





MPEI 2024-2025

Probabilistic Methods I

Naïve Bayes Classifier

Positive or negative **movie review**?

-  • unbelievably disappointing
-  • Full of zany characters and richly applied satire, and some great plot twists
-  • this is the greatest screwball comedy ever filmed
-  • It was pathetic. The worst part about it was the boxing scenes.

Is this spam?

Good morning Dan,

Please familiarize yourself with the attached file.
Reply here if you have any questions.

Thank you.

John and Mike,

Appreciate your flexibility this week, as the team navigates the sensitivities surrounding some of the project work taking place at the sites. Please tentatively plan for mobilization on 05/16/2022, in order to begin the final stages of the upgrade.

I will follow-up tomorrow with a confirmation if all indications are we will be given the “all-clear” before EOB Wednesday/SOB Thursday.

Appreciate your support.

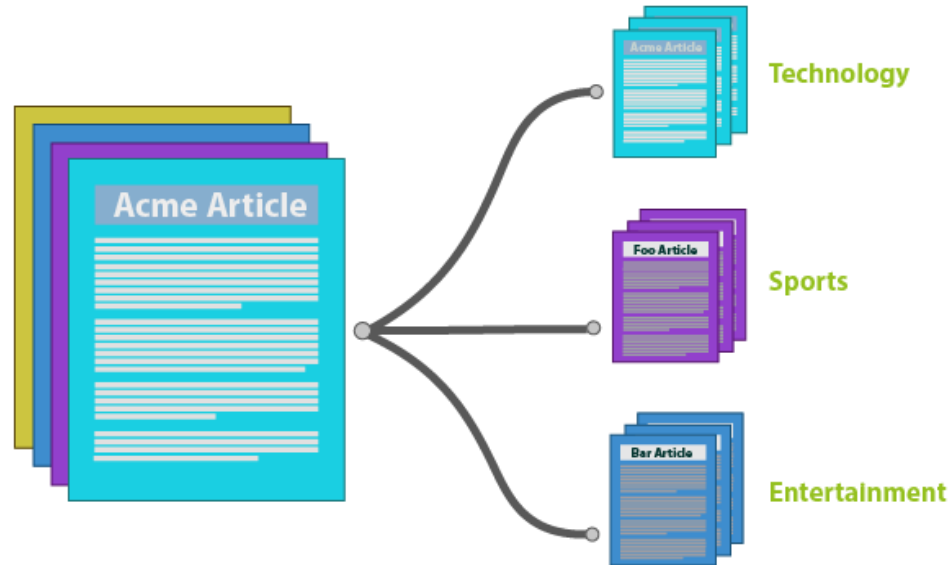
Regards,

Judy Sewell
Project Manager

Text Classification

- Assigning subject categories, topics, or genres
- Spam detection
- Authorship identification (who wrote this?)
- Language Identification (is this Portuguese?)
- Sentiment analysis
- ...

Text Classification: definition



- *Input:*
 - a document d
 - a fixed set of classes $C = \{c_1, c_2, \dots, c_J\}$
- *Output:* a predicted class $c \in C$

Naïve Bayes Intuition

- Simple ("naïve") classification method **based on Bayes rule**
- Relies on **very simple representation of document**
 - Bag of words

The Bag of Words 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
...	...

Bayes' Rule Applied to Documents and Classes

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

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

Naïve Bayes Classifier (I)

- Chooses the **class with maximum a posteriori probability** (MAP)

(the most likely class)

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

Applying Bayes Rule →

$$= \operatorname{argmax}_{c \in C} \frac{P(d | c)P(c)}{P(d)}$$

Dropping the denominator →

$$= \operatorname{argmax}_{c \in C} P(d | c)P(c)$$

Naive Bayes Classifier (II)

"Likelihood"

"Prior"

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

Considering **document d represented by features x1..xn**

$$= \operatorname{argmax}_{c \in C} P(x_1, x_2, \dots, x_n | c)P(c)$$

Likelihood translates to “probabilidade” ou “verossimilhança” in Portuguese

Naïve Bayes Classifier (IV)

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

How often does this class occur?

