**Machine Learning Using Naïve Bayes Classification**

**Abstract:**

It is very needful problem in current scenario to predict the text category of a new data from existing categories data. In this page we have tried to explore Bayesian probability approach to predict the class of new text.

It uses the probability that new text belongs to which category with a numeric score. The maximum score will tell which category the new query text belongs to. This process is easy and works to good accuracy.

**Introduction:**

**Naive Bayes** are also known as **Naive Bayes Classifiers.** It works with the assumption that features are [statistically independent](https://brilliant.org/wiki/statistical-independence/?wiki_title=statistically%20independent) of one another. Unlike many other classifiers which assume that, for a given class, there will be some [correlation](https://brilliant.org/wiki/correlation/) between features, naive Bayes explicitly assumes the features as [conditionally independent](https://brilliant.org/wiki/conditional-independence/?wiki_title=conditionally%20independent) given the class. While this seems an overly simplistic (naive) restriction on the data, in practice naive Bayes is competitive with more sophisticated techniques and enjoys some theoretical support for its presumption.

Because of the independence assumption, naive Bayes classifiers are highly scalable and can quickly learn to use high dimensional features with limited training data. This is useful for many real world datasets where the amount of data may be small in comparison with the number of features for each individual piece of data, such as speech, text, and image data. Examples of modern applications include recommendation system, spam filtering, automatic medical diagnoses, medical image processing, and vocal emotion recognition.

**Example with Naïve Bayes**

Suppose we are building a classifier that checks from symptoms text whether disease is cough or conjunctivitis. Our training set has 255 sentences for cough and 255 for conjunctivitis, some examples are listed below:

|  |  |
| --- | --- |
| **Text** | **Disease** |
| Watery eyes and itchy nose | Conjunctivitis |
| and the sneezing and coughing begin | Cough |
| my eyes are like forever itchy what should i do | Conjunctivitis |
| When you start coughing and then you | Cough |
| my eyes so watery | Conjunctivitis |

Now, which category does the symptom “*I have red eyes and nose”* belong to?

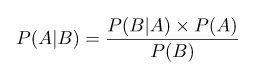
Since Naive Bayes is a probabilistic classifier, we want to calculate the probability that the symptom “*I have red eyes and nose”* is Cough, and the probability that it’s Conjunctivitis. Then, we take the largest one. Written mathematically, what we want is P(Cough | I have red eyes and nose ) — the probability that the category of a disease symptom is Cough given that the sentence is “I have red eyes and nose”.

To get the probability we need to convert text to numeric values. For this we can use word frequencies and count of distinct words and total words and use them to compute probabilities.

**Bayes’ Theorem**

Now we need to transform the probability we want to calculate into something that can be calculated using word frequencies.

Bayes’ Theorem is useful when working with conditional probabilities (like we are doing here), because it provides us with a way to reverse them:



In our case, we have P(Cough | I have red eyes and nose ), so using this theorem we can reverse the conditional probability:

  P(Cough | I have red eyes and nose ) = P(I have red eyes and nose | Cough ) \* P(Cough)/ P(I have red eyes and nose )

Since for our classifier we’re just trying to find out which category has a bigger probability, we can discard the divisor —which is the same for both diseases - and just compare

P(I have red eyes and nose | Cough ) \* P(Cough)

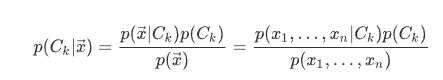
with

P(I have red eyes and nose | Conjunctivitis ) \* P(Conjunctivitis)

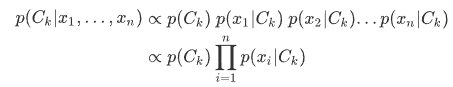
This is better, since we could actually calculate these probabilities.

There’s a problem though: “I have red eyes and nose” doesn’t appear in our training set, so this probability is zero. Unless every sentence that we want to classify appears in our training set, the model won’t be very useful.

