text classification with naive Bayes

CS 585, Fall 2019

Introduction to Natural Language Processing

http://people.cs.umass.edu/~miyyer/cs585/

Mohit Iyyer

College of Information and Computer Sciences

University of Massachusetts Amherst

pick up an exercise at the front of the class!

also, *if you are not currently registered*, please write your name/ID on the sheet at the front of class.

In-class exercise policies

- Attendance, which we keep track of via in-class exercises, will be a part of your overall participation grade. Everyone can miss up to **two** in-class exercises with no penalty; any further absences will lower your attendance score
- If you have to miss more than two classes for legitimate preplanned reasons (e.g., interviews) or for health/personal emergencies, please contact the instructors at cs585nlp@gmail.com

Late policies:

- Late policy: everyone will get **three** late days to use for homework assignments. After all three late days have been exhausted, no more late submissions will be accepted.
- For unforeseen health and personal emergencies, please email the instructors account. Job interviews / other schoolwork are **not** excuses for late homework.

questions from last class…

- why am i not on gradescope?
  - please consent on the poll or we can’t add you!

- do NOT email me or cs585nlp@gmail with course registration issues! we can’t do anything

tentative roadmap

- today: naive Bayes for text classification
- next week: count-based language models
- following week: logistic regression for text classification
- following week: word representations and neural language models

text classification

- input: some text **x** (e.g., sentence, document)
- output: a label **y** (from a finite label set)
- goal: learn a mapping function *f* from **x** to **y**

text classification

- input: some text **x** (e.g., sentence, document)
- output: a label **y** (from a finite label set)
- goal: learn a mapping function *f* from **x** to **y**

fyi: basically every NLP problem reduces to learning a mapping function with various definitions of **x** and **y**!

problem	**x**	**y**
sentiment analysis	text from reviews (e.g., IMDB)	{positive, negative}
topic identification	documents	{sports, news, health, …}
author identification	books	{Tolkien, Shakespeare, …}
spam identification	emails	{spam, not spam}

… many more!

input **x**:

From European Union <info@eu.org>
Subject
Reply to [redacted]

Please confirm to us that you are the owner of this very email address with your copy of identity card as proof.

YOU EMAIL ID HAS WON $10,000,000.00 ON THE ONGOING EUROPEAN UNION COMPENSATION FOR SCAM VICTIMS. CONTACT OUR EMAIL:
CONTACT US NOW VIA EMAIL: [redacted] NOW TO CLAIM YOUR COMPENSATION

label **y**: **spam** or **not spam**

we'd like to learn a mapping *f* such that
$f(\mathbf{x})$ = **spam**

f can be hand-designed rules

- if “won $10,000,000” in **x**, **y** = **spam**
- if “CS585 Fall 2019” in **x**, **y** = **not spam**

what are the drawbacks of this method?

f can be **learned** from data

- given **training data** (already-labeled **x,y** pairs) learn *f* by maximizing the likelihood of the training data
- this is known as **supervised learning**

training data:

x (email text)	**y** (spam or not spam)
learn how to fly in 2 minutes	spam
send me your bank info	spam
CS585 Gradescope consent poll	not spam
click here for trillions of $$$	spam

… ideally many more examples!

heldout data:

x (email text)	**y** (spam or not spam)
CS585 important update	not spam
ancient unicorns speaking english!!!	spam

training data:

x (email text)	**y** (spam or not spam)
learn how to fly in 2 minutes	spam
send me your bank info	spam
CS585 Gradescope consent poll	not spam
click here for trillions of $$$	spam
… ideally many more examples!	

heldout data:

x (email text)	**y** (spam or not spam)
CS585 important update	not spam
ancient unicorns speaking english!!!	spam

learn mapping function on training data, measure its accuracy on heldout data

probability review

- random variable X takes value x with probability $p(X = x)$; shorthand $p(x)$
- joint probability: $p(X = x, Y = y)$
- conditional probability: $p(X = x \mid Y = y)$

$$= \frac{p(X = x, Y = y)}{p(Y = y)}$$

- when does $p(X = x, Y = y) = p(X = x) \cdot p(Y = y)$?