We can just count the relative frequencies in a corpus

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

Assumptions / Simplifications

- $P(X_1, X_2, \dots, X_n | c)$?
- Bag of Words assumption: Assume position doesn't matter
- Conditional Independence: Assume the feature probabilities $P(X_i | c_j)$ are independent given the class.

$$P(X_1, X_2, \dots, X_n | c) = P(X_1 | c) \times P(X_2 | c) \times P(X_3 | c) \times \dots \times P(X_n | c)$$

Naïve Bayes Classifier

- The initial MAP classifier

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

- Becomes the Naïve Bayes (NB) classifier

$$c_{NB} = \underset{c \in C}{\operatorname{argmax}} P(c_j) \prod_{x \in X} P(x | c)$$

Applying to Text Classification

- Apply to all the words in test document

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

- positions \leftarrow all word positions in **test document**

Problem with multiplying probs

- There's a problem with this formula:

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

- We end up with multiplications such as:
 $.0006 * .0007 * .0009 * .01 * .5 * .000008....$
- Multiplying lots of probabilities can result in floating-point underflow!
- Solution: Sum logs of probabilities** instead of multiplying probabilities!
 - because $\log(ab) = \log(a) + \log(b)$

Do everything in log space

- In fact, everything is done in log space, using:

$$c_{\text{NB}} = \operatorname{argmax}_{c_j \in C} \left[\log P(c_j) + \sum_{i \in \text{positions}} \log P(x_i | c_j) \right]$$

- Taking **log doesn't change the ranking of classes!**
 - The class with highest probability also has highest log probability!
- It's a linear model:
 - **Just a max of a sum of weights:** a linear function of the inputs
 - So **Naïve Bayes is a linear classifier**

Learning the Model

Sec13.3

- First attempt: Simply use the frequencies in the data
(maximum likelihood estimates)

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

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

V = Vocabulary

c = Class

w = Word

Parameter estimation

- Create mega-document for class j by concatenating all docs in this class
- Use frequency of w in mega-document

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

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

Problem with this approach

- What if we have seen no training documents with the word **fantastic** and classified in the class positive (thumbs-up)?

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

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

$$\prod_i \hat{P}(X_i \mid c) \text{ will be zero}$$

Simple solution

- Laplace (add-1) smoothing for NB

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

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

Naïve Bayes: Learning

1. Calculate $P(c_j)$ terms

$$P(c_j) \leftarrow \frac{|docs_j|}{|\text{total \# documents}|}$$

For each c_j in C do

$docs_j \leftarrow$ all docs with class = c_j

2. From training corpus, extract *Vocabulary*

3. Calculate $P(w_k | c_j)$ terms

$$P(w_k | c_j) \leftarrow \frac{n_k + a}{n + a |Vocabulary|}$$

$Text_j \leftarrow$ single doc containing all $docs_j$

For each word w_k in *Vocabulary*

$n_k \leftarrow$ # of occurrences of w_k in $Text_j$

Unknown words

- What about unknown words
 - that appear in our test data
 - but not in our training data or vocabulary?
- We ignore them
 - Remove them from the test document!
 - Pretend they weren't there!
 - Don't include any probability for them at all!

Stop words

- Stop words: very frequent words like the and a.
- Some systems ignore stop words
 - Sort the vocabulary by word frequency in training set
 - Call the top 10 or 50 words the stop word list.
 - Remove all stop words from both training and test sets
 - As if they were never there!
- But removing stop words doesn't usually help
 - in practice most NB algorithms use all words and don't use stop word lists



Simple example

	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

Worked example with add-1 smoothing

1. Prior from training:

$$\hat{P}(c_j) = \frac{N_{c_j}}{N_{total}} \quad \begin{array}{l} P(-) = 3/5 \\ P(+) = 2/5 \end{array}$$

2. Drop "with"

3. Likelihoods from training:

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

$$P(\text{"predictable"}|-) = \frac{1+1}{14+20} \quad P(\text{"predictable"}|+) = \frac{0+1}{9+20}$$

$$P(\text{"no"}|-) = \frac{1+1}{14+20} \quad P(\text{"no"}|+) = \frac{0+1}{9+20}$$

$$P(\text{"fun"}|-) = \frac{0+1}{14+20} \quad P(\text{"fun"}|+) = \frac{1+1}{9+20}$$

