

Copyright Notice

These slides are distributed under the Creative Commons License.

[DeepLearning.AI](#) makes these slides available for educational purposes. You may not use or distribute these slides for commercial purposes. You may make copies of these slides and use or distribute them for educational purposes as long as you cite [DeepLearning.AI](#) as the source of the slides.

For the rest of the details of the license, see <https://creativecommons.org/licenses/by-sa/2.0/legalcode>



deeplearning.ai

Probability and Bayes' Rule

Outline

- Probabilities
- Bayes' rule (Applied in different fields, including NLP)
- Build your own Naive-Bayes tweet classifier!

Introduction

Corpus of tweets

Positive

Negative

Tweets containing the word
“happy”

Positive

“happy”

Negative

Probabilities

Corpus of tweets

Positive

Negative

$A \rightarrow$ Positive tweet

$$P(A) = P(\text{Positive}) = N_{\text{pos}} / N$$

Probabilities

Corpus of tweets

Positive

Negative

$A \rightarrow$ Positive tweet

$$P(A) = N_{\text{pos}} / N = 13 / 20 = 0.65$$

$$P(\text{Negative}) = 1 - P(\text{Positive}) = 0.35$$

Probabilities

Tweets containing the word
“happy”

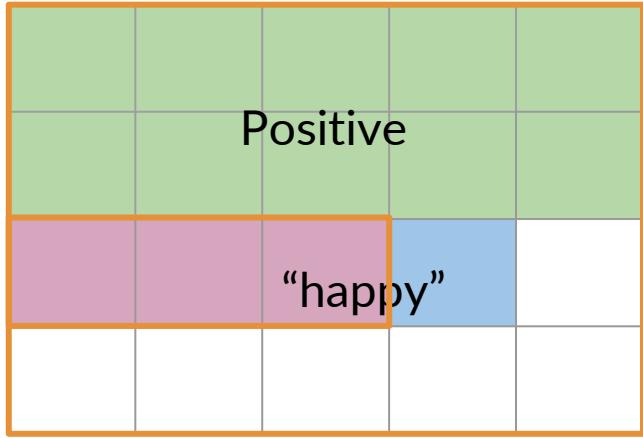
The table consists of 5 columns and 5 rows. The third column contains the word "happy" in yellow text.

$B \rightarrow$ tweet contains “happy”.

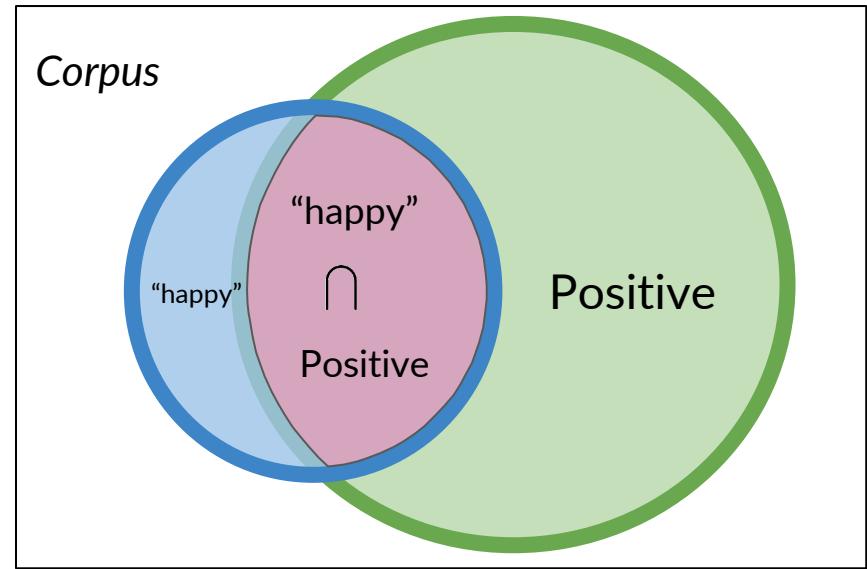
$$P(B) = P(\text{happy}) = N_{\text{happy}} / N$$