probability of some input text

- goal: assign a probability to a sentence
  - sentence: sequence of *tokens*
    $p(w_1, w_2, w_3, \ldots, w_n)$
    p(the cat sleeps) > p(cat sleeps the)
  - $w_i \in V$ where V is the **vocabulary** (*types*)
- some constraints:

non-negativity for any $w \in V, p(w) \geq 0$

probability distribution, sums to 1 $\sum_{w \in V} p(w) = 1$

how to estimate p(sentence)?

$$p(w_1, w_2, w_3, \ldots, w_n)$$

we could count all occurrences of the sequence

$$w_1, w_2, w_3, \ldots, w_n$$

in some large dataset and normalize by the number of sequences of length n in that dataset

how many *parameters* would this require?

chain rule

$$p(w_1, w_2, w_3, \ldots, w_n)$$

$$= p(w_1) \cdot p(w_2 \,|\, w_1) \cdot p(w_3 \,|\, w_1, w_2) \ \ldots \cdot p(w_n \,|\, w_{1 \ldots n-1})$$

in naive Bayes, the probability of generating a word is independent of all other words

$$= p(w_1) \cdot p(w_2) \cdot p(w3) \ \ldots \cdot p(w_n)$$

this is called the **unigram probability**.
what are its limitations?

an aside:

models that estimate *p*(text) are called **language models**. we will be seeing a lot of these in the rest of the class. naive Bayes uses a unigram language model, which is the simplest possible LM.

toy sentiment example

- vocabulary V: {i, hate, love, the, movie, actor}
- training data (movie reviews):
  - i hate the movie
  - i love the movie
  - i hate the actor
  - the movie i love
  - i love love love love love the movie
  - hate movie
  - i hate the actor i love the movie

labels:
positive
negative

bag-of-words representation

i hate the actor i love the movie

bag-of-words representation

i hate the actor i love the movie

word	count
i	2
hate	1
love	1
the	2
movie	1
actor	1

bag-of-words representation

i hate the actor i love the movie

word	count
i	2
hate	1
love	1
the	2
movie	1
actor	1

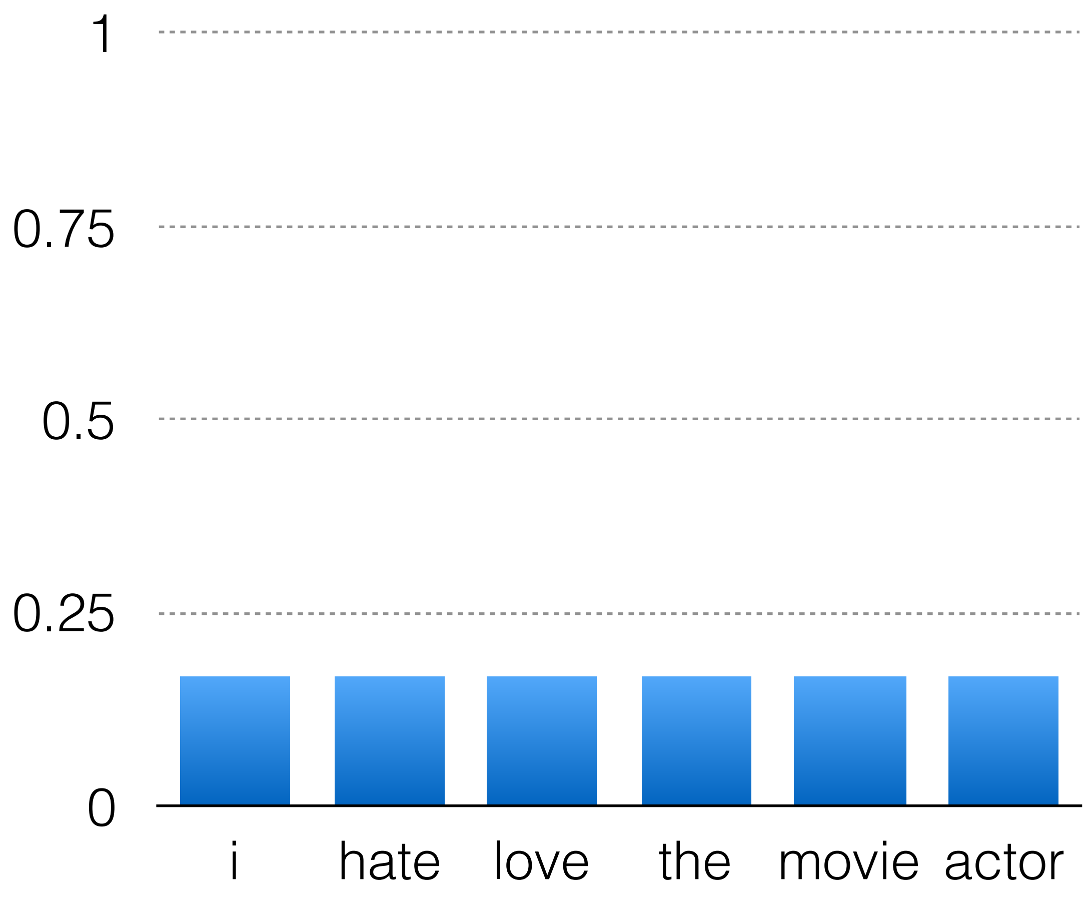
equivalent representation to:

