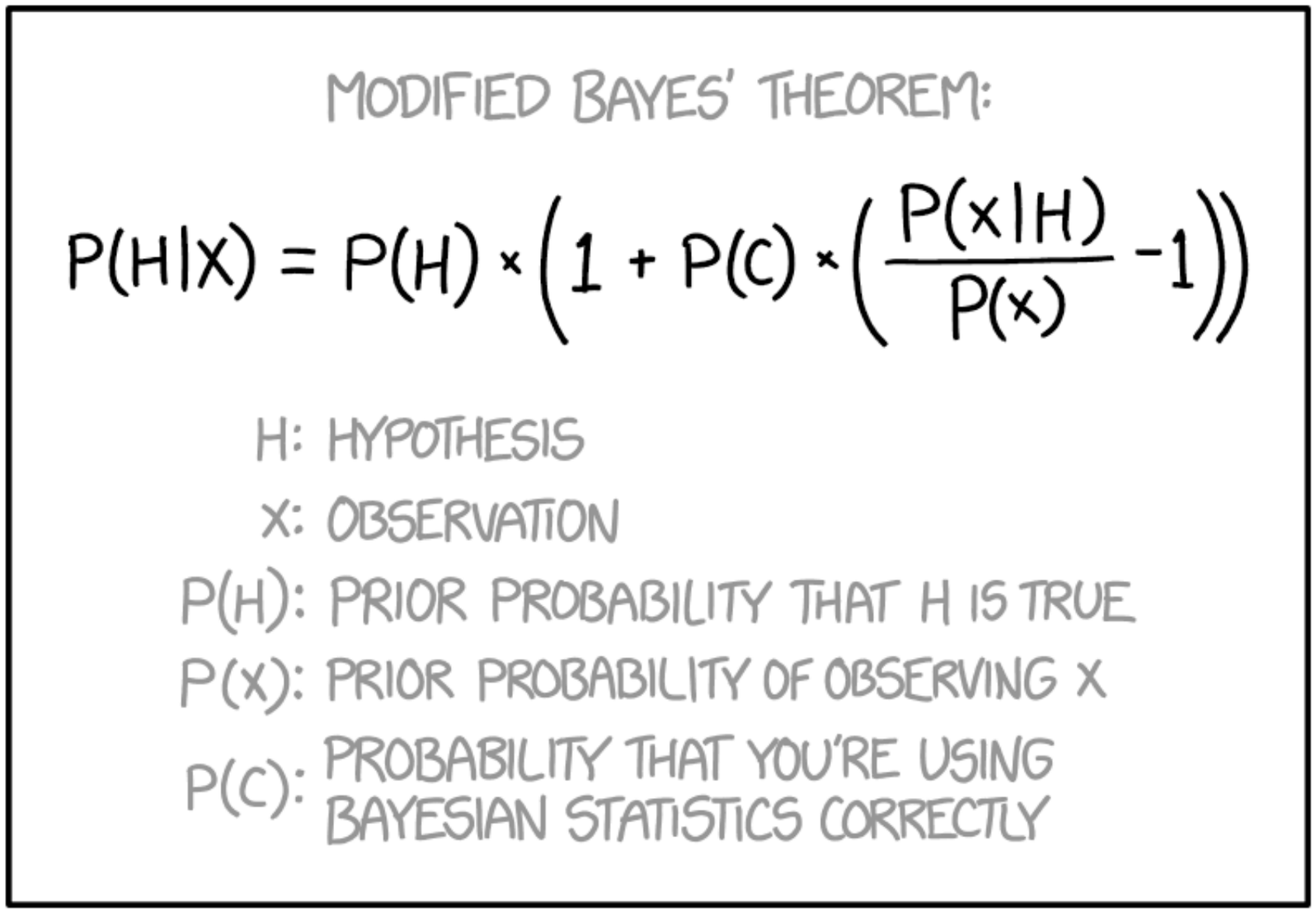
- a learned classifier $\gamma:d \rightarrow c$

Supervised Machine Learning Classifiers

There are many kinds of classifiers

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

Naive Bayes Classifiers



https://xkcd.com/2059/

Naive Bayes Intuition

"Naive" classification method based on Bayes rule:

