Computational Statistics & Probability

Problem Set 1
Bayesian Inference

Author: Neelesh Bhalla
Collaborators: Nils Marthiensen, Chia-Jung Chang

2022-11-16

1. COVID Home Test

A home COVID-19 antigen test developed by BinaxNOW is accepted in the US for travel. BinaxNOW reports that in a clinical trial their test correctly gave a positive result 84.6% of the time and correctly gave a negative result 98.5% of the time. We are assuming a PCR result is ground truth: if a PCR test is positive, then the patient has COVID-19; if a PCR test is negative, then the patient does not have COVID-19. Suppose that 10% of the people in your community are currently infected. (This is your base rate for exposure.) Now suppose you take a BimaxNOW COVID-19 antigent test.

a) Given that your BinaxNOW test is positive, what is the probability that you have COVID-19?

```
Pr_Positive_Rona <- 0.846
Pr_Negative_NoRona <- 0.985
Pr_Rona <- 0.1
Pr_Positive <- Pr_Positive_Rona * Pr_Rona + (1 - Pr_Negative_NoRona) * (1 - Pr_Rona)
(Pr_Rona_Positive <- Pr_Positive_Rona * Pr_Rona / Pr_Positive)
```

```
## [1] 0.8623853
```

b) Given that your BinaxNOW test is negative, what is the probability that you have COVID-19?

```
Pr_Positive_Rona <- 0.846
Pr_Negative_NoRona <- 0.985
Pr_Rona <- 0.1
Pr_Negative <- Pr_Negative_NoRona * (1 - Pr_Rona) + (1 - Pr_Positive_Rona) * Pr_Rona
(Pr_Rona_Negative <- (1 - Pr_Positive_Rona) * Pr_Rona / Pr_Negative)
```

```
## [1] 0.01707506
```

c) Suppose instead the base rate for infection is 30%. This is extreme: The peak daily number of cases in Germany (so far) was 250,000 in April 2022. People are considered infectious for 10 days. So, no more than 2.5M people were infected with COVID in Germany during this peak, which is 3% of the population. Do you think a negative test result of a BinaxNOW home test is sufficient to conclude that you do not have COVID-19? Why or why not?

Let's calculate the probability of not having COVID, given the test is negative with the new base rate of 30%.

```
Pr_Positive_Rona <- 0.846
Pr_Negative_NoRona <- 0.985
Pr_Rona <- 0.3
Pr_Negative <- Pr_Negative_NoRona * (1 - Pr_Rona) + (1 - Pr_Positive_Rona) * Pr_Rona
(Pr_NoRona_Negative <- Pr_Negative_NoRona * (1 - Pr_Rona) / Pr_Negative)
```

```
## [1] 0.9372027
```

Given a negative test, the probability of actually not having COVID is nearly 94% using a BinaxNOW home test. A 6% chance of having COVID even though the test is negative seems pretty high, so the test (at least if only performed once) is no longer a sufficient indicator.

2. Swing Voters

Imagine a country where there are only two political parties, Red and Blue, which divide the electorate equally. One difference between registered Blue voters and registered Red voters is their willingness to vote for the opposing party's candidate. Blue voters vote Red 20% of the time, otherwise they vote Blue. Red voters vote Blue 10% of the time, otherwise they vote Red. Voters who switch are called swing voters. Smith was a swing voter in the last election but you do not know whether he is Red or Blue.(Nobody changes parties.) What is the probability that Smith will be a swing voter in the next election?

```
red_swing <- 0.1
blue_swing <- 0.2
# Smith was a swing voter, but he could be red or blue swinger.
smith_is_red <- red_swing / (red_swing + blue_swing)
smith_is_blue <- 1 - smith_is_red
smith_swing_again <- smith_is_red * red_swing + smith_is_blue * blue_swing
(smith_swing_again)
```

```
## [1] 0.1666667
```

3. More Precision

Suppose you want a very precise estimate of the proportion of the Earth's surface that is covered in water. Specifically, suppose you would like the 99% percentile interval of the posterior distribution of p (the estimated proportion of water) to have a width of no greater than 0.05 – that is, the distance between the lower and upper bound on p should be no greater than 0.05. How many times must you toss the globe to achieve this precision? An exact count is unnecessary. I am primarily interested in your approach.

```
library(rethinking)
```