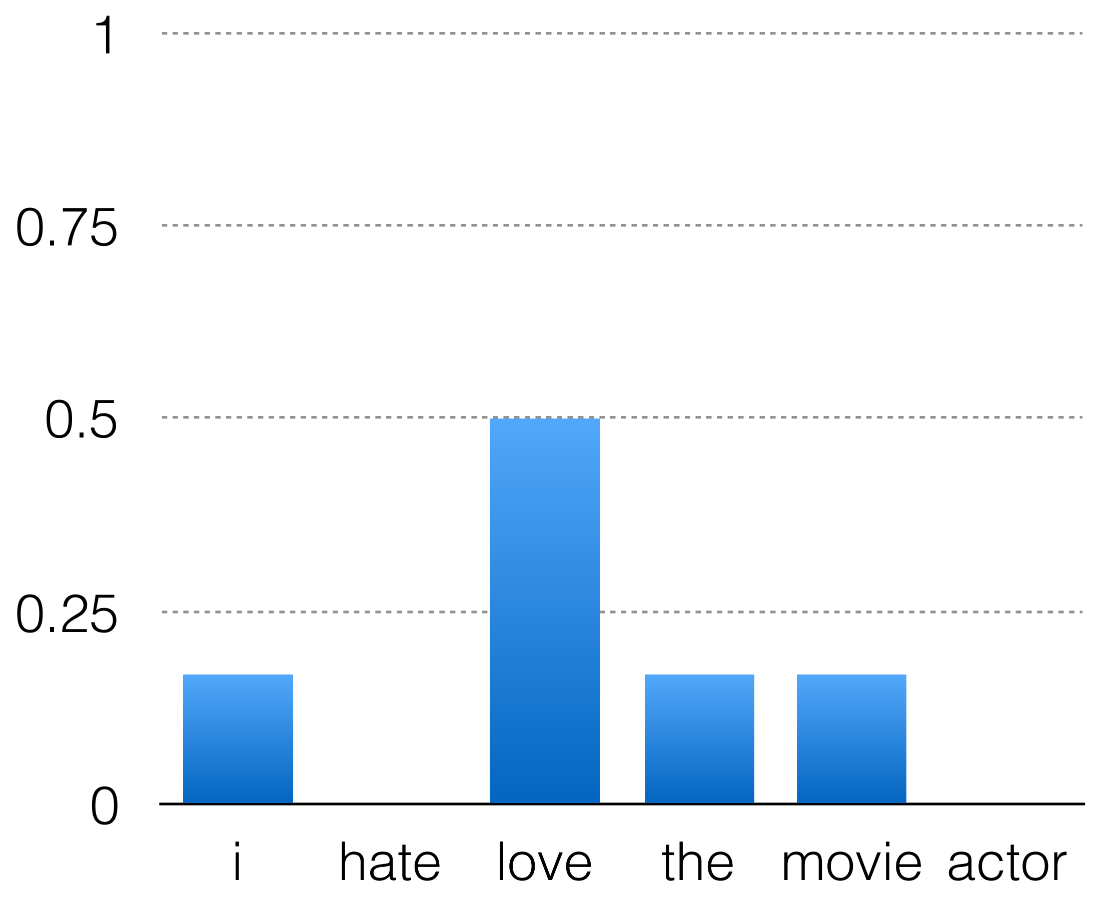
actor i i the love the movie hate

naive Bayes

- represents input text as a bag of words
- assumption: each word is independent of all other words
- given labeled data, we can use naive Bayes to estimate probabilities for unlabeled data
- **goal:** infer probability distribution that generated the labeled data for each label

which of the below distributions most likely generated the positive reviews?



