Statistics Fall 2005

Marcus Kracht

Assignments, Part 3.

Ex 3.1 Consider a French roulette table: it is a wheel of luck with 37 cells. There are various betting schemes. We consider some of them. You may bet on a particular number between 1 and 36. If that number appears, you get 36 times the amount. If the number does not appear, you lose. Let X_i be the associated dollar amount you receive for 1 dollar you place on the number i. Compute the expectation of X_i. Compute the variance of X_i. Do the same for Y_p: this represents your bet on even numbers. Interpret the results you get.

Solution If i shows up (probability 1/37) we win 35 (!). This is because although we do get 36 dollars, one of them was still ours. Otherwise (probability 36/37) we get -1 (we loose the money we put on the table). Thus $\mathsf{Ex}X_i = 35/37 - 36/37 = -1/37$. This is what the casino will cash on average from the customers. If the number is even (probability 18/37) you win 1. Otherwise (probability 19/37). Thus $\mathsf{Ex}Y_p = 18/37 - 19/37 = -1/37$. Thus, betting on even/odd or particular numbers is any better. Numbers:

$$\begin{aligned}\mathsf{E}X_i &= -0.0270273\\ \mathsf{V}X_i &= 34.08035\\ \mathsf{E}Y_p &= -0.0270273\\ \mathsf{V}Y_p &= 0.9992695\end{aligned}$$

This can be seen as follows: the betting scheme is fair up to the point that we lose our money when 0 shows up. This is 1/37 of the entire cases. And this is the same for all betting schemes of roulette. (So, the casion cashes around 2.7 percent on average ...) The variance is high when you bet on a number: the difference between winning and losing is big. That's for real gamblers. With Y_p the difference with the expected value is small: always around 1.

Ex 3.2 Write an R-program that inputs a vector and, assuming that it represents a random variable over a Laplace-space, computes its expectation and variance. Test it on the results you obtained in the previous exercise.

Solution The expectation is actually the mean. If you really want to define a function, do this:

(1)
```
> ex <- function (x) mean(x)
```

The variance is computed as follows:

(2)
```
> vr <- function (x) sum ((x - mean(x)) ** 2)/
    length(x) ** 2
```

(Alternatively, you can use the caret for exponentiation.) I then performed

(3)
```
> mean (rep(c(1,-1), c(1,36)))
[1] - 0.0270273
> vr (rep(c(1,-1), c(1,36)))
[1] 34.08035
```

Notice that R has an inbuilt function `var`, which computes the sample mean. This is slightly different (as we will see later).

Ex 3.3 Write an R-program that inputs the a priori probabilities for a Bernoulli experiment and a vector and computes the a posterior probabilities. *Hint.* The a priori probabilities take the form of just a single value; the vector you may take either as a vector of reals (either 0 or 1), or a Boolean vector.

Solution I gave the formula as $f_{\pi(k,n)}(\alpha)$, where n is the length of the vector, and k the number of successes. It is

(4) $$f_{\pi(k,n)}(\alpha) = \frac{\alpha}{\pi(k,n)(1-\alpha)+\alpha}$$

I assume that the vector is given in reals. Then k equals sum (x), while n is length (x).

(5)
```
post <- function (p, x) p/(1.25 ** (sum (x)) *
    (.5/.6) ** (length (x) - sum (x)) * (1 - p) + p)
```

Here is a sneaky input a logical vector and compute k:

(6) `lp <- function (x) y <- (rep(1, length (x))); sum (y[x])`

This takes a logical vector, creates a vector of equal length consisting just of 1s, and then cuts out all coordinates where x is false. Thus, we get a vector consisting of 1s whose length is k. (I used 'sum' but 'length' would have done just as well and would be a little bit faster.)