**Naïve Bayes**

There are two Bayes Classification schemes out there. Sour…

**Bernoulli Naïve Bayes**

There’s a certain analogy with the Logistic Classification scheme. With logistic regression we have two labels (0,1) and some d-dimensional feature vector **x** = (x1, x2, …, xd). Further, the coordinates xj could take on either discrete or continuous values. Here we can have a bunch of mutually exclusive labels – aj­­ – and primitive events/features bk, which I guess we could group into a feature vector of sorts **b** = (b1, b2, …, bd). And likewise the b1 can also take on discrete or continuous values. For now we’ll restrict our consideration to the case where bk can be on or off (True or False). This is the domain of *Bernoulli Naïve Bayes*.

Diagram

Description automatically generated with low confidence

Anyway, we want to know, given some event E(**b**), like b1∩b2 or whatever, which of the aj is more likely. We can answer this question if we know P(aj), the a priori probabilities of each of the aj, and if we know P(bk|aj), the probabilities of the primitive events given the aj. Then given some event E(**b**) (could denote E(**b**) by just **b**, where vector **b** specifies the values of all the events bk=1…d), the probability of any given ai is, from the Event Composition file in Prob/Stat folder:



If we just trying to determine which of the ai is more likely, then since the denominators are all the same for a given E(**b**), we need only compute the numerator,



So from a geometrical perspective, we are just looking for which of the ai’s contains the most area of the event E(**b**). By the way, this is called Naïve Baye’s because we presume the events **b** are all independent.

**Example**

This process is often applied to spam filters. The aj = spam, not spam. And the primitive events bk = different words, like: Andrew, Hi, Penis, etc. Note k would just be the difference between the total area of our universal set and bk. Can draw a diagram,

Chart

Description automatically generated

The bk’s all overlap surely, but can’t crowd the diagram too much. Usually P(S) and P(NS) would be ascertained from some training data. This would just be P(S) = #S/(#S + #NS) and P(NS) = #NS/(#S + #NS) respectively. And from this training data, we could could also work out the probabilities P(Hi|S), P(Hi|NS), etc., by finding the number of times the word Hi shows up in a Spam mail and dividing by the total number of words.

And then in the testing data, we’d probably let E(**b**) = b1 ∩ b2 ∩ b3 … ∩ bn, i.e., the intersection of any subset of the words in our list, and calculate:



where in the last line we’re presuming the bk primitive events are all independent (this doesn’t mean they can’t happen at the same time). And we’d just compare to see whether a = S or a = NS gives the higher probability.

*Technical Note: one generally sets a floor on the probability of a primitive event P(bk|ai) to, in this context, 1/total number of words in ai = S or NS emails.*

**Example**

Let’s do a more concrete example. Let’s say we have the following table (LP, LS, LC),

|  |  |  |
| --- | --- | --- |
| **Loves Popcorn** | **Loves Soda** | **Loves Cool as Ice** |
| Yes | Yes | No |
| Yes | No | No |
| No | Yes | Yes |
| No | Yes | Yes |
| Yes | Yes | Yes |
| Yes | No | No |
| No | No | No |

The classification we’re interested in is the last column, a1,2 = {LC = Yes, LC = No}. Our events are: b1 = {LP = Yes, LP = No}, and b2 = {LS = Yes, LS = No}. And we’d like to calculate P(aj|E(**b**)). Where E(**b**) is some composite event having to do with the outcomes of b1 and b2. Well. So we need,



and,



Then we have of course,



Let’s try to classify someone as either Loving Cool as Ice (a1) or not Loving Cool as Ice (a2), given that Loves Popcorn = Y and Loves Soda = N, i.e., P(a1|E(b)) vs. P(a2|E(b)), where E(b) = (b1=y)∩(b2=n). So then,



whereas,



This is clearly larger, and so we’d say:



We assumed evens b1 and b2 are independent. Are they? Well,



and so,



On the other hand, P(b1b2) is:



Can see clearly that P(b1b2) ≠ P(b1)P(b2). But really, it’s not too far off. So the approximation is probably okay. And in a larger set of data, it would probably be a good assumption.

