
SUPPORT
VECTOR
MACHINES
REPEA
POWE
LAWS
END
DUALITY
GAP
Map Reduce
Map Reuse
Map Recycle
BAYSIANS
AGAINST
DISCRIMINATION
FREE
VARIABLES!
BAN
GENETIC
ALGORITHMS

CS 2731 Introduction to Natural Language Processing

Session 4: Naive Bayes

Michael Miller Yoder

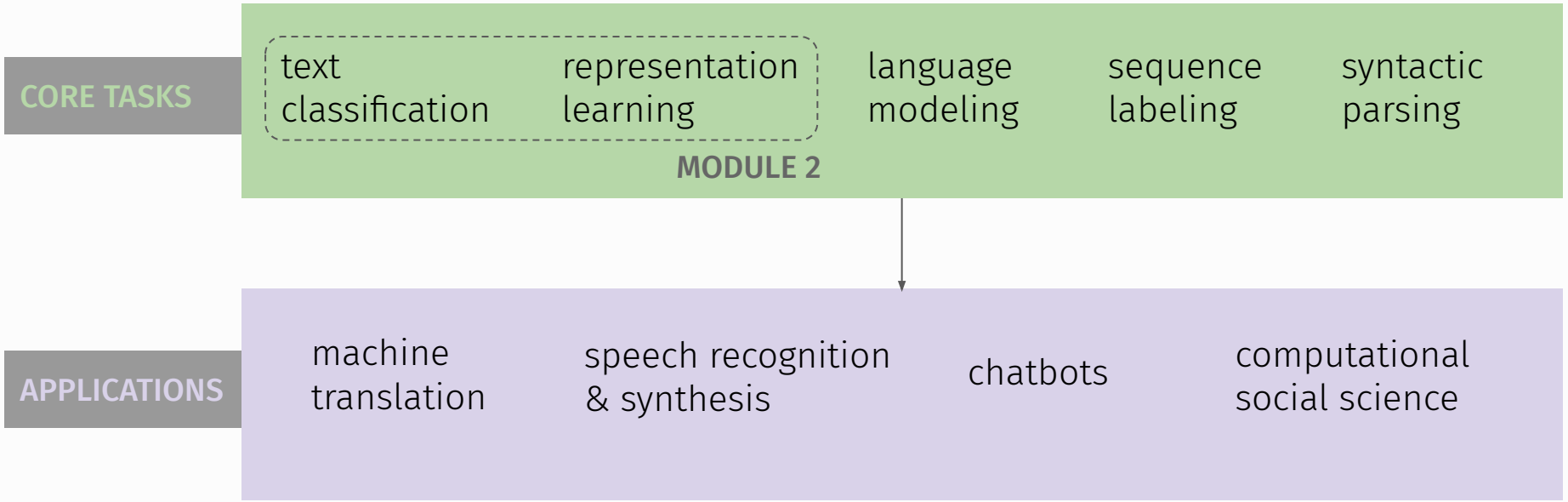
September , 2023

- Course logistics
- Text classification tasks
- Naive Bayes: input, formulation, training

Course logistics

- Homework 1
  - Programming problems, write report based on results
  - Due Thu at 11:59pm
- Course project matching
  - We will email you today
  - Please plan meeting with groups to discuss project ideas
  - Project area and contribution form will be due 09-21 (not released yet)
- First discussion post due Wed at noon

Core tasks and applications of NLP



Text classification tasks

Text classification

"My dear Mr. Bennet," said his lady to him one day, "have you heard that Netherfield Park is let at last?"

ROMANCE

Pride and Prejudice

DIALOG

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.

