text classification with naive Bayes

CS 490A, Fall 2020

Applications of Natural Language Processing

https://people.cs.umass.edu/~brenocon/cs490a_f20/

Brendan O'Connor

College of Information and Computer Sciences

University of Massachusetts Amherst

- Thanks for your exercises!
    - Type/token ratio def'n $= \frac{\text{Num Tokens}}{\text{Num Types}}$
- HW1 released today: Naive Bayes text classification!
    - Due Friday, 9/18
- Python tutorial in Tomas' OH
    - Wed 11:15am to 12:45pm
    - See Piazza "logistics" post, as always
- Schedule: https://people.cs.umass.edu/~brenocon/cs490a_f20/schedule.html

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**

$f(x) \mapsto y$

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!

incoming lead classif - from call transcripts

detect hate speech

artist pred. from lyrics

acad. honesty

grammar + spelling

text -> case mismatch?

lang. detection

text $\mapsto$ major world lang.

author's native pred.

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

$f(\mathbf{x}) = \textbf{spam}$

f can be hand-designed rules

- if "won $10,000,000" in **x**, **y** = **spam**
- if "CS490A Fall 2020" in **x**, **y** = **not spam**

what are the drawbacks of this method?

Time consuming

User-specific?

Getting details right is hard! (conjugations)

f can be **learned** from data

Train Data

(x, y)

(x, y)

$\xRightarrow{\text{Learn}}$ $f(x) \mapsto y$

- 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!	

or test set

heldout data: for evaluation

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

$\hat{y} = f(x)$ Then: $\hat{y} \stackrel{?}{=} y$

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

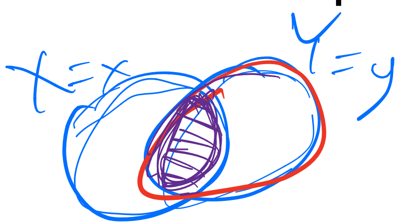
learn mapping function on training data, measure its accuracy on heldout data

probability review

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

$P((X=x) \wedge (Y=y))$

- conditional probability: $p(X = x \mid Y = y)$



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

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

Indep! $\Rightarrow P(x \mid y) = P(x)$

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(W1 = w1, W2 = w2, ...)
    p(the cat sleeps) > p(cat sleeps the)
  - $w_i \in V$ where V is the **vocabulary** (*types*)
    set of word 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)?

n tokens

$$p(w_1, w_2, w_3, \ldots, w_n)$$ V sized vocab

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

$\boxed{\text{how many } parameters \text{ would this require?}}$

V^n possible sentences

w_1 w_2 w_3

$p(w_1, w_2 \ldots w_n) = \prod_{i=1}^{n} p(w_i | w_1 \ldots$

chain rule

condition on doc class

$$p(w_1, w_2, w_3, \ldots w_n \mid y=k)$$

$$= p(w_1 \mid y=k)\ \cancel{p(w_2 , w_1 \mid y=k)}\ \cancel{p(w_3 \mid w_1, w_2)} \ldots$$

$$p(w_2 \mid w_1, y=k)\ p(w_3 \mid w_1, w_2, y=k) \ldots$$

naive Bayes' conditional indepedence assumption:

$w_i \perp w_j \mid y=k$

the probability of generating a word is independent of all other words (conditional on doc class)

How many params?

$$= \prod_{i=1}^{n} p(w_i \mid y=k)$$

V (per cat)

this is called the **unigram probability.**

what are its limitations?