$$P(B) = 4 / 20 = 0.2$$

Probability of the intersection



$$P(A \cap B) = P(A, B) = \frac{3}{20} = 0.15$$

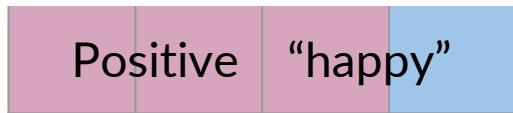




deeplearning.ai

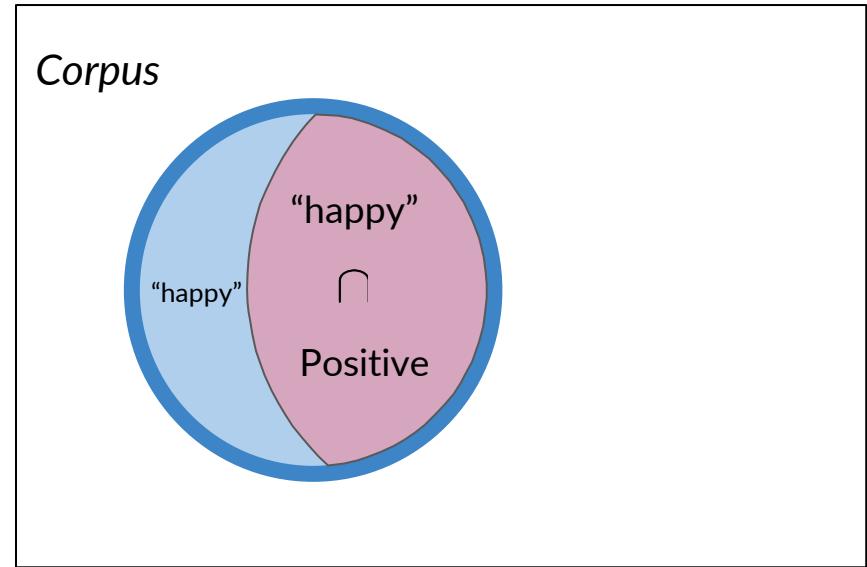
Bayes' Rule

Conditional Probabilities



$$P(A | B) = P(\text{Positive} | \text{"happy"})$$

$$P(A | B) = 3 / 4 = 0.75$$

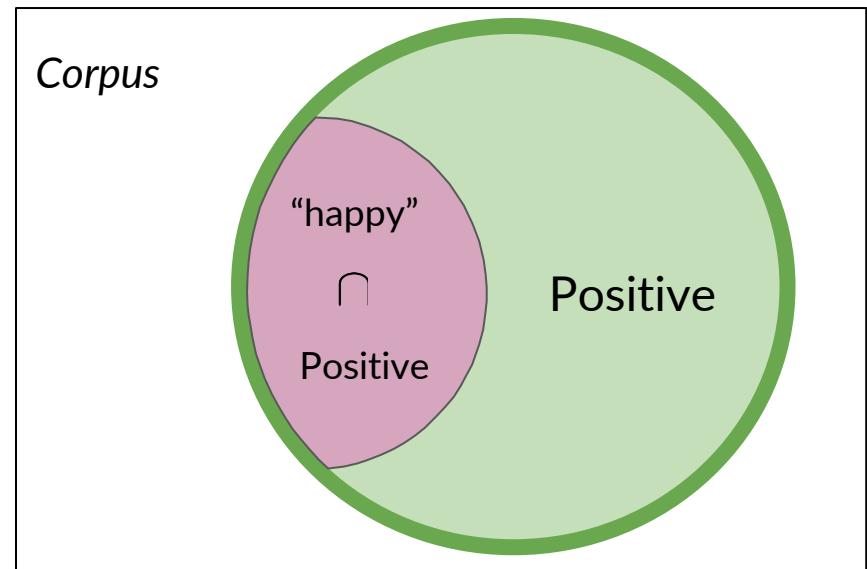


Conditional Probabilities

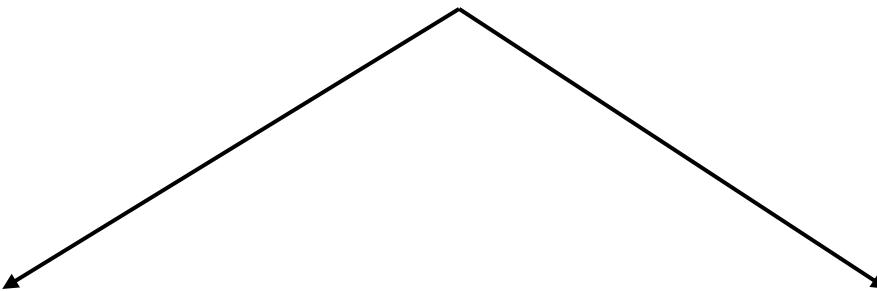
				Positive
“happy”				