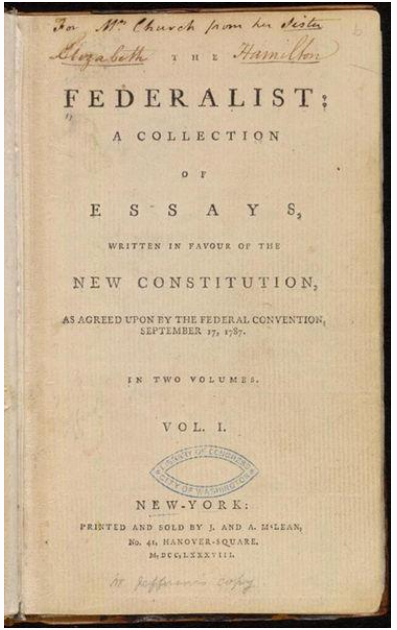
http://www.123contactform.com/contact-form-StanfordNew1-236335.html

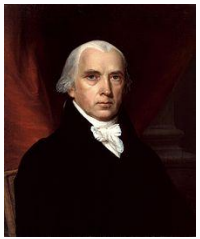
Click on the above link to login for more information about this new exciting forum. You can also copy the above link to your browser bar and login for more information
about the new services.

Slide adapted from Jurafksy & Martin

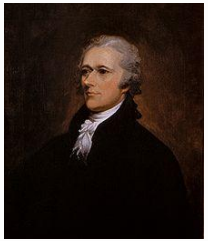
Who wrote which Federalist papers?

- 1787-1788: anonymous essays try to convince New York to ratify U.S Constitution: Jay, Madison, Hamilton.
- Authorship of 12 of the letters in dispute
- 1963: solved by Mosteller and Wallace using Bayesian methods





James Madison



Alexander Hamilton

Slide adapted from Jurafksy & Martin

What is the subject of this medical article?

MEDLINE Article

Brain and Cognition

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

?

MeSH Subject Category Hierarchy

Antagonists and Inhibitors

Blood Supply

Chemistry

Drug Therapy

Embryology

Epidemiology

...

Slide adapted from Jurafksy & Martin

Text Classification

We have a set of documents that we want to *classify* into a small set *classes*.

Applications:

- **Topic classification:** you have a set of news articles that you want to classify as finance, politics, or sports.
- **Sentiment detection:** you have a set of movie reviews that you want to classify as good, bad, or neutral.
- **Language Identification:** you have a set of documents that you want to classify as English, Mandarin, Arabic, or Hindi.
- **Reading level:** you have a set of articles that you want to classify as kindergarten, 1st grade, ...12th grade.
- **Author identification:** you have a set of fictional works that you want to classify as Shakespeare, James Joyce, ...
- **Genre identification:** you have a set of documents that you want to classify as report, editorial, advertisement, blog, ...

Slide credit: David Mortensen

Notation and Setting

- We have a set of n documents (texts) $d_i \in \mathcal{V}^+$, where $\mathcal{V}$ is the vocabulary of the corpus.
  - We assume the texts are segmented already.
- We have set $\mathcal{L}$ of labels, ℓ_i
- Human experts annotate documents with labels and give us $\{(d_1, \ell_1), (d_2, \ell_2), \cdots, (d_n, \ell_n)\}$
- We learn a *classifier* $\texttt{classify} : \mathcal{V}^+ \rightarrow \mathcal{L}$ with this labeled training data.
- Afterwards, we use `classify` to classify new documents into their classes.

Slide credit: David Mortensen

Example: Sentiment Detection

	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

Slide credit: David Mortensen

Naive Bayes classification

Do You Like Sportsball?

Slide credit: David Mortensen

Your mission: identify every English-language sports article on the Internet

Slide credit: David Mortensen

Observation: 80% of sports articles contain the word *win*.

