1 The Naive Bayes classification method

1.1 Introduction

In our discussion of Bayes Theorem, we looked at a situation in which we had a population consisting of people infected with COVID-19 and people not infected, and we had a test that we could apply to determine which class an individual belonged to. Because our test was not 100 percent reliable, a positive test result didn't guarantee that a person was infected, and we used Bayes Theorem to evaluate how to interpret the positive test result. More specifically, our information about the test performance gave us the the conditional probabilities of positive and negative test results given infection status – so for example we were given $P(+|\text{infected})$, the chance of getting a positive test assuming the person is infected – and we used Bayes Theorem to determine $P(\text{infected}|+)$, the chance that a person was infected given a positive test result.

The Naive Bayes classification method is a generalization of this idea. We have data that belongs to one of two classes, and based on the results of a series of tests, we wish to decide which class a particular data point belongs to. For one example, we are given a collection of product reviews from a website and we wish to classify those reviews as either "positive" or "negative." This type of problem is called "sentiment analysis." For another, related example, we have a collection of emails or text messages and we wish to label those that are likely "spam" emails. In both of these examples, the "test" that we will apply is to look for the appearance or absence of certain key words that make the text more or less likely to belong to a certain class. For example, we might find that a movie review that contains the word "great" is more likely to be positive than negative, while a review that contains the word "boring" is more likely to be negative.

The reason for the word "naive" in the name of this method is that we will derive our probabilities from empirical data, rather than from any deeper theory. For example, to find the probability that a negative movie review contains the word "boring," we will look at a bunch of reviews that our experts have said are negative, and compute the proportion of those that contain the word boring. Indeed, to develop our family of tests, we will rely on a training set of already classified data from which we can determine estimates of probabilities that we need.

1.2 An example dataset

To illustrate the Naive Bayes algorithm, we will work with the "Sentiment Labelled Sentences Data Set" ([1]). This dataset contains 3 files, each containing 1000 documents labelled 0 or 1 for "negative" or "positive" sentiment. There are 500 of each type of document in each file. One file contains reviews of products from amazon.com; one contains yelp restaurant reviews, and one contains movie reviews from imdb.com.

Let's focus on the amazon reviews data. Here are some samples:

```
So there is no way for me to plug it in here in the US unless I go by a
    converter. 0
Good case, Excellent value. 1
Great for the jawbone. 1
Tied to charger for conversations lasting more than 45 minutes.MAJOR
    PROBLEMS!! 0
The mic is great. 1
I have to jiggle the plug to get it to line up right to get decent volume
    . 0
If you have several dozen or several hundred contacts, then imagine the
    fun of sending each of them one by one. 0
If you are Razr owner...you must have this! 1
Needless to say, I wasted my money. 0
What a waste of money and time!. 0
```

As you can see, each line consists of a product review followed by a 0 or 1; in this file the review is separated from the text by a tab character.

1.3   Bernoulli tests

We will describe the "Bernoulli" version of a Naive Bayes classifier for this data. The building block of this method is a test based on a single word. For example, let's consider the word **great** among all of our amazon reviews. It turns out that **great** occurs 5 times in negative reviews and 92 times in positive reviews among our 1000 examples. So it seems that seeing the word **great** in a review makes it more likely to be positive. The appearances of great are summarized in table 1 . We write ~**great** for the case where **great** does *not* appear.

Table 1: Ocurrences of **great** by type of review .

	+	-	total
great	92	5	97
~**great**	408	495	903
total	500	500	1000

In this data, positive and negative reviews are equally likely so $P(+) = P(-) = .5$ From this table, and Bayes Theorem, we obtain the empirical probabilities (or "naive" probabilities).

$$P(\textbf{great}|+) = \frac{92}{500} = .184$$

and

$$P(\textbf{great}) = \frac{97}{1000} = .097$$

Therefore

$$P(+|\textbf{great}) = \frac{.184}{.097}(.5) = 0.948.$$

In other words, *if* you see the word **great** in a review, there's a 95% chance that the review is positive.

What if you *do not* see the word **great**? A similar calculation from the table yields

$$P(+|\ \textbf{great}) = \frac{408}{903} = .452$$

In other words, *not* seeing the word great gives a little evidence that the review is negative (there's a 55% chance it's negative) but it's not that conclusive.