$$P(B | A) = P(\text{“happy”} | \text{Positive})$$

$$P(B | A) = 3 / 13 = 0.231$$



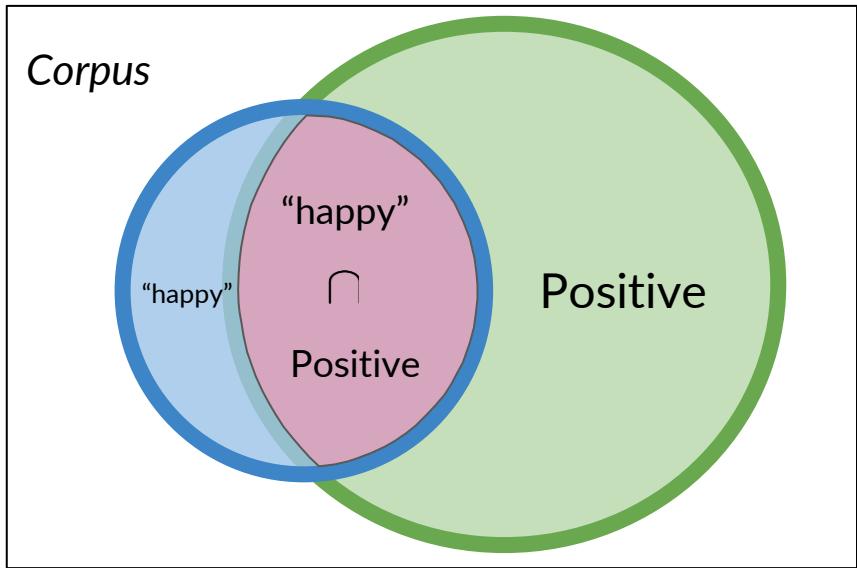
Conditional probabilities



Probability of B, given A happened

Looking at the elements of set A,
the chance that one also belongs to
set B

Conditional probabilities



$$P(\text{Positive} | \text{“happy”}) = \frac{P(\text{Positive} \cap \text{“happy”})}{P(\text{“happy”})}$$

Bayes' rule

$$P(\text{Positive} | \text{"happy"}) = \frac{P(\text{Positive} \cap \text{"happy"})}{P(\text{"happy"})}$$

$$P(\text{"happy"} | \text{Positive}) = \frac{P(\text{"happy"} \cap \text{Positive})}{P(\text{Positive})}$$

Quiz

Objective: Derive Bayes' rule from the equations given on the last slide.

Question:

From the equations presented below, express the probability of a tweet being positive given that it contains the word happy in terms of the probability of a tweet containing the word happy given that it is positive

$$P(\text{Positive}|\text{"happy"}) = \frac{P(\text{Positive} \cap \text{"happy"})}{P(\text{"happy"})} \quad P(\text{"happy"}|\text{Positive}) = \frac{P(\text{"happy"} \cap \text{Positive})}{P(\text{Positive})}$$

Type: Multiple Choice, single answer

Options and solution:

$$P(\text{Positive}|\text{"happy"}) = P(\text{"happy"}|\text{Positive}) \times \frac{P(\text{Positive})}{P(\text{"happy"})}$$

That's right. You just derived Bayes' rule.

$$P(\text{Positive}|\text{"happy"}) = P(\text{"happy"}|\text{Positive}) \times \frac{P(\text{"happy"})}{P(\text{Positive})}$$

The ratio is upside-down in this equation.

$$P(\text{Positive}|\text{"happy"}) = P(\text{"happy"} \cap \text{Positive}) \times \frac{P(\text{Positive})}{P(\text{"happy"})}$$

Your result should not include any intersection probabilities.

$$P(\text{Positive}|\text{"happy"}) = P(\text{"happy"} \cap \text{Positive}) \times \frac{P(\text{"happy"})}{P(\text{Positive})}$$

Your result should not include any intersection probabilities.

Bayes' rule

$$P(\text{Positive} | \text{"happy"}) = P(\text{"happy"} | \text{Positive}) \times \frac{P(\text{Positive})}{P(\text{"happy"})}$$



Quiz: Bayes' Rule Applied

Objective: Compute conditional probability using Bayes Rule

Question:

Here, again, is Bayes' rule:

$$P(X|Y) = P(Y|X) \times \frac{P(X)}{P(Y)}$$

Suppose that in your dataset, 25% of the positive tweets contain the word 'happy'. You also know that a total of 13% of the tweets in your dataset contain the word 'happy', and that 40% of the total number of tweets are positive. You observe the tweet: "happy to learn NLP". What is the probability that this tweet is positive?