You assume that all articles containing *win* are likely to be sports articles

Slide credit: David Mortensen

Enter Bayes

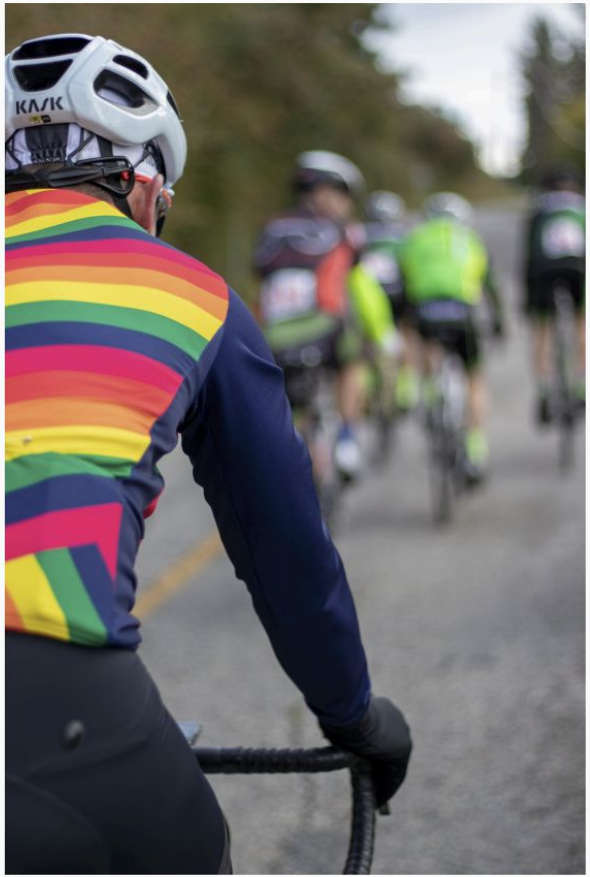
But the Good Reverend Bayes asks you:

How many times does *win* occur generally?
And what percentage of articles on your newfangled (and possibly demonic) Internet are about sports?

Slide credit: David Mortensen

Frequency of *win*: 0.0005

Frequency of **sports**: 0.01



Slide credit: David Mortensen

Bayes' Rule

$$\hat{P}(\text{sports}|\text{win}) = \frac{P(\text{win}|\text{sports})P(\text{sports})}{P(\text{win})}$$

Slide credit: David Mortensen

Bayes' Rule

$$\hat{P}(\text{sports}|\text{win}) = \frac{0.8P(\text{sports})}{P(\text{win})}$$

Slide credit: David Mortensen

Bayes' Rule

$$\hat{P}(\text{sports}|\text{win}) = \frac{0.8(0.01)}{P(\text{win})}$$

Slide credit: David Mortensen

Bayes' Rule

$$0.16 = \frac{0.8(0.01)}{0.0005}$$

Slide credit: David Mortensen

In this example, the probability that a document is about `sports` given that it contains *win* is only 16%

- LIKELIHOOD is high $P(\text{win}|\text{sports})$
- PRIOR is relatively low $P(\text{sports})$
- MARGINAL LIKELIHOOD is relatively high $P(\text{win})$

Slide credit: David Mortensen

Input to classification tasks: features

Movie Ratings

- A training set of movie reviews (with star ratings 1 - 5)
- A set of features for each message (considered as a bag of words)
  - For each word: Number of occurrences
  - Whether phrases such as *Excellent*, *sucks*, *blockbuster*, *biggest*, *Star Wars*, *Disney*, *Adam Sandler*, ...are in the review

Slide credit: David Mortensen

Bag of words document representation

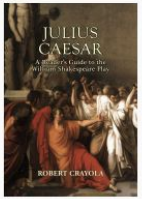
I love this movie! It's sweet, but with satirical humor. The dialogue is great and the adventure scenes are fun... It manages to be whimsical and romantic while laughing at the conventions of the fairy tale genre. I would recommend it to just about anyone. I've seen it several times, and I'm always happy to see it again whenever I have a friend who hasn't seen it yet!

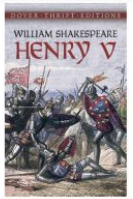
it	6
I	5
the	4
to	3
and	3
seen	2
yet	1
would	1
whimsical	1
times	1
sweet	1
satirical	1
adventure	1
genre	1
fairy	1
humor	1
have	1
great	1
…	…

Slide from Jurafsky & Martin

Term-document matrix

- Each cell is the count of term t in a document d ($tf_{t,d}$).
- Each document is a count vector in $\mathbb{N}^V$, a column below.