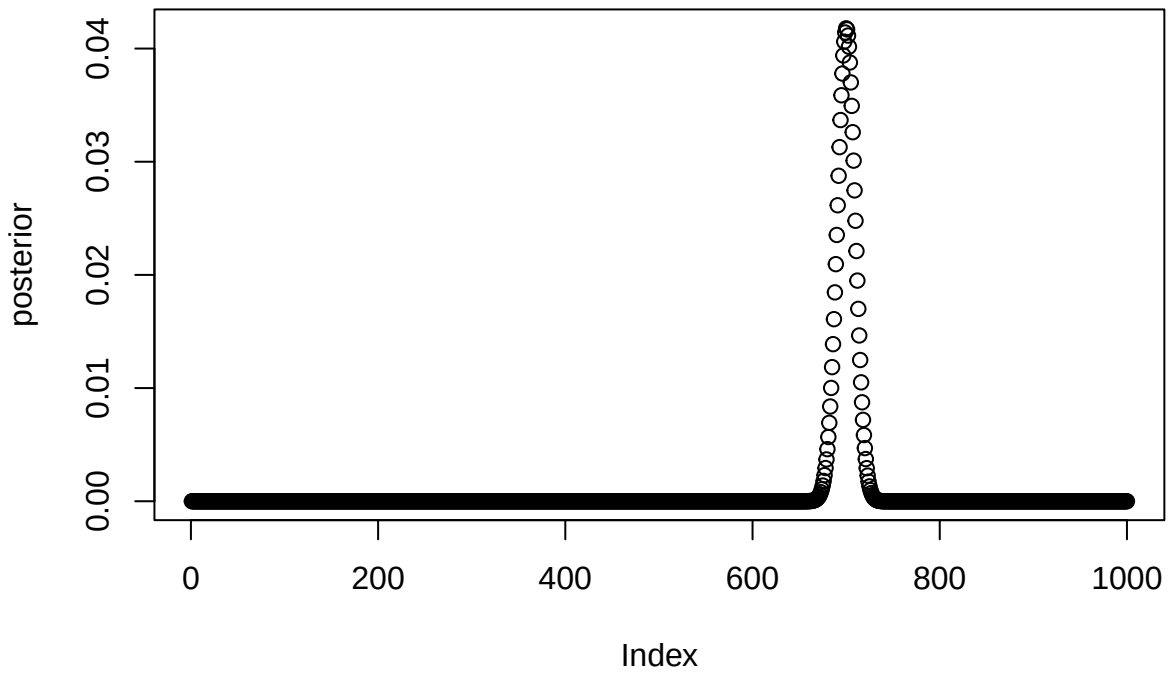
```
## Loading required package: rstan
```

```
## Loading required package: StanHeaders
```

```
##
## rstan version 2.26.13 (Stan version 2.26.1)
```

```
## For execution on a local, multicore CPU with excess RAM we recommend calling
## options(mc.cores = parallel::detectCores()).
## To avoid recompilation of unchanged Stan programs, we recommend calling
## rstan_options(auto_write = TRUE)
## For within-chain threading using ‘reduce_sum()‘ or ‘map_rect()‘ Stan functions,
## change ‘threads_per_chain‘ option:
## rstan_options(threads_per_chain = 1)

## Do not specify '-march=native' in 'LOCAL_CPPFLAGS' or a Makevars file

## Loading required package: cmdstanr

## This is cmdstanr version 0.5.3

## - CmdStanR documentation and vignettes: mc-stan.org/cmdstanr

## - CmdStan path: C:/Users/nmart/OneDrive - fs-students.de/Dokumente/.cmdstan/cmdstan-2.30.1

## - CmdStan version: 2.30.1

## Loading required package: parallel

## rethinking (Version 2.21)

##
## Attaching package: 'rethinking'

## The following object is masked from 'package:rstan':
##
##     stan

## The following object is masked from 'package:stats':
##
##     rstudent
```

```r
p_grid <- seq( from=0 , to=1 , length.out=1000 )
prior <- rep( 1 , 1000 )
likelihood <- dbinom( 1610 , size=2300 , prob=p_grid )
posterior <- likelihood * prior
posterior <- posterior / sum(posterior)
plot(posterior)
```




```
(interval <- PI(posterior, prob = 0.99))
```

```
##          1%       100%
## 0.00000000 0.04016246
```

With 2300 tosses the distance between the lower and upper bound comes down to 0.04. This is therefore sufficient to achieve the desired accuracy.