Type: Multiple Choice, single answer

Options and solution:

A: $P(\text{Positive} | \text{"happy"}) = 0.77$ That's right. You just applied Bayes' rule.

B: $P(\text{Positive} | \text{"happy"}) = 0.08$ Oops, looks like you might have the ratio of $P(X)$ and $P(Y)$ upside-down.

C: $P(\text{Positive} | \text{"happy"}) = 0.10$ Remember to calculate the ratio in the formula for Bayes' rule.

D: $P(\text{Positive} | \text{"happy"}) = 1.92$ Did you use the probability of a tweet being positive? Remember that a fractional probability must be between 0 and 1.

Summary

- Conditional probabilities —→ Bayes' Rule
- $P(X|Y) = P(Y|X) \times \frac{P(X)}{P(Y)}$



deeplearning.ai

Naïve Bayes Introduction

because we assume features are independent in this method.
it's naive because in reality that's not the case.

Naïve Bayes for Sentiment Analysis

Positive tweets

I am happy because I am learning NLP

I am happy, not sad.

Negative tweets

I am sad, I am not learning NLP

I am sad, not happy

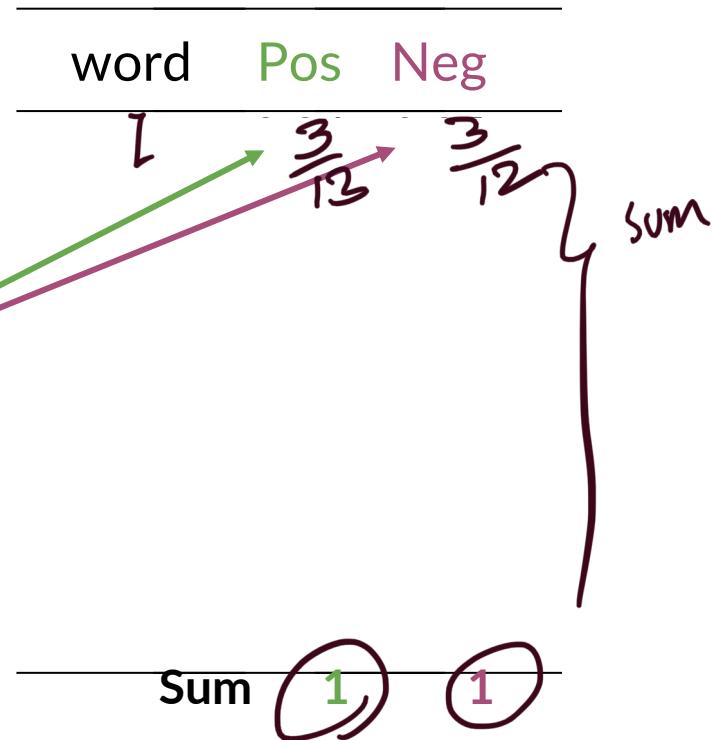
word	Pos	Neg
I	3	3
am	3	3
happy	2	1
because	1	0
learning	1	1
NLP	1	1
sad	1	2
not	1	2
N_{class}	13	12

needed to calculate Conditional probabilities

$P(w_i | \text{class})$

word	Pos	Neg
I	3	3
am	3	3
happy	2	1
because	1	0
learning	1	1
NLP	1	1
sad	1	2
not	1	2
Nclass	13	
		12

$$p(I|R_{eg}) = \frac{3}{13}$$



$P(w_i | \text{class})$

like a training
process

word	Pos	Neg
I	0.24	0.25
am	0.24	0.25
happy	0.15	0.08
because	0.08	0
learning	0.08	0.08
NLP	0.08	0.08
sad	0.08	0.08
not	0.08	0.17
Sum	1	1

Power words
diff probs

↳ neutral words (equal prob)

Naïve Bayes inference for binary classification

new test sample ↗

Tweet: I am happy today; I am learning.

$$\prod_{i=1}^m \frac{P(w_i|pos)}{P(w_i|neg)} = \frac{0.14}{0.10} = 1.4 > 1 \quad \text{pos neg}$$

$$\frac{0.20}{0.20} * \frac{0.20}{0.20} * \frac{0.14}{0.10} * \frac{0.20}{0.20} * \frac{0.20}{0.20} * \frac{0.10}{0.10}$$

I am happy | am learning (NO "Today")

word	Pos	Neg
I	0.20	0.20
am	0.20	0.20
happy	0.14	0.10
because	0.10	0.05
learning	0.10	0.10
NLP	0.10	0.10
sad	0.10	0.10
not	0.10	0.15

