Naive Bayes
(Text classification)

Objectives

- Review NLP topics/vocabulary
- Review Bayes Rule
- Apply Naive Bayes to text classification
- Describe the Naive Bayes text algorithm (and be able to implement it in code).
    - Priors
    - Conditional probabilities
    - MLE
- Describe what Laplace smoothing is and why it’s used.

NLP review

- Describe
  - corpus
  - stop-words
  - stemming
  - lemmatization
  - tf matrix, tf-idf matrix
- What metrics can you use to compare to documents?

What is Bayes Rule?

What is Bayes Rule?

Bayes rule (theorem) describes the probability of an event, based on prior knowledge of conditions that might be related to the event.

$$P(A \mid B) = \frac{P(B \mid A)\, P(A)}{P(B)},$$

where A and B are events and $P(B) \neq 0$.

- $P(A \mid B)$ is a conditional probability: the likelihood of event A occurring given that B is true.
- $P(B \mid A)$ is also a conditional probability: the likelihood of event B occurring given that A is true.
- $P(A)$ and $P(B)$ are the probabilities of observing A and B independently of each other; this is known as the marginal probability.

Wikipedia

Bayes Rule for text classification

Text data happens to fit very nicely to the bullet points of Where/When to use Naive Bayes.

Due to bag-of-words featurization of text, where all features are assumed independent (naive), the input feature matrix is often very wide (~10,000 - 50,000 columns/dimensions) and can even be greater than the number of data samples ($n << p$, we have many more words than documents).

Naive Bayes is computationally efficient in this case, it’s just sums (as you’ll see).

Use cases: classifying emails (spam vs not), classifying news articles into genres, sentiment analysis (positive or negative review)

Bayes Rule for text classification

If A is spam or not, and B is email text, how would you state Bayes theorem?

If A is types of news articles (politics, sports, etc.), and B is the article text, how would you state Bayes theorem?

If A is sentiment, and B is review text, how would you state Bayes theorem?

$$P(A \mid B) = \frac{P(B \mid A)\, P(A)}{P(B)}$$

Naive Bayes classifier - overview

For every document, calculate the probability that the document belongs to each class and chose the class with the highest probability.

$$P(A \mid B) = \frac{P(B \mid A)\, P(A)}{P(B)} \Rightarrow P(y_c \mid w_1, \ldots, w_n) = \frac{P(y_c) \prod_{i=1}^{n} P(w_i \mid y_c)}{P(w_1, \ldots, w_n)}$$

To do this, calculate two things:

- **Priors**: The probability that a generic document belongs to each class: $P(y_c)$
- **Conditional Probabilities (Likelihoods)**: The probability that each word appears in each class: $P(w_i \mid y_c)$

So how do we actually get all those probabilities? Counting! Count occurrences in our training set to get approximations of the probabilities.

Naive Bayes Classifier algorithm, the details

$$P(y_c \mid w_1, \ldots, w_n) = P(y_c) \prod_{i=1}^{n} P(w_i \mid y_c)^{*}$$

Priors

$$P(y_c) = \frac{\sum y \equiv y_c}{|D|}$$

# of documents

Conditional Probabilities

count of token i across D, given y_c

$$P(w_i \mid y_c) = \frac{count(w_{D,i} \mid y_c) + 1}{\sum_{w \in V} [count(w \mid y_c) + 1]}$$

count of tokens across D, given y_c

We'll go through these equations in a detailed example.

* Note marginal probability denominator is not required to pick which class (same for all classes).

Priors

The priors are the likelihood of each class. Based on the distribution of classes in our training set, we can assign a probability to each class:

$$P\left(y_c\right) = \frac{\text{\# of articles in class c}}{\text{total \# of articles}}$$

Take a very simple example where 3 classes: *sports*, *politics* and *arts*. There are 3 sports articles, 4 politics articles and 1 arts article. There are 8 articles total. Here are our priors:

$$P\left(sports\right) = \tfrac{3}{8} = 0.375$$

$$P\left(politics\right) = \tfrac{4}{8} = 0.5$$

$$P\left(arts\right) = \tfrac{1}{8} = 0.125$$

Conditional probabilities (likelihoods)

We would like to get, for every word, the count of the number of times it appears in each class. We are calculating the probability that a random word chosen from an article of class *c* is word *w*.

$$P(w_i \mid y_c) = \frac{\#\text{ of times } w_i \text{ appears in articles of class } y_c}{\text{total }\#\text{ of words in articles of class } y_c}$$ *

Let’s continue our simple example and look at the word “ball”.