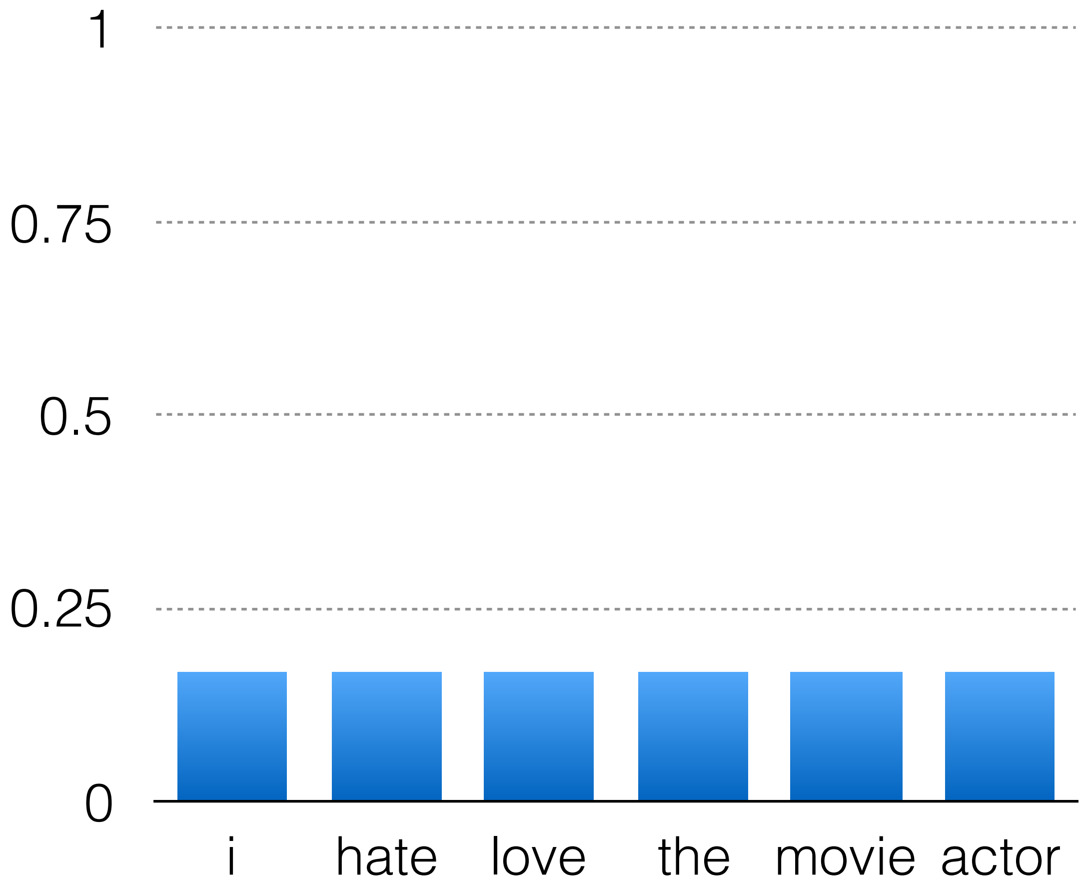


… back to our reviews

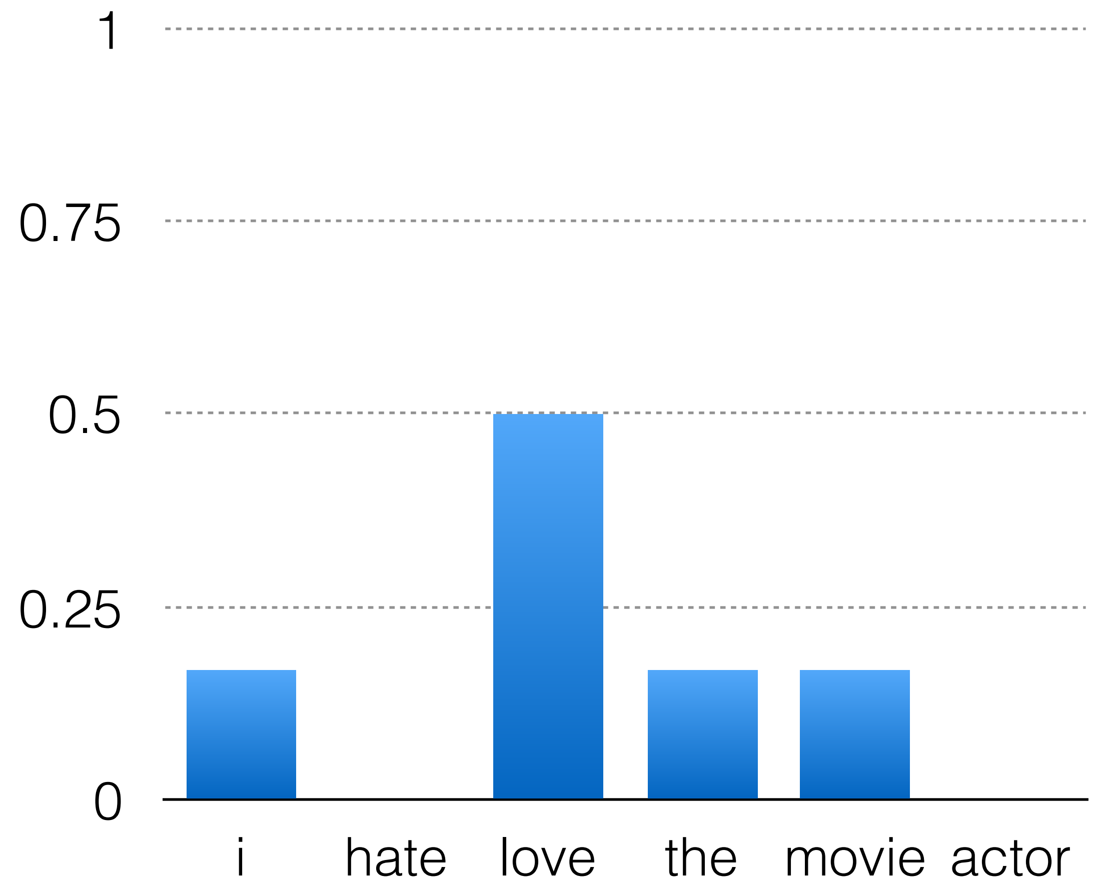
$$p(\text{i love love love love love the movie})$$

$$= p(\text{i}) \cdot p(\text{love})^5 \cdot p(\text{the}) \cdot p(\text{movie})$$