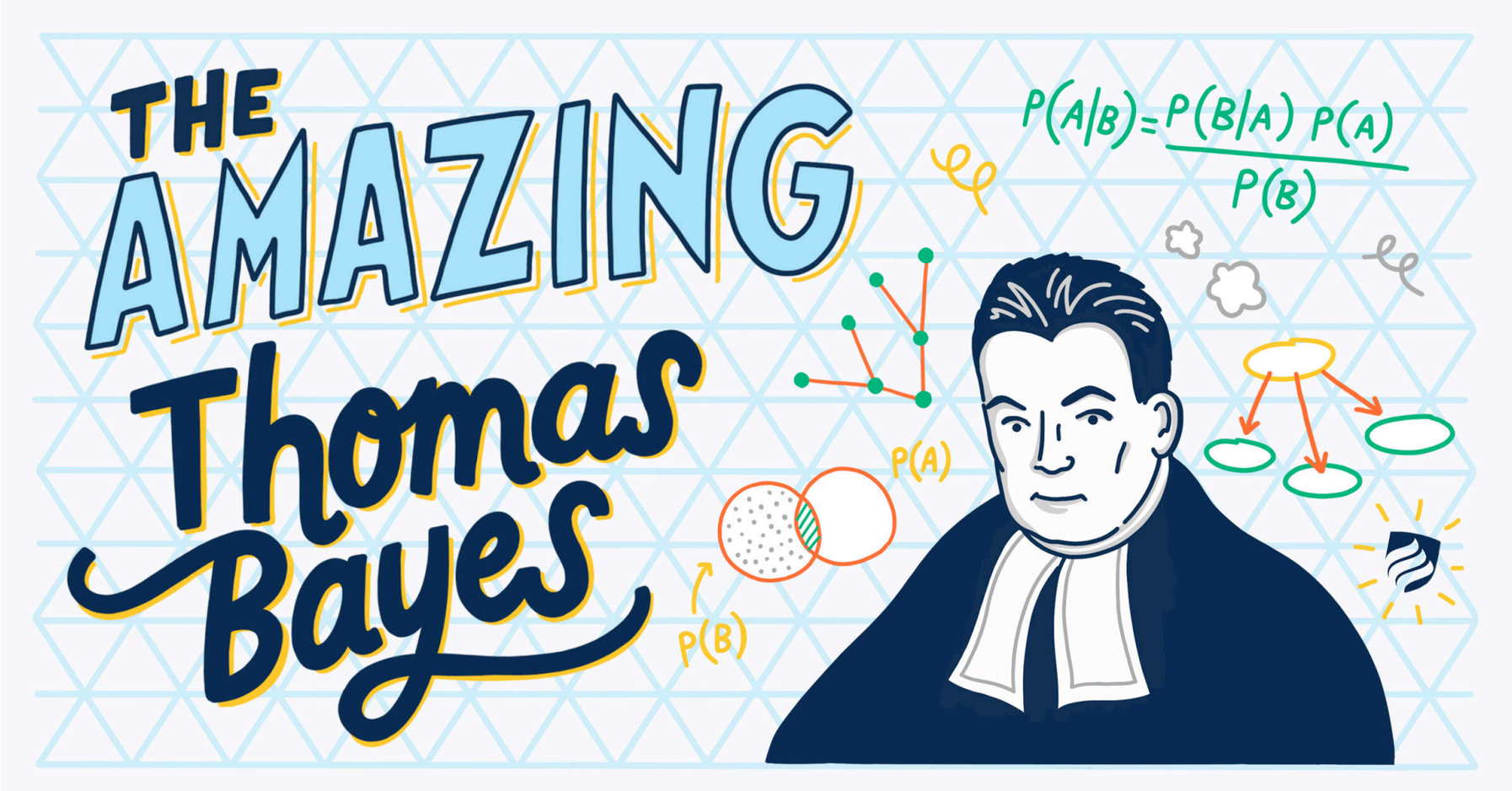


Image: Jim Kulich (https://www.elmhurst.edu/blog/thomas-bayes)

Naive Bayes Intuition

A "naive" classification method based on Bayes rule:

Naive Bayes Intuition

A "naive" classification method based on Bayes rule:

$$P(A \mid B) = P(B)\, P(B \mid A) \;/\; P(A)$$

$$P(\text{label} \mid \text{features}) = P(\text{label})\, P(\text{features} \mid \text{label})$$