* Compare to equation a few slides previous. Not including Laplace smoothing for now...

Assume this is our count of “ball” in our corpus

Article type	Count of “ball”	Total # of words in article
Sports 1	5	101
Sports 2	7	93
Sports 3	0	122
Politics 1	0	39
Politics 2	0	81
Politics 3	0	142
Politics 4	0	77
Art 1	2	198

Calculate the following likelihoods:

Article type	Count of “ball”	Total # of words in article
Sports 1	5	101
Sports 2	7	93
Sports 3	0	122
Politics 1	0	39
Politics 2	0	81
Politics 3	0	142
Politics 4	0	77
Art 1	2	198

$$P(\text{"}ball\text{"} \mid sports) =$$

$$P(\text{"}ball\text{"} \mid politics) =$$

$$P(\text{"}ball\text{"} \mid arts) =$$

Calculate the following likelihoods:

Article type	Count of “ball”	Total # of words in article
Sports 1	5	101
Sports 2	7	93
Sports 3	0	122
Politics 1	0	39
Politics 2	0	81
Politics 3	0	142
Politics 4	0	77
Art 1	2	198

$$P(\text{"}ball\text{"} \mid sports) = \frac{5+7+0}{101+93+122} = \frac{12}{316} = 0.038$$

$$P(\text{"}ball\text{"} \mid politics) = \frac{0+0+0+0}{39+81+142+77} = \frac{0}{339} = 0$$

$$P(\text{"}ball\text{"} \mid arts) = \frac{2}{198} = 0.010$$

Putting it all together

Maximum Likelihood Estimation:
We need to pull this all together to use these calculations to make a prediction. Here, W is the content of an article and w1, w2, … , w_n are the words that make up the article.

$$P(y_c|W)=P(y_c)\times P(w_1|y_c)\times P(w_2|y_c)\times\ldots\times P(w_n|y_c)$$

We assign the article that has the largest probability. Note that since we made our incredibly naive assumption, these "probabilities" will not add to 1.

What topic does the document “**The Giants beat the Nationals**” belong to?

Putting it all together

$$
\begin{aligned}
P(sports \mid W) = P(sports) & \\
&\times P(\text{"}the\text{"} \mid sports) \\
&\times P(\text{"}giants\text{"} \mid sports) \\
&\times P(\text{"}beat\text{"} \mid sports) \\
&\times P(\text{"}the\text{"} \mid sports) \\
&\times P(\text{"}nationals\text{"} \mid sports)
\end{aligned}
$$

The first probability is the prior, and the remaining come from the Conditional Probabilities (likelihoods). We make the same calculation for each of the 3 classes and choose the class with the biggest probability.

Laplace Smoothing

What if a word has never appeared before in a document of a certain class?

The probability will be 0. Since we are multiplying the probabilities, the whole probability becomes 0! We basically lose all information.

To mitigate this effect, add 1 to the numerator and the number of words in the vocabulary to the denominator.

This is called Laplace Smoothing and serves to remove the possibility of having a 0 in the denominator or the numerator, both of which would break our calculation.

Preventing Numerical Underflow

Take the log of both sides. (this helps prevent numerical underflow in our calculations).

$$P\left(y_c \mid w_{d,1},\ldots,w_{d,n}\right) = P\left(y_c\right)\prod_{i=1}^{n} P\left(w_{d,i} \mid y_c\right)$$

$$\log\left(P\left(y_c \mid w_{d,1},\ldots,w_{d,n}\right)\right) = \log\left(P\left(y_c\right)\right) + \sum_{i=1}^{n} \log\left(P\left(w_{d,i} \mid y_c\right)\right)$$

Reference

sklearn: Naive Bayes

Wood Classifier Capstone

Naive Bayes demo

`naive_bayes_sklearn.ipynb`

Naive bayes summary

Pros:

- Good with wide data (p >> n)
- Good if n is small or n is quite big
- Fast to train - it's just counting!
- Good at online learning, streaming data, learns by processing one data point at a time
- Simple to implement, not necessarily memory-bound (DB implementations)
- Multi-class classification

Cons:

- Naive assumption means correlated features are not actually treated right
- Sometimes outperformed by other models

Objectives

- Review NLP topics/vocabulary
- Review Bayes Rule
- Apply Naive Bayes to text classification
- Describe the Naive Bayes text algorithm (and be able to implement it in code).
    - Priors
    - Conditional probabilities
    - MLE
- Describe what Laplace smoothing is and why it's used.