Summary

- Naive Bayes inference condition rule for binary classification
- Table of probabilities

$$\prod_{i=1}^m \frac{P(w_i|pos)}{P(w_i|neg)}$$



deeplearning.ai

Laplacian Smoothing

Laplacian Smoothing to avoid probs being zero

$$P(w_i|class) = \frac{\text{freq}(w_i, \text{class})}{N_{\text{class}}} \quad \begin{matrix} \text{freq of word} \\ \text{in corpus} \end{matrix} \quad \text{class} \in \{\text{Positive}, \text{Negative}\}$$

$$P(w_i|class) = \frac{\text{freq}(w_i, \text{class}) + 1}{N_{\text{class}} + V_{\text{class}}} \quad \begin{matrix} 1 \\ V_{\text{class}} \end{matrix}$$

so prob
won't be
zero

N_{class} = frequency of all words in class

V_{class} = number of unique words in class

Introducing $P(w_i | \text{class})$ with smoothing

word	Pos	Neg
I	3	3
am	3	3
happy	2	1
because	1	0
learning	1	1
NLP	1	1
sad	1	2
not	1	2
Nclass	13	
	12	

$Z: P(Z | \text{Pos}) = \frac{3+1}{13+8}$

$$P(Z | \text{Neg}) = \frac{3+1}{13+8}$$

Patition Smoothing

$$V = 8$$

8 words in
each class

word	Pos	Neg
...
...
...

Sum 1 1
 will still be one
 because at V

Summary

- Laplacian smoothing to avoid $P(w_i|class) = 0$
- Naïve Bayes formula

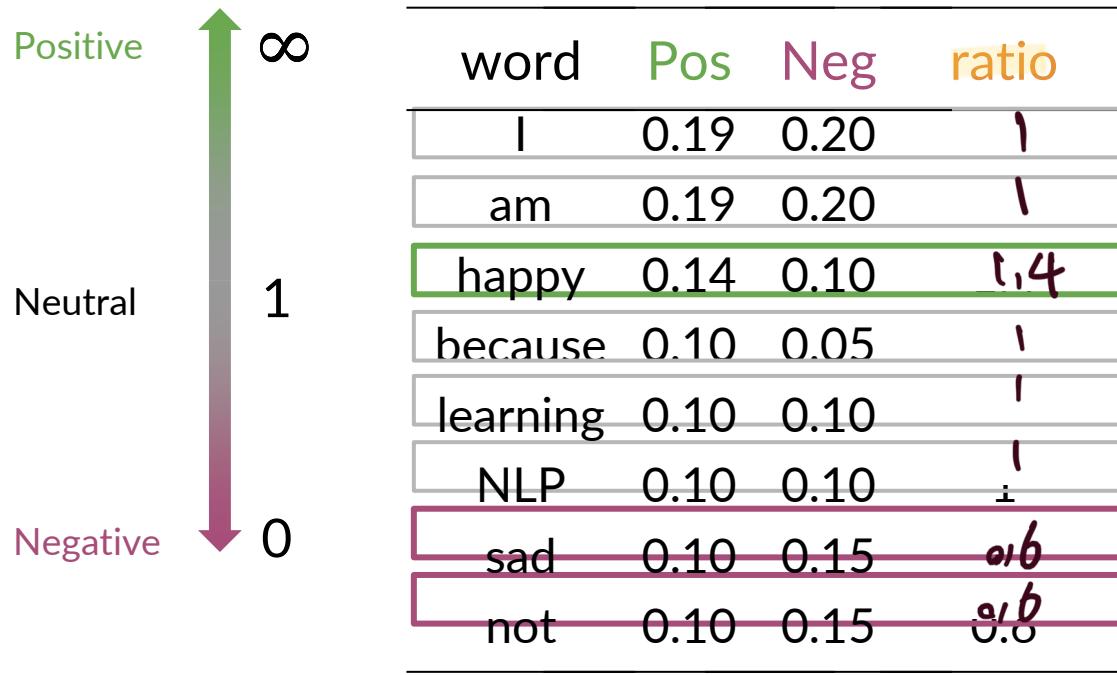
$$\prod_{i=1}^m \frac{P(w_i|pos)}{P(w_i|neg)}$$



deeplearning.ai

Log Likelihood, Part 1

Ratio of probabilities



larger ratio
→ more pos

$$\text{ratio}(w_i) = \frac{P(w_i | \text{Pos})}{P(w_i | \text{Neg})}$$

$$\approx \frac{\text{freq}(w_i, 1) + 1}{\text{freq}(w_i, 0) + 1}$$

Naïve Bayes' inference

$class \in \{pos, neg\}$

$w \rightarrow Set of m words in a tweet$

$$\frac{P(pos)}{P(neg)} \prod_{i=1}^m \frac{P(w_i|pos)}{P(w_i|neg)} > 1$$

- A simple, fast, and powerful baseline
- A probabilistic model used for classification