	As You Like It	Twelfth Night	Julius Caesar	Henry V
battle	1	1	8	15
soldier	2	2	12	36
fool	37	58	1	5
clown	6	117	0	0

Slide credit: David Mortensen

Spam Detection

- A training set of email messages (marked *Spam* or *Not-Spam*)
- A set of features for each message
  - For each word: Number of occurrences
  - Whether phrases such as “Nigerian Prince”, “email quota full”, “won ONE HUNDRED MILLION DOLLARS” are in the message
  - Whether it is from someone you know
  - Whether it is a reply to your message
  - Whether it is from your domain (e.g., `cmu.edu`)

Slide credit: David Mortensen

Features in Text Classification

- Running example $\boldsymbol{d}$ = "The spirit is willing but the flesh is weak"
- *Feature random variables*
- For $j \in \{1, \ldots, d\}$ F_j is a discrete random variable taking values in $\mathcal{F}_j$
- Most of the time these can be frequencies of words or n-grams (including character n-grams) in a text.
  - $f_{f-spirit} = 1, f_{f-is} = 2, f_{f-the-flesh} = 1, \ldots$
- They can be boolean "exists" features.
  - $f_{e-spirit} = 1, f_{e-is} = 1, f_{f-strong} = 0, \ldots$

Slide credit: David Mortensen

Naive Bayes formulation

Bayes' Rule applied to documents and classes

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

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

Slide adapted from Jurafksy & Martin

Naive Bayes classification formula

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

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

Slide adapted from Jurafksy & Martin

Naive Bayes classification formula

"Likelihood" "Prior"

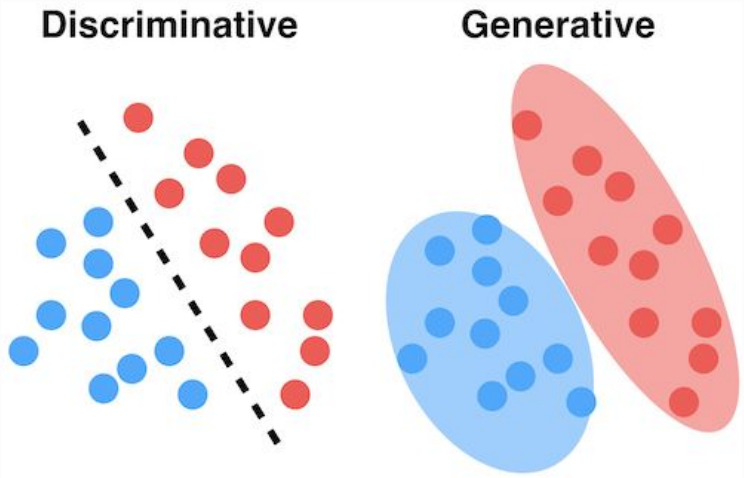
$$c_{MAP} = \underset{c \in C}{\operatorname{argmax}} P(d \mid c) P(c)$$

$P(c,d) = P(d|c)P(c)$ by def of conditional probability/chain rule

Slide adapted from Jurafksy & Martin

Generative vs discriminative classifiers

- Naive Bayes is a generative classifier
- Generative classifiers build a model of each class
  - $P(d|c)P(c) = P(c,d)$ joint probability
  - Given an observation, return the class most likely to have generated that observation
- Discriminative classifiers instead learn what features are useful to discriminate between possible classes
  - $P(c|d)$ directly model conditional probability



Slide credit: David Mortensen

Generative Classifiers and “Generative AI”

In 2023, journalists and MBAs became very interested in “Generative AI”.

- DALL-E
- ChatGPT
- Stable Diffusion
- Midjourney

These are all generative models—like NB classifiers—in that they can **generate** novel outputs from the same distribution as their inputs.
They are different from generative classifiers in that they typically take natural language PROMPTS as input.

Slide credit: David Mortensen

(Back to the) Naive Bayes classification formula

"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 $x_1 \ldots x_n$

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

How often does this class occur? Count the relative frequency in a corpus

Slide adapted from Jurafksy & Martin

Naive Bayes independence assumptions

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

Bag of words assumption: Assume word position doesn't matter

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

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

Slide adapted from Jurafksy & Martin

Naive Bayes classification formula

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

For all word positions (tokens) in a test document:

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

Slide adapted from Jurafksy & Martin

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!

