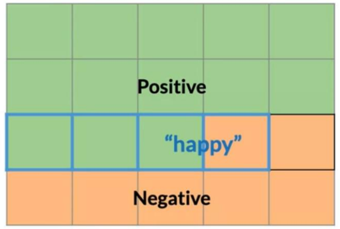
Outline

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

Introduction

Corpus of tweets

Tweets containing the word "happy"



Probabilities

Corpus of tweets

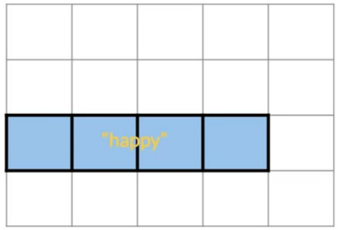
A → Positive tweet

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

$P(Negative) = 1 - P(Positive) = 0.35$

Probabilities

Tweets containing the word "happy"

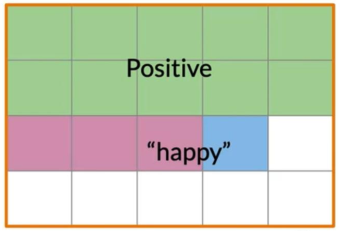


B → tweet contains "happy".

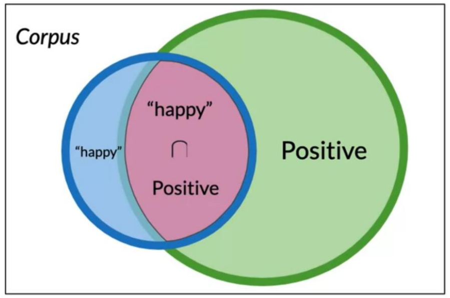
$P(B) = P(happy) = N_{happy} / N$

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

Probability of the intersection



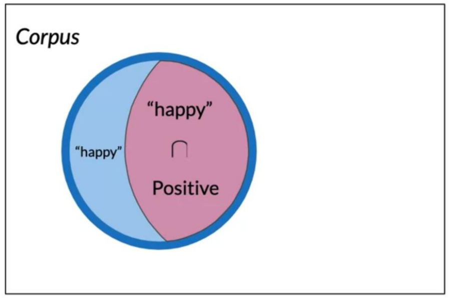
$$P(A \cap B) = P(A, B) = \frac{3}{20}$$



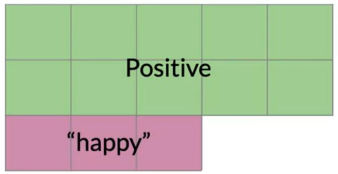
Conditional Probabilities

P(A | B) = P(Positive | "happy")

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

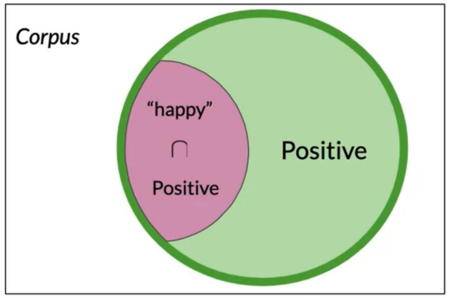


Conditional Probabilities



P(B | A) = P("happy"| Positive)

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

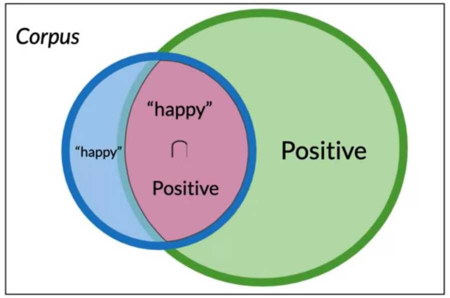


Conditional probabilities

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

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

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

Bayes' rule

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

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

Summary

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

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

$P(w_i \mid class)$

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

word	Pos	Neg
I	0.24	0.25
am	0.24	0.25
happy	0.15	0.08
because	0.08	0.00
learning	0.08	0.08
NLP	0.08	0.08
sad	0.08	0.17
not	0.08	0.17

$P(w_i \mid class)$

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	0.17
not	0.08	0.17

Naïve Bayes

Tweet: I am happy today; I am learning.