Log Likelihood (log of Naive Bayes)

$$\frac{P(pos)}{P(neg)} \prod_{i=1}^m \frac{P(w_i|pos)}{P(w_i|neg)}$$

- Products bring risk of underflow
- $\log(a * b) = \log(a) + \log(b)$

For computation stability
easier to deal with

$$\log\left(\frac{P(pos)}{P(neg)} \prod_{i=1}^n \frac{P(w_i|pos)}{P(w_i|neg)}\right) \Rightarrow$$

$$\log \frac{P(pos)}{P(neg)} + \sum_{i=1}^n \log \frac{P(w_i|pos)}{P(w_i|neg)}$$

log prior + log likelihood

$$\lambda(\text{happy}) = \log \frac{0.109}{0.01} \sim 2.2$$

Calculating Lambda

tweet: I am happy because I am learning.

$$\lambda(w) = \log \frac{P(w|pos)}{P(w|neg)}$$

$$\lambda(\text{am}) = \log \frac{0.04}{0.04} = \log(1) = 0$$

$$\lambda(\text{I}) = \log \frac{0.05}{0.05} = \log(1) = 0$$

word	Pos	Neg	λ
I	0.05	0.05	0 (neutral)
am	0.04	0.04	0
happy	0.09	0.01	2.2 (pos)
because	0.01	0.01	
learning	0.03	0.01	
NLP	0.02	0.02	
sad	0.01	0.09	-2.2 (neg)
not	0.02	0.03	-0.4 (neg)

Summing the Lambdas

doc: I am happy because I am learning.

$$\lambda(w) = \log \frac{P(w|pos)}{P(w|neg)}$$

$$\lambda(\text{happy}) = \log \frac{0.09}{0.01} \approx 2.2$$

word	Pos	Neg	λ
I	0.05	0.05	0
am	0.04	0.04	0
happy	0.09	0.01	0.4
because	0.01	0.01	
learning	0.03	0.01	
NLP	0.02	0.02	
sad	0.01	0.09	
not	0.02	0.03	-0.4

Summary

- Word sentiment

$$\left. \begin{array}{l} ratio(w) = \frac{P(w|pos)}{P(w|neg)} \\ \lambda(w) = \log \frac{P(w|pos)}{P(w|neg)} \end{array} \right\}$$



deeplearning.ai

Log Likelihood, Part 2

Log Likelihood

doc: I am happy because I am learning.

$$\sum_{i=1}^m \log \frac{P(w_i|pos)}{P(w_i|neg)} = \sum_{i=1}^m \lambda(w_i)$$

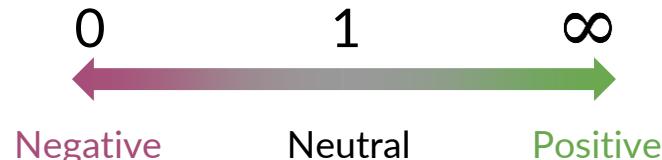
$$\text{log likelihood} = 0 + 0 + 2.2 + 0 + 0 + 0 + 1.1 = 3.3$$

I am happy . . .

word	Pos	Neg	λ
I	0.05	0.05	0
am	0.04	0.04	0
happy	0.09	0.01	2.2
because	0.01	0.01	0
learning	0.03	0.01	1.1
NLP	0.02	0.02	0
sad	0.01	0.09	-2.2
not	0.02	0.03	-0.4

