

Pattern Recognition: Week 6

Naïve Bayes – accompaniment to class exercises

John Quinn

May 26, 2008

Bayes rule

- Well known formula to calculate posterior probabilities.

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)}$$

- A could be “has malaria”, and B could be “blood test positive for parasites”.
- $P(A)$ is the prior probability of having malaria.
- $P(A = 0|B) + P(A = 1|B)$ always equals one.

If we have several features, B_1, \dots, B_N , and we want to work out a class C , then Bayes rule tells us:

$$P(C|B_1, \dots, B_N) = \frac{P(B_1, \dots, B_N|C)P(C)}{P(B_1, \dots, B_N)} \quad (1)$$

To make things simpler, we can miss out the denominator and say the probability is proportional:

$$P(C|B_1, \dots, B_N) \propto P(B_1, \dots, B_N|C)P(C) \quad (2)$$

The conditional term on the right is tricky though! We can make an approximation which is easier to work out:

$$P(C|B_1, \dots, B_N) \propto P(B_1|C)P(B_2|C) \dots P(B_N|C)P(C) \quad (3)$$

This approximation is a *conditional independence assumption*.

- So, for each class, we just work out the conditional probability of each variable.
- We also calculate the prior of the class (what proportion of the rows are class 1, and what proportion are class 2).
- For a new sample that we want to classify, we multiply together all the conditional probabilities according to (3), and see which class is most likely.

Discrete example

V_1	V_2	Class
0	1	1
1	1	1
1	0	0
1	1	1
0	0	0
0	0	1
1	0	0
1	0	0

If you have new data, $V_1 = 0$ and $V_2=1$, what is the most likely class?

Continuous data

For continuous data, it is common to express $P(B|C)$ (probability of a feature B value given a particular class C) as a Gaussian distribution.

We can learn the mean μ_B and variance σ_B^2 of the features within each class.

For a new point x , the conditional probability is then

$$P(B = x|C) = \frac{1}{\sqrt{2\pi\sigma_B^2}} \exp -\frac{(x - \mu_B)^2}{2\sigma_B^2} \quad (4)$$

Which is easy to work out automatically. . .

How to do this in Python

All you need are three commands:

- `mu = mean(a)` - finds the mean of array `a`.
- `sigma = var(a)` - finds the variance of array `a`.
- `normpdf(x, mu, sigma)` - finds the Gaussian (normal) probability density for value `x` given the mean `mu` and variance `sigma`.

Just import the right libraries:

- `from numpy import *`
- `from pylab import *`

Evaluation of classification performance

Having produced a system to recognise patterns, you need to evaluate how well it has done. If you have a binary classifier, you can calculate the number of:

- True positives (TP)
- False positives (FP)
- True negatives (TN)
- False negatives (FN)

Having calculated these quantities you can work things out such as the accuracy:

$$accuracy = \frac{TP + TN}{TP + TN + FP + FN} \quad (5)$$

or (equivalently) the error rate:

$$error\ rate = \frac{FP + FN}{TP + TN + FP + FN} \quad (6)$$

In medical applications it is common to use sensitivity and specificity:

$$\textit{sensitivity} = \frac{TP}{TP + FN} \quad (7)$$

$$\textit{specificity} = \frac{TN}{TN + FP} \quad (8)$$

The closer each of these are to 1 the better. We want to be sensitive (pick up as many cases of a problem as possible), but also specific (not too many false alarms).