**Naïve Bayes Theorem**

This can be seen from the equation of form:

Or



So here comes the Naive part: we assume that every word in a sentence is **independent** of the other ones. This means that we’re no longer looking at entire sentences, but rather at individual words. We write this as:

P(I have red eyes and nose ) = P(I) \* P(have) \* P(red) \* P(eyes) \* P(and) \* P(nose)

This assumption is very strong but very useful. It makes this model work well with little data. The next step is just applying this to what we had before:

P(I have red eyes and nose | Cough ) = P(I | Cough) \* P(have | Cough) \* P(red | Cough) \* P(eyes | Cough) \* P(and | Cough) \* P(nose | Cough)

And now, all of these individual words actually show up several times in our training set, and we can calculate them.

### Calculating probabilities

The final step is just to calculate every probability and see which one turns out to be larger.

Calculating a probability is just counting in our training set.

First, we calculate the a priori probability of each category: for a given sentence in our training set, the probability that it is Cough P(Sports) is 255/510. Then, P(Conjunctivitis) is 255/510. That’s easy to compute.

Then, calculating P(nose | Cough) means counting how many times the word “nose” appears in Cough samples (255) divided by the total number of words in Cough.

However, we run into a problem here: “red” doesn’t appear in any *Cough* sample! That means that P(red | Cough) = 0.  This is rather not helping since we are going to be multiplying it with the other probabilities, so we’ll end up P(I have red eyes and nose | Cough ) = 0. Doing things this way will not give information at all.

We solve it using [Laplace smoothing](https://en.wikipedia.org/wiki/Laplace_smoothing): we add 1 to every count so it’s never zero. To balance this, we add the number of possible words (total distinct words in both samples) to the divisor, so the division will never be greater than 1.

Thus P(eyes | Cough ) = (frequency of eyes in Cough samples) + 1)/(total words in Cough samples + total distinct words in both sample)

= (91 +1)/(3168 + 1827) = 0.0184

Doing in this we get:

P(I have red eyes and nose | Cough ) = **7.77280828952263E-17**

P(I have red eyes and nose | Conjunctivitis ) = **4.57427484108468E-14**

Hence our classifier gives “I have red eyes and nose” the **Conjunctivitis** category.

**References**

[1] A Bayesian Classification Approach Using Class-Specific Features for Text Categorization, Bo Tang, Student Member, IEEE, Haibo He, Senior Member, IEEE, Paul M. Baggenstoss, Senior Member, IEEE, and Steven Kay, Fellow, IEEE, IEEE TRANSACTIONS ON KNOWLEDGE AND DATA ENGINEERING, VOL. 28, NO. 6, JUNE 2016.

[2] Halftone Image Classification Using LMS Algorithm and Naive Bayes, Yun-Fu Liu, Student Member, IEEE, Jing-Ming Guo, Senior Member, IEEE, and Jiann-Der Lee, Member, IEEE, IEEE TRANSACTIONS ON IMAGE PROCESSING, VOL. 20, NO. 10, OCTOBER 2011.

[3] Internet Traffic Classification by Aggregating Correlated Naive Bayes Predictions, Jun Zhang, Member, IEEE, Chao Chen, Yang Xiang, Senior Member, IEEE, Wanlei Zhou, Senior Member, IEEE, and Yong Xiang, Senior Member, IEEE, IEEE TRANSACTIONS ON INFORMATION FORENSICS AND SECURITY, VOL. 8, NO. 1, JANUARY 2013.

[4] Some Effective Techniques for Naive Bayes Text Classification, Sang-Bum Kim, Kyoung-Soo Han, Hae-Chang Rim, and Sung Hyon Myaeng, IEEE TRANSACTIONS ON KNOWLEDGE AND DATA ENGINEERING, VOL. 18, NO. 11, NOVEMBER 2006

[5] Toward Optimal Feature Selection in Naive Bayes for Text Categorization, Bo Tang, Student Member, IEEE, Steven Kay, Fellow, IEEE, and Haibo He, IEEE TRANSACTIONS ON KNOWLEDGE AND DATA ENGINEERING, VOL. 28, NO. 9, SEPTEMBER 2016