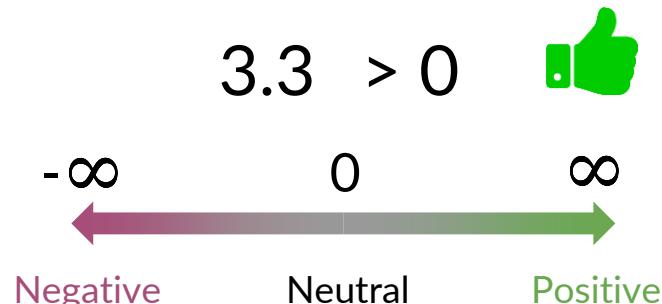
Log Likelihood

$$\prod_{i=1}^m \frac{P(w_i|pos)}{P(w_i|neg)} > 1$$



$$\sum_{i=1}^m \log \frac{P(w_i|pos)}{P(w_i|neg)}$$

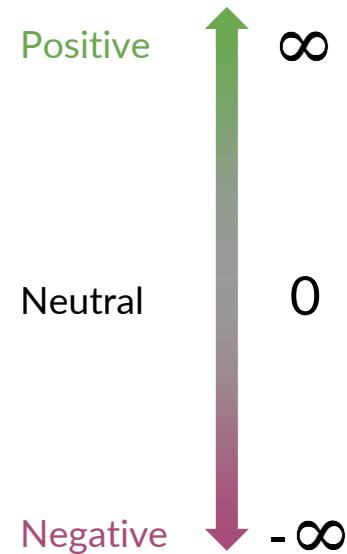
because at
log
> 0



Summary

Tweet sentiment:

$$\log \prod_{i=1}^m ratio(w_i) = \sum_{i=1}^m \lambda(w_i) > 0$$





deeplearning.ai

Training Naïve Bayes

Outline

- Five steps for training a Naïve Bayes model

Training Naïve Bayes

Step 0: Collect and annotate corpus

Positive tweets

I am happy because I am learning
NLP
I am happy, not sad. @NLP

Negative tweets

I am sad, I am not learning NLP
I am sad, not happy!!

Step 1:
Preprocess

- Lowercase
- Remove punctuation, urls, names
- Remove stop words
- Stemming
- Tokenize sentences

Positive tweets

[happi, because, learn, NLP]
[happi, not, sad]

Negative tweets

[sad, not, learn, NLP]
[sad, not, happi]

Training Naïve Bayes

freq(w, class)

Positive tweets

[happi, because, learn, NLP]

[happi, not, sad]

Negative tweets

[sad, not, learn, NLP]

[sad, not, happi]

Step 2:
Word
count

word	Pos	Neg
happi	2	1
because	1	0
learn	1	1
NLP	1	1
sad	1	2
not	1	2
N_{class}	7	7

Training Naïve Bayes

$\text{freq}(w, \text{class})$

word	Pos	Neg
happi	2	1
because	1	0
learn	1	1
NLP	1	1
sad	1	2
not	1	2
N_{class}	7	7

Step 3:
 $P(w|\text{class})$

$$V_{\text{class}} = 6$$

$$\frac{\text{freq}(w, \text{class}) + 1}{N_{\text{class}} + V_{\text{class}}}$$

with Laplace smoothing

$$\lambda(w) = \log \frac{P(w|\text{pos})}{P(w|\text{neg})}$$

Step 4:
 Get
 lambda

word	Pos	Neg	λ
happy	0.23	0.15	0.43
because	0.15	0.07	0.6
learning	0.08	0.08	0
NLP	0.08	0.08	0
sad	0.08	0.17	-0.75
not	0.08	0.17	-0.75



Training Naïve Bayes

Step 5:
Get the
log prior

D_{pos} = Number of positive tweets
 D_{neg} = Number of negative tweets

$$\text{logprior} = \log \frac{D_{pos}}{D_{neg}}$$

for naive
formula

If dataset is balanced, $D_{pos} = D_{neg}$ and $\text{logprior} = 0$.

Summary

1. Get or annotate a dataset with positive and negative tweets
2. Preprocess the tweets: $\text{process_tweet(tweet)} \rightarrow [w_1, w_2, w_3, \dots]$
3. Compute $\text{freq}(w, \text{class})$
4. Get $P(w | \text{pos})$, $P(w | \text{neg})$
5. Get $\lambda(w)$
6. Compute $\text{logprior} = \log(P(\text{pos}) / P(\text{neg}))$



deeplearning.ai

Testing Naïve Bayes

Outline

- Predict using a Näive Bayes Model
- Using your validation set to compute model accuracy

Predict using Naïve Bayes

- log-likelihood dictionary $\lambda(w) = \log \frac{P(w|pos)}{P(w|neg)}$

$$\text{logprior} = \log \frac{D_{pos}}{D_{neg}} = 0$$

- Tweet: [I, pass, the, NLP, interview]. 
 $\sqrt{\text{not seen}} \Rightarrow 0$

$$score = -0.01 + 0.5 - 0.01 + 0 + \text{logprior} = 0.48$$

$pred = score > 0$ DaS

word	λ
I	-0.01
the	-0.01
happi	0.63
because	0.01
pass	0.5
NLP	0
sad	-0.75
not	-0.75