The word **waste** is associated with negative reviews. It's statistics are summarized in table 2.

Table 2: Ocurrences of **waste** by type of review .

	+	-	total
waste	0	14	14
~**waste**	500	486	986
total	500	500	1000

Based on this data, the "naive" probabilities we are interested in are:

$$P(+|\textbf{waste}) = 0$$
$$P(+|\ \textbf{waste}) = .51$$

In other words, if you see **waste** you definitely have a negative review, but if you don't, you're only slightly more likely to have a positive one.

What about combining these two tests? Or using even more words? We could analyze our data to count cases in which both **great** and **waste** occur, in which only one occurs, or in which neither occurs, within the two different categories of reviews, and then use those counts to estimate empirical probabilities of the joint events. But while this might be feasible with two words, if we want to use many words, the number of combinations quickly becomes huge. So instead, we make a basic, and probably false, assumption, but one that makes a simple analysis possible.

Assumption: We assume that the presence or absence of the words **great** and **waste** in a particular review (positive or negative) are independent events.

More generally, given a collection of words $w_1, \ldots, w_k$, we assume that their occurences in a given review are independent events.

Independence means that we have

$$P(\mathbf{great}, \mathbf{waste}|\pm) = P(\mathbf{great}|\pm)P(\mathbf{waste}|\pm)$$
$$P(\mathbf{great}, \ \mathbf{waste}|\pm) = P(\mathbf{great}|\pm)P(\ \mathbf{waste}|\pm)$$
$$\vdots$$

To generalize this, suppose that we have extracted keywords $w_1, \ldots, w_k$ from our data and we have computed the empirical values $P(w_i|+)$ and $P(w_i|-)$ by counting the fraction of positive and negative reviews that contain the word w_i:

$$P(w_i|\pm) = \frac{\text{number of } \pm \text{ reviews that mention } w_i}{\text{number of } \pm \text{ reviews total}}$$

Notice that we only count *reviews*, not *ocurrences*, so that if a word occurs multiple times in a review it only contributes 1 to the count. That's why this is called the *Bernoulli* Naive Bayes – we are thinking of each keyword as yielding a yes/no test on each review.

Given a review, we look to see whether each of our k keywords appears or does not. We encode this information as a vector of length k containing 0's and 1's indicating the absence or presence of the kth keyword. Let's call this vector the *feature vector* for the review.

For example, if our keywords are $w_1 = \mathbf{waste}$, $w_2 = \mathbf{great}$, and $w_3 = \mathbf{useless}$, and our review says

```
This phone is useless, useless, useless! What a waste!
```

then the associated feature vector is $f = (1, 0, 1)$.

If a review has an associated feature vector $f = (f_1, \ldots, f_k)$, then by independence the probability of that feature vector ocurring within one of the $\pm$ classes is

$$P(f|\pm) = \prod_{i:f_i=1} P(w_i|\pm) \prod_{i:f_i=0} (1 - P(w_i|\pm))$$

which we can also write

$$P(f|\pm) = \prod_{i=1}^{k} P(w_i|\pm)^{f_i} (1 - P(w_i|\pm))^{(1-f_i)}. \tag{1}$$

By Bayes Theorem, we can express the chance that our review with feature vector f is positive or negative by the formula:

$$P(\pm|f) = \frac{P(f|\pm)P(\pm)}{P(f)}$$

4

where
$$P(\pm) = \frac{\text{the number of } \pm \text{ reviews}}{\text{total number of reviews}}$$
and $P(f)$ is the fraction of reviews with the given feature vector.

A natural classification rule would be to say that a review is positive if $P(+|f) > P(-|f)$, and negative otherwise. In applying this, we can avoid computing $P(f)$ by just comparing the numerators $P(f|+)P(+)$ and $P(f|-)P(-)$ computed using eq. 1, and deciding that a review is positive if $P(f|+)P(+) > P(f|-)P(-)$ and negative otherwise.

References

[1]   U.C. IRVINE ML REPOSITORY. Sentiment Labelled Sentences Data Set.Available at https://archive.ics.uci.edu/ml/datasets/Sentiment+Labelled+Sentences.