Slide adapted from Jurafksy & Martin

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**

Slide adapted from Jurafksy & Martin

Types of Naive Bayes

How to calculate $P(x_i|c_j)$ depends on what type of feature x_i and classifier

- **Gaussian Naïve Bayes** Considers a feature vector of continuous variables
- **Multinomial Naïve Bayes** Considers a feature vector representing frequencies
- **Bernoulli Naïve Bayes** Considers a vector of binary features

For sentiment analysis, language ID, and other document classification tasks, **Multinomial Naïve Bayes** is generally best

Slide credit: David Mortensen

Multinomial Naive Bayes Model: Training

First attempt: maximum likelihood estimates

- Simply use the frequencies in the data (c=class, w=word)

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

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

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

Slide adapted from Jurafksy & Martin

Problem with Maximum Likelihood (Zeros)

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

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

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

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

Slide adapted from Jurafksy & Martin

Laplace (add 1) smoothing for Naïve Bayes

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

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

Slide adapted from Jurafksy & Martin

Smoothing Example

- Two words found in test set: good and wonderful
- There are 100 word types found with the label "positive" (+)
- In documents labeled +: *good*, 5 times; *wonderful*, 0 times
- Vocabulary (V): 1000 types

$$\hat{P}(\text{good}|+) = \frac{\text{count}(\text{good},+)+1}{\left(\sum_{w\in V}\text{count}(w,+)\right)+|V|}$$

$$\frac{5+1}{100+1000}$$

$$\hat{P}(\text{wonderful}|+) = \frac{\text{count}(\text{wonderful},+)+1}{\left(\sum_{w\in V}\text{count}(w,+)\right)+|V|}$$

$$\frac{0+1}{100+1000}$$

Slide credit: David Mortensen

A Blueprint for NB

To classify a document d:

$$\text{classify}(d) = \text{argmax}\, c \in C \left(\log\, p(c) + \sum_{i=1}^{|d|} \log\, p(w_i \mid c) \right)$$

Training:

$\mathcal{D} \leftarrow$ all documents

for each $c \in C$ **do**

$\mathcal{D}_c \leftarrow$ docs with class c

$p(c) \leftarrow \frac{|\mathcal{D}_c|}{|\mathcal{D}|}$

Training:

$T_c \leftarrow \text{concatenate}(\mathcal{D}_c)$

for each $w_i \in V$ **do**

$n_i \leftarrow \text{count}(w_i, T_c)$

$p(w_i|c) \leftarrow \frac{n_i+\alpha}{|T_c|+\alpha|V|}$

At inference time, compute smoothed probabilities for words that were not in the training set.

Slide credit: David Mortensen

An Example

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

- $|V| = 20$
- Add 1 Laplace smoothing

$$p(\text{“predictable”} \mid -) = \frac{1+1}{13+20} \quad p(\text{“predictable”} \mid +) = \frac{0+1}{9+20}$$

$$p(\text{“no”} \mid -) = \frac{1+1}{13+20} \quad p(\text{“no”} \mid +) = \frac{0+1}{9+20}$$

$$p(\text{“fun”} \mid -) = \frac{0+1}{13+20} \quad p(\text{“fun”} \mid +) = \frac{1+1}{9+20}$$

$$\pi_- = p(-) = \frac{3}{5}$$

$$\pi_+ = p(+) = \frac{2}{5}$$

$$N_- = 13$$

$$N_+ = 9$$

$$p(+)p(s \mid +) = \frac{2}{5} \times \frac{1 \times 1 \times 2}{29^3} = 3.2 \times 10^{-5}$$

$$p(-)p(s \mid -) = \frac{3}{5} \times \frac{2 \times 2 \times 1}{33^3} = 6.7 \times 10^{-5}$$

Slide credit: David Mortensen

Other Optimizations for Sentiment Analysis

- Ignore unknown words in the test.
- Ignore **stop words** like *the*, *a*, *with*, etc.
  - Remove most frequent 10-100 words from the training and test documents.
- Count vs existence of words: Binarized features.
- Negation Handling `didnt like this movie , but I` $\rightarrow$ `didnt NOT_like NOT_this NOT_movie , but I`

Questions?

Homework 1 due Thu, Sep 13