Testing Naïve Bayes

- $X_{val} \ Y_{val} \ \lambda \ logprior$

$score = predict(X_{val}, \lambda, logprior)$

$pred = score > 0$

$$\begin{bmatrix} 0.5 \\ -1 \\ 1.3 \\ \vdots \\ score_m \end{bmatrix} > 0 = \begin{bmatrix} 0.5 > 0 \\ -1 > 0 \\ 1.3 > 0 \\ \vdots \\ score_m > 0 \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \\ 1 \\ \vdots \\ pred_m \end{bmatrix}$$

Testing Naïve Bayes

- X_{val} Y_{val} λ logprior

$score = predict(X_{val}, \lambda, logprior)$

$pred = score > 0$

$$\frac{1}{m} \sum_{i=1}^m (pred_i == Y_{val_i})$$

$$\begin{bmatrix} 0 \\ 1 \\ 1 \\ \vdots \\ pred_m \end{bmatrix} == \begin{bmatrix} 0 \\ 0 \\ 1 \\ \vdots \\ Y_{val_m} \end{bmatrix}$$

$$\begin{bmatrix} 1 \\ 0 \\ 1 \\ \vdots \\ pred_m == Y_{val_m} \end{bmatrix}$$

Summary

- $X_{val} \ Y_{val} \longrightarrow$ Performance on unseen data
- Predict using λ and *logprior* for each new tweet
- Accuracy $\longrightarrow \frac{1}{m} \sum_{i=1}^m (pred_i == Y_{val_i})$
- What about words that do not appear in $\lambda(w)$?



deeplearning.ai

Applications of Naïve Bayes

Applications of Naïve Bayes

$$P(pos|tweet) \approx P(pos)P(tweet|pos)$$

$$P(neg|tweet) \approx P(neg)P(tweet|neg)$$

$$\frac{P(pos|tweet)}{P(neg|tweet)} = \frac{P(pos)}{P(neg)} \prod_{i=1}^m \frac{P(w_i|pos)}{P(w_i|neg)}$$

Applications of Naïve Bayes

Author identification:

$$\frac{P(\text{Shakespeare} \mid \text{book})}{P(\text{Dostoevsky} \mid \text{book})}$$

Spam filtering:

$$\frac{P(\text{spam} \mid \text{email})}{P(\text{nonspam} \mid \text{email})}$$

Applications of Naïve Bayes

Information retrieval:

$$P(\text{document}_k | \text{query}) \propto \prod_{i=0}^{|\text{query}|} P(\text{query}_i | \text{document}_k)$$

Retrieve document if $P(\text{document}_k | \text{query}) > \text{threshold}$

"Icon made by [Vector Market](#) from [www.flaticon.com](#)"

Applications of Naïve Bayes

Word disambiguation:

$$\frac{P(\text{river}|\text{text})}{P(\text{money}|\text{text})}$$

Bank:



["Pictures with CC"](#)

Naïve Bayes Applications

- Sentiment analysis
- Author identification
- Information retrieval
- Word disambiguation
- Simple, fast and robust!



deeplearning.ai

Naïve Bayes Assumptions

Outline

- Independence
- Relative frequency in corpus

Naïve Bayes Assumptions

- Independence → it's so naive that because of this we call this method naive
“It is sunny and hot in the Sahara desert.”



Naïve Bayes Assumptions

“It’s always cold and snowy in ____.”



spring?? summer? fall?? winter??

Naïve Bayes may give
equal probs to all 4 seasons

Naïve Bayes Assumptions

- Relative frequencies in corpus

*needs to be balanced
& it's usually not*



Summary

- Independence: Not true in NLP
- Relative frequency of classes affect the model



deeplearning.ai

Error Analysis

Outline

- Removing punctuation and stop words
- Word order
- Adversarial attacks

Processing as a Source of Errors: Punctuation

Tweet: My beloved grandmotherX(

processed_tweet: [belov, grandmoth]

Processing as a Source of Errors: Removing Words

Tweet: This is not good, because your attitude is not even close to being nice.

processed_tweet: [good, attitude, close, nice]

Processing as a Source of Errors: Word Order

Tweet: I am happy because I do not go.



Tweet: I am not happy because I did go.



Adversarial attacks

Sarcasm, Irony and Euphemisms

Tweet: This is a ridiculously powerful movie. The plot was gripping and I cried right through until the ending!

processed_tweet: [ridicul, power, movi, plot, grip, cry, end]

Summary

- Removing punctuation
- Removing words
- Word order
- Adversarial attacks

$$\frac{25}{100} \times \frac{100 \times 40}{13 \times 100}$$