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

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

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	0.15
not	0.10	0.15

Summary

- Naive Bayes inference condition rule for binary classification $\prod_{i=1}^{m} \frac{P(w_i|pos)}{P(w_i|neg)}$
- Table of probabilities

deeplearning.ai

Laplacian Smoothing

$$P(\mathrm{w_i}|\mathrm{class}) = \frac{\mathrm{freq}(\mathrm{w_i}, \mathrm{class})}{\mathrm{N_{class}}}$$

class ∈ {Positive, Negative}

$$P(\mathrm{w_i}|\mathrm{class}) = \frac{\mathrm{freq}(\mathrm{w_i}, \mathrm{class}) + 1}{\mathrm{N_{class}} + \mathrm{V}}$$

N_{class} = frequency of all words in class

V = number of unique words in vocabulary

deeplearning.ai

Introducing $P(w_i \mid class)$ 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**

V = 8

word	Pos	Neg
I	0.19	0.20
am	0.19	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	0.15
not	0.10	0.15
Sum	**1**	**1**

deeplearning.ai

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

Ratio of probabilities

Positive ∞

Neutral 1

Negative 0

word	Pos	Neg	ratio
I	0.20	0.20	1
am	0.20	0.20	1
happy	0.14	0.10	1.4
because	0.10	0.10	1
learning	0.10	0.10	1
NLP	0.10	0.10	1
sad	0.10	0.15	0.6
not	0.10	0.15	0.6

$$\text{ratio}(w_i) = \frac{P(w_i \mid \text{Pos})}{P(w_i \mid \text{Neg})} \approx \frac{freq(w_i, 1) + 1}{freq(w_i, 0) + 1}$$

Naïve Bayes' inference

class ∈ {pos, neg}
w *-> Set of m words in a tweet*

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

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

Log Likelihood

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

- Products bring risk of underflow
- $log(a * b) = log(a) + log(b)$
- $log(\frac{P(pos)}{P(neg)} \prod_{i=1}^{n} \frac{P(w_i|pos)}{P(w_i|neg)}) \Rightarrow log\frac{P(pos)}{P(neg)} + \sum_{i=1}^{n} log\frac{P(w_i|pos)}{P(w_i|neg)}$

log prior +　log likelihood

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	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

Summary

- Word sentiment

$$\begin{cases} ratio(w) = \dfrac{P(w|pos)}{P(w|neg)} \\[2ex] \lambda(w) = log\dfrac{P(w|pos)}{P(w|neg)} \end{cases}$$

Log Likelihood

doc: I am happy because I am learning.

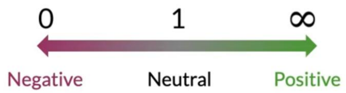
$$\sum_{i=1}^{m} log\frac{P(w_i|pos)}{P(w_i|neg)} = \sum_{i=1}^{m} \lambda(w_i)$$

log likelihood = 0 + 0 + 2.2 + 0 + 0 + 0 + 1.1 = **3.3**

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

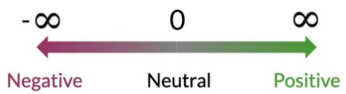
Log Likelihood

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



3.3 > 0

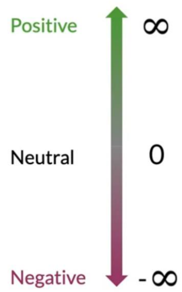
$$\sum_{i=1}^{m} log\frac{P(w_i|pos)}{P(w_i|neg)} > 0$$



Summary

Tweet sentiment:

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



Outline

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

Predict using Naïve Bayes

- log-likelihood dictionary $\lambda(w) = log\frac{P(w|pos)}{P(w|neg)}$
- $logprior = log\frac{D_{pos}}{D_{neg}} = 0$
- Tweet: I passed the NLP interview.

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

Predict using Naïve Bayes

- log-likelihood dictionary $\lambda(w) = log\frac{P(w|pos)}{P(w|neg)}$
- $logprior = log\frac{D_{pos}}{D_{neg}} = 0$
- Tweet: [I, pass, the, NLP, interview]

$$score = -0.01 + 0.5 - 0.01 + 0 + logprior = 0.48$$

$$pred = score > 0$$

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} λ $logprior$

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

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

Testing Naïve Bayes

- X_{val} Y_{val} λ $logprior$

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

$pred = score > 0$

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

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

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 λ(w)?

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:

$P($  $|\text{book})$ / $P($  $|\text{book})$

Spam filtering:

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

Applications of Naïve Bayes

Information retrieval:

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

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

Applications of Naïve Bayes

Word disambiguation:

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

Bank:

Naïve Bayes Applications

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

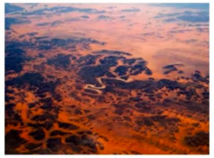
Outline

- Independence
- Relative frequency in corpus

Naïve Bayes Assumptions

- Independence

"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 Assumptions

- Relative frequencies in corpus

Summary

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

Outline

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

Processing as a Source of Errors: Punctuation

Tweet: My beloved grandmother

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 did 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]