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

Want: $P(w \mid k) \quad \forall w, k$

e.g. $P(W = \text{"hate"} \mid y = \text{POS}) = 0$

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 the love movie hate

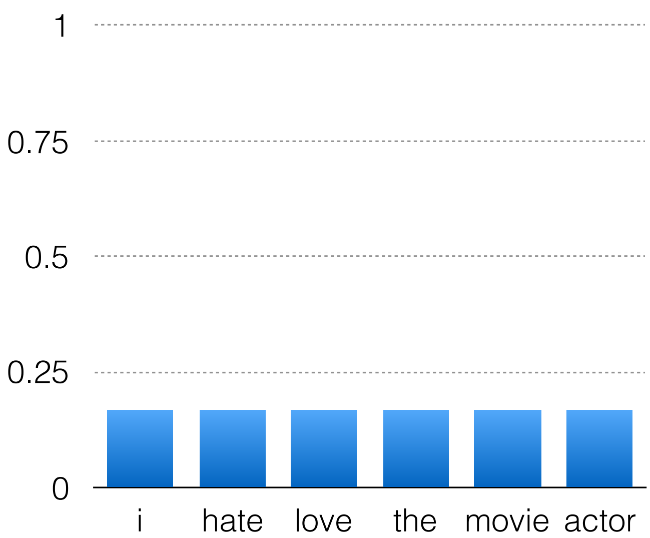
naive Bayes

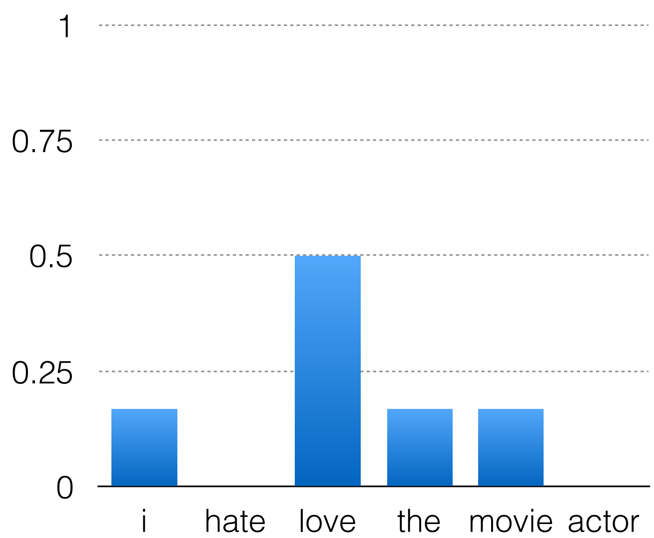
- represents input text as a bag of words
- assumption: each word is independent of all other words
- given labeled data, we can use naive Bayes to estimate probabilities for unlabeled data
- **goal:** infer probability distribution that generated the labeled data for each label

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

p(w|k')

p(w|k)





… back to our reviews

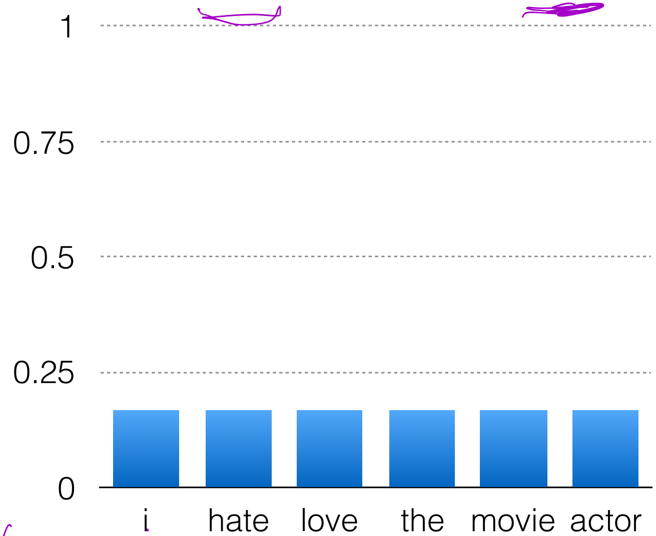
$$p(\text{i love love love love love the movie})$$

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

Rand dist.
more likely!

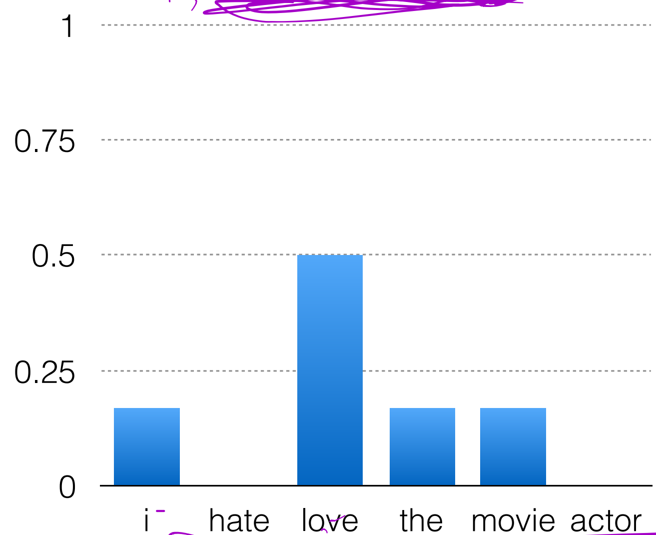
= .000000595

= 5.95374181e-7



.00014

= 1.4467592e-4



Stopped here 9/8

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*

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

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

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

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

class conditional probabilities

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

$$\underbrace{p(y \mid x)}_{\text{posterior}} = \frac{\overbrace{p(y)}^{\text{prior}} \cdot \overbrace{P(x \mid y)}^{\text{likelihood}}}{p(x)}$$

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 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 | y=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**	

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**	

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}

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

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

What does NB predict?

word likelihood ratios

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

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

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-α smoothing as α increases?