= 5.95374181e-7



= 1.4467592e-4



logs to avoid underflow

$$p(w_1) \cdot p(w_2) \cdot p(w3) \ \ldots \cdot p(w_n)$$

can get really small esp. with large *n*

$$\log \prod p(w_i) = \sum \log p(w_i)$$

$$p(\text{i}) \cdot p(\text{love})^5 \cdot p(\text{the}) \cdot p(\text{movie}) \quad = 5.95374181\text{e-}7$$

$$\log p(\text{i}) + 5 \log p(\text{love}) + \log p(\text{the}) + \log p(\text{movie})$$

$$= -14.3340757538$$

class conditional probabilities

Bayes rule (ex: x = sentence, y = label in {pos, neg})

posterior, prior, likelihood

$$p(y \mid x) = \frac{p(y) \cdot P(x \mid y)}{p(x)}$$

our predicted label is the one with the highest posterior probability, i.e.,

$$\hat{y} = \arg\max_{y \in Y} p(y) \cdot P(x \mid y)$$

class conditional probabilities

Bayes rule (ex: *x* = sentence, *y* = label in {pos, neg})

prior likelihood

posterior

$$p(y \mid x) = \frac{p(y) \cdot P(x \mid y)}{p(x)}$$

our predicted label is the one with the highest posterior probability, i.e.,

$$\hat{y} = \arg\max_{y \in Y} p(y) \cdot P(x \mid y)$$

what happened to the denominator???

remember the independence assumption!

maximum a posteriori (MAP) class

$$\hat{y} = \arg\max_{y \in Y} p(y) \cdot P(x \mid y)$$

$$= \arg\max_{y \in Y} p(y) \cdot \prod_{w \in x} P(w \mid y)$$

$$= \arg\max_{y \in Y} \log p(y) + \sum_{w \in x} \log P(w \mid y)$$

computing the prior…

- i hate the movie
- i love the movie
- i hate the actor
- the movie i love
- i love love love love love the movie
- hate movie
- i hate the actor i love the movie

$p(y)$ lets us encode inductive bias about the labels

we can estimate it from the data by simply counting…

label y	count	p(Y=y)	log(p(Y=y))
positive	3	0.43	-0.84
negative	4	0.57	-0.56

computing the likelihood…

p(X | y=positive)

word	count	p(w I y)
i	3	0.19
hate	0	0.00
love	7	0.44
the	3	0.19
movie	3	0.19
actor	0	0.00
total	**16**	

p(X | y=negative)

word	count	p(w I y)
i	4	0.22
hate	4	0.22
love	1	0.06
the	4	0.22
movie	3	0.17
actor	2	0.11
total	**18**	

$p(X \mid y=\text{positive})$

word	count	p(w l y)
i	3	0.19
hate	0	0.00
love	7	0.44
the	3	0.19
movie	3	0.19
actor	0	0.00
total	**16**	

$p(X \mid y=\text{negative})$

word	count	p(w l y)
i	4	0.22
hate	4	0.22
love	1	0.06
the	4	0.22
movie	3	0.17
actor	2	0.11
total	**18**	

new review X_{new}: love love the movie

$$\log p(X_{\text{new}} \mid \text{positive}) = \sum_{w \in X_{\text{new}}} \log p(w \mid \text{positive}) = -4.96$$

$$\log p(X_{\text{new}} \mid \text{negative}) = -8.91$$

posterior probs for X_{new}

$$p(y \mid x) \propto \arg\max_{y \in Y} p(y) \cdot P(\mathrm{X}_{\text{new}} \mid y)$$

$$\log p(\text{positive} \mid X_{\text{new}}) \propto \log P(\text{positive}) + \log p(X_{\text{new}} \mid \text{positive})$$
$$= -0.84 - 4.96 = -5.80$$

$$\log p(\text{negative} \mid X_{\text{new}}) \propto -0.56 - 8.91 = -9.47$$

naive Bayes predicts a positive label!

what if we see no positive training documents containing the word “awesome”?

$$p(\text{awesome} \mid \text{positive}) = 0$$

any review that contains “awesome” will have zero probability for the positive class!

Add-1 (Laplace) smoothing

$$\text{unsmoothed } P(w_i \mid y) = \frac{\text{count}(w_i, y)}{\sum_{w \in V} \text{count}(w, y)}$$

$$\text{smoothed } P(w_i \mid y) = \frac{\text{count}(w_i, y) + 1}{\sum_{w \in V} \text{count}(w, y) + |V|}$$

what happens if we do
add-*n* smoothing as *n* increases?

exercise!