Naive Representation

A "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-e

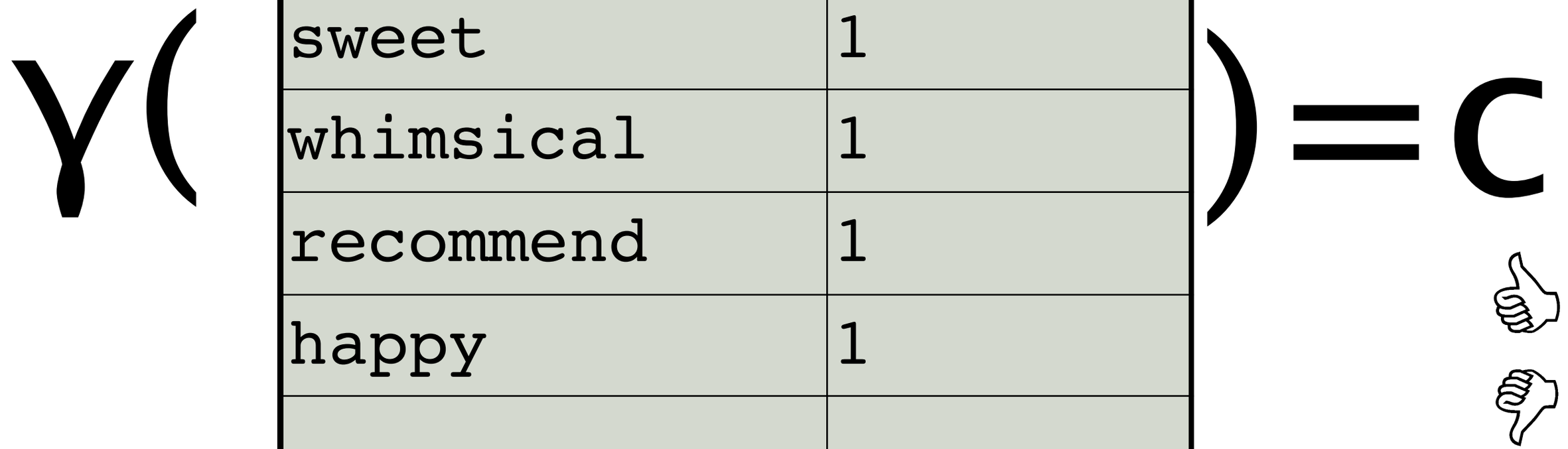
Bag of Words Representation

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
…	…

Word Counts as Features

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



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

MAP is "maximum a posteriori" = most likely class

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

Bayes Rule

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

Dropping the denominator

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

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

Naive Bayes Classifier

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

How often does this class occur?

We can count frequencies in a corpus

Naive 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

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

Multinomial Naive Bayes: Assumptions

Bag of Words Assumption: Assume position doesn't matter

Multinomial Naive Bayes: Assumptions

Bag of Words Assumption: Assume position doesn't matter

Conditional Independence: Assume the feature probabilities $P(x_i \mid c_j)$ 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

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

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

Multinomial Naive Bayes

positions ← all word positions in test document

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

One Issue

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….

Log Fix

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….

Solution: Use logs, because $\log(ab) = \log(a) + \log(b)$

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

Multinomial Naive Bayes

Instead of this:

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

Multinomial Naive Bayes

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

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

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 Classification: Training Phase

Learning A Multinomial Naive Bayes Model

First attempt: maximum likelihood estimates.

Simply use the frequencies in the data!

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

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

Learning A Multinomial Naive Bayes Model

- Create mega-document for topic *j* by concatenating all docs in this topic.
- Use frequency of *w* in mega-document

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

Learning A Multinomial Naive Bayes Model

- Create mega-document for topic *j* by concatenating all docs in this topic.
- Use frequency of *w* in mega-document

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

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

Problem with Maximum Likelihood

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

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

Problem with Maximum Likelihood

Sec. 13.3

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!

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

Solution: Smoothing!

Laplace (add-1) smoothing for Naive Bayes:

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

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

Training A Multinomial Naive Bayes Classifier

- Step 1: from training corpus, extract vocabulary V
- Step 2: Calculate priors
- Step 3: Calculate likelihoods

Calculating Priors

Calculate *priors:*

- For each c_j in C:

  $docs_j$ = n docs in class c

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

Calculating Likelihoods

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

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

where α = smoothing parameter

Unknown Words

What about **unknown words**: 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?

- Generally it doesn’t help to know which class has more unknown words.

Sentiment Analysis: A Binary Naive Bayes Example

Sentiment Analysis

Let's work through a 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

Smoothed Model

	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 unknown words ("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. Score test set:

$$P(-)P(S|-) =$$

$$P(+)P(S|+) =$$

Smoothed Model

	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 unknown words ("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. Score 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 Naive Bayes:

clip word counts at 1 to create binary features

Sentiment Lexicons

Can use features that track counts in sentiment lexicons:

- Features represent *"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, supplementing with lexicon features can help

MPQA Subjectivity Cues Lexicon

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*

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.