**Multinomial/Gaussian Naïve Bayes**

Say the events bk can take a range of values, for fixed k, bk = 0, 1, 2, 3, 4, etc., or whatever, just like the xk can in a feature vector. This is the domain of *Multinomial Naïve Bayes*. For instance we could look at the *number* of times Hi shows up in an email, instead of just whether or not it shows up in an email. I’m going to represent this by slicing the bk boxes into pieces. A different shade represents a different value. And I guess bk = 0 would be equivalent to k and represent all the space surrounding the bk box. The area allocated to bk = 0 is probably greatly exaggerated, but whatever. We could extend this concept to the case where the bk’s take a continuous range of values. This would be *Gaussian Naïve Bayes*. Then their number of slices would go to ∞, and their width would shrink to 0 (they’d become lines basically). And certainly the area associated with k would shrink to a line as well.

Chart, funnel chart

Description automatically generated

In the discrete bk case, we can still use our formula,



without any issues. E(**b**) would be something like (b1=3) ∩ (b2=4) ∩ (b3=1). However, in the case that bk becomes associated with a continuous probability distribution function, P(bk|aj) would be a probability density function, i.e., a likelihood, not a probability. And so likewise the P(E(**b**)|ai)’s would also not be probabilities, but rather likelihoods. Often these will take the form of Gaussian distributions. In order to compare the likelihood of two different a’s, we still just have to compare numerators though. So this won’t practically alter our methodology.

**Example**

In StatQuest video, he examines probability someone does or does not like Troll 2. In particular, we want a rubric/classification scheme we can use to predict whether some random person will like it. So we’ll call these events a1 = T and a2 = . And then we examine populations of people who fall into these categories. We examine their popcorn, soda, and candy consumption. Each of these three events: b1,2,3 = p, s, c are continuous variables. And so P(p|T), P(s|T), P(c|T) as well as P(p|), P(s|), P(c|) are continuous, presumably Gaussian, distributions. We find following distributions:

Chart, line chart

Description automatically generated

say. So to determine whether some random person with values p\*, s\*, c\*, does or doesn’t like Troll 2, we’d calculate and compare:



Obviously we implicitly presumed the events p, s, and c are independent.

**Example**

Now let’s take a look at this table. We’d like to rig a Gaussian Naïve Bayes classifier which accounts for the Age data.

|  |  |  |  |
| --- | --- | --- | --- |
| **Loves Popcorn** | **Loves Soda** | **Age** | **Loves Cool as Ice** |
| Yes | Yes | 7 | No |
| Yes | No | 12 | No |
| No | Yes | 18 | Yes |
| No | Yes | 35 | Yes |
| Yes | Yes | 38 | Yes |
| Yes | No | 50 | No |
| No | No | 83 | No |

So we have a1,2 → {LC = y, LC = n}, b1 = LP, b2 = LS, and b3 = Age. I guess we’d have to presume some kind of Gaussian distribution for the Age. To work this out, we could make a histogram of the ages and then try to find a best fit Gaussian. For simplicity, we can just calculate the average and std, and then approximate with a Gaussian with the same features. So a quick calculation shows μ = 35, and σ = 24. So,



Actually it’s more to the point to calculate P(age|a1) and P(age|a2). So we have to calculate separate means and std’s for age|a1 = {18, 35, 38} and age|a2 = {7, 12, 50, 83}. These are: μ1 = 30, σ1 = 9, and μ2 = 38, σ2 = 31 respectively. And also going to call age b3 to keep consistent with following notation. So we have:



and then we’ll recall from previous example that we found,



and



Now let’s try to classify someone as either Loving Cool as Ice (a1) or not Loving Cool as Ice (a2), given that Loves Popcorn = Y and Loves Soda = y, and Age = 44, i.e., P(a1|E(b)) vs. P(a2|E(b)), where E(b) = (b1=y)∩(b2=y)∩(b3=44). So first we examine,



vs.



Without actually calculating anything, it is evident that a2 is the more likely. So we’d say,



**Exploring the Model and Hyperparameters**

I don’t know of any hyperparameters. So just going to do the model on linearly classified data, pure and 10% outlier,

A diagram of a red and blue diagram

Description automatically generated A diagram of a graph

Description automatically generated with medium confidence

and here’s a quadratic surface,

A diagram of a graph

Description automatically generated A diagram of a graph

Description automatically generated

Here’s a triple class linear-ish boundary,

A diagram of a graph

Description automatically generated with medium confidence A diagram of a graph

Description automatically generated with medium confidence

and here’s the triple class circles,

A diagram of a cell

Description automatically generated with medium confidence A diagram of different colored circles

Description automatically generated

Doesn’t do too great here. But it did do super well on the Iris dataset. So have to look into this more.