	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

14 words for -
9 word for +

4. Scoring the test set:

$$P(-)P(S|-) = \frac{3}{5} \times \frac{2 \times 2 \times 1}{34^3} = 6.1 \times 10^{-5}$$

$$P(+)P(S|+) = \frac{2}{5} \times \frac{1 \times 1 \times 2}{29^3} = 3.2 \times 10^{-5}$$



Binary NB

- For tasks like sentiment, word occurrence seems to be more important than word frequency.
 - The occurrence of the word fantastic tells us a lot
 - The fact that it occurs 5 times may not tell us much more.
- Binary multinominal Naïve Bayes, or Binary NB
 - Clips our word counts at 1

Binary Naïve Bayes: Learning

1. Calculate $P(c_j)$ terms

$$P(c_j) \leftarrow \frac{|docs_j|}{|\text{total \# documents}|}$$

For each c_j in C do

$docs_j \leftarrow$ all docs with class = c_j

2. From training corpus, extract *Vocabulary*

3. Calculate $P(w_k | c_j)$ terms

$$P(w_k | c_j) \leftarrow \frac{n_k + \alpha}{n + \alpha |Vocabulary|}$$

Remove duplicates in each doc:

For each word type w in doc_j

Retain only a single instance of w

$Text_j \leftarrow$ single doc containing all $docs_j$

For each word w_k in *Vocabulary*

$n_k \leftarrow$ # of occurrences of w_k in $Text_j$

Applying to a test document

- First **remove all duplicate words** from d
- Then compute NB using the same equation:

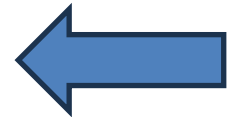
$$c_{NB} = \underset{c_j \in C}{\operatorname{argmax}} P(c_j) \prod_{i \in \text{positions}} P(w_i | c_j)$$

Example of Binary NB

Four original documents:

- it was pathetic the worst part was the boxing scenes
- no plot twists or great scenes
- + and satire and great plot twists
- + great scenes great film

	NB Counts	
	+	–
and	2	0
boxing	0	1
film	1	0
great	3	1
it	0	1
no	0	1
or	0	1
part	0	1
pathetic	0	1
plot	1	1
satire	1	0
scenes	1	2
the	0	2
twists	1	1
was	0	2
worst	0	1



Example of Binary NB

Four original documents:

- it was pathetic the worst part was the boxing scenes
- no plot twists or great scenes
- + and satire and great plot twists
- + great scenes great film

After per-document binarization:

- it was pathetic the worst part boxing scenes
- no plot twists or great scenes
- + and satire great plot twists
- + great scenes film

	NB Counts	
	+	–
and	2	0
boxing	0	1
film	1	0
great	3	1
it	0	1
no	0	1
or	0	1
part	0	1
pathetic	0	1
plot	1	1
satire	1	0
scenes	1	2
the	0	2
twists	1	1
was	0	2
worst	0	1

Example of Binary NB

Four original documents:

- it was pathetic the worst part ~~was the~~
boxing scenes
- no plot twists or great scenes
- + and satire ~~and~~ great plot twists
- + great scenes ~~great~~ film

After per-document binarization:

- it was pathetic the worst part boxing
scenes
- no plot twists or great scenes
- + and satire great plot twists
- + great scenes film

	NB Counts		Binary Counts	
	+	–	+	–
and	2	0	1	0
boxing	0	1	0	1
film	1	0	1	0
great	3	1	2	1
it	0	1	0	1
no	0	1	0	1
or	0	1	0	1
part	0	1	0	1
pathetic	0	1	0	1
plot	1	1	1	1
satire	1	0	1	0
scenes	1	2	1	2
the	0	2	0	1
twists	1	1	1	1
was	0	2	0	1
worst	0	1	0	1

Counts can still be 2! Binarization is within-doc!

Improvements - Dealing with Negation

- I really like this movie
 - I really **don't** like this movie
- Negation changes the meaning of "like" to negative.
- Negation can also change negative to positive-ish
 - **Don't** dismiss this film
 - **Doesn't** let us get bored



Dealing with Negation

- Simple baseline method:
Add NOT_ to every word between negation and following punctuation:

- didn't like this movie , but I



- didn't NOT_like NOT_this NOT_movie but I

Das, Sanjiv and Mike Chen. 2001. Yahoo! for Amazon: Extracting market sentiment from stock message boards. In Proceedings of the Asia Pacific Finance Association Annual Conference (APFA).

Bo Pang, Lillian Lee, and Shivakumar Vaithyanathan. 2002. Thumbs up? Sentiment Classification using Machine Learning Techniques. EMNLP 2002, 79-86. LEI/LECI



Other tasks: Spam Filtering

- SpamAssassin **Features:**
 - Mentions millions of (dollar) ((dollar) NN,NNN,NNN.NN)
 - From: starts with many numbers
 - Subject is all capitals
 - HTML has a low ratio of text to image area
 - "One hundred percent guaranteed"
 - Claims you can be removed from the list

Evaluating how well does our classifier work

- Let's address only binary classifiers:
 - Is this email spam?
 - spam (+) or not spam (-)
- We'll need to know
 - What did our classifier say about each email ?
 - What should our classifier have said,
 - i.e., the correct answer
 - usually as defined by humans ("gold label")

First step in evaluation: The confusion matrix

		<i>gold standard labels</i>	
		gold positive	gold negative
<i>system output labels</i>	system positive	true positive	false positive
	system negative	false negative	true negative

Accuracy

		<i>gold standard labels</i>	
		gold positive	gold negative
<i>system output labels</i>	system positive	true positive	false positive
	system negative	false negative	true negative

- $$\text{Accuracy} = \frac{tp+tn}{tp+fp+tn+fn}$$

Why don't we use accuracy?

- Suppose we look at 1,000,000 social media posts to find Pie-lovers (or haters)
 - 100 of them talk about our pie
 - 999,900 are posts about something unrelated
- Imagine the following simple classifier

Every post is "not about pie"

Why don't we use accuracy?

- In "nothing is pie" classifier we have:
 - 999,900 true negatives
 - 100 false negatives
 - 0 true positives
 - 0 true negatives
- Accuracy is $999,900/1,000,000 = 99.99\%$!
- But useless at finding pie-lovers (or haters)!!
 - Which was our goal!
- Accuracy doesn't work well for unbalanced classes
 - Most posts are not about pie!

Instead of accuracy we use Precision and Recall

		<i>gold standard labels</i>		
		gold positive	gold negative	
<i>system output labels</i>	system positive	true positive	false positive	precision = $\frac{tp}{tp+fp}$
	system negative	false negative	true negative	
		recall = $\frac{tp}{tp+fn}$		accuracy = $\frac{tp+tn}{tp+fp+tn+fn}$

- Precision: % of selected items that are correct
- Recall: % of correct items that are selected

Precision/Recall aren't fooled

- For our "nothing is pie" classifier we have:
- Precision = $0/1,000,000 = 0\%$
- Recall = $0/1,000,000 = 0\%$

$$\text{Precision} = \frac{\text{true positives}}{\text{true positives} + \text{false positives}}$$

$$\text{Recall} = \frac{\text{true positives}}{\text{true positives} + \text{false negatives}}$$

A combined measure: F1

- F1 is a combination of Precision and Recall.

$$F_1 = \frac{2PR}{P + R}$$



Summary: Naïve Bayes is Not So Naïve

- Very Fast
- Low storage requirements
- Work well with very small amounts of training data
- Robust to Irrelevant Features
 - Irrelevant Features cancel each other without affecting results
- Very good in domains with many equally important features
- A good dependable baseline for text classification

Credits

These slides are an adaptation of the slides made available by Dan Jurafsky and James M. Martin for the chapter [Naive Bayes, Text Classification, and Sentiment](#) of the book **Speech and Language Processing (3rd ed. draft)**

Available @ [Speech and Language Processing \(stanford.edu\)](#)