Lab 2 – Beta-Binomial Distribution

Rebecca C. Steorts

January 2018

```
library(tidyverse)
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.1 --

## v ggplot2 3.3.5     v purrr   0.3.4
## v tibble  3.1.3     v dplyr   1.0.7
## v tidyr   1.1.3     v stringr 1.4.0
## v readr   2.0.1     v forcats 0.5.1

## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

In class, you saw the Binomial-Beta model. We will now use this to solve a very real problem! Suppose I wish to determine whether the probability that a worker will fake an illness is truly 1%. Your task is to assist me! Tasks 1–3 will be completed in lab and tasks 3–5 should be completed in your weekly homework assignment. You should still upload task 3 even though this will be worked through in lab!

Task 1

Let's start by quickly deriving the Beta-Binomial distribution.

We assume that

$$X \mid \theta \sim \text{Binomial}(\theta)$$

,

$$\theta \sim \text{Beta}(a, b),$$

where a, b are assumed to be known parameters. What is the posterior distribution of $\theta \mid X$?

$$\begin{align}
p(\theta \mid X) &\propto p(X \mid \theta)p(\theta) \tag{1}\\
&\propto \theta^{x}(1-\theta)^{(n-x)} \times \theta^{(a-1)}(1-\theta)^{(b-1)} \tag{2}\\
&\propto \theta^{x+a-1}(1-\theta)^{(n-x+b-1)}. \tag{3}
\end{align}$$

This implies that

$$\theta \mid X \sim \text{Beta}(x+a, n-x+b).$$

Task 2

Simulate some data using the `rbinom` function of size $n = 100$ and probability equal to 1%. Remember to `set.seed(123)` so that you can replicate your results.

The data can be simulated as follows:

```
# set a seed
set.seed(123)
# create the observed data
obs.data <- rbinom(n = 100, size = 1, prob = 0.01)
# inspect the observed data
head(obs.data)
```

```
## [1] 0 0 0 0 0 0
```

```
tail(obs.data)
```

```
## [1] 0 0 0 0 0 0
```

```
length(obs.data)
```

```
## [1] 100
```

Task 3

Write a function that takes as its inputs that data you simulated (or any data of the same type) and a sequence of θ values of length 1000 and produces Likelihood values based on the Binomial Likelihood. Plot your sequence and its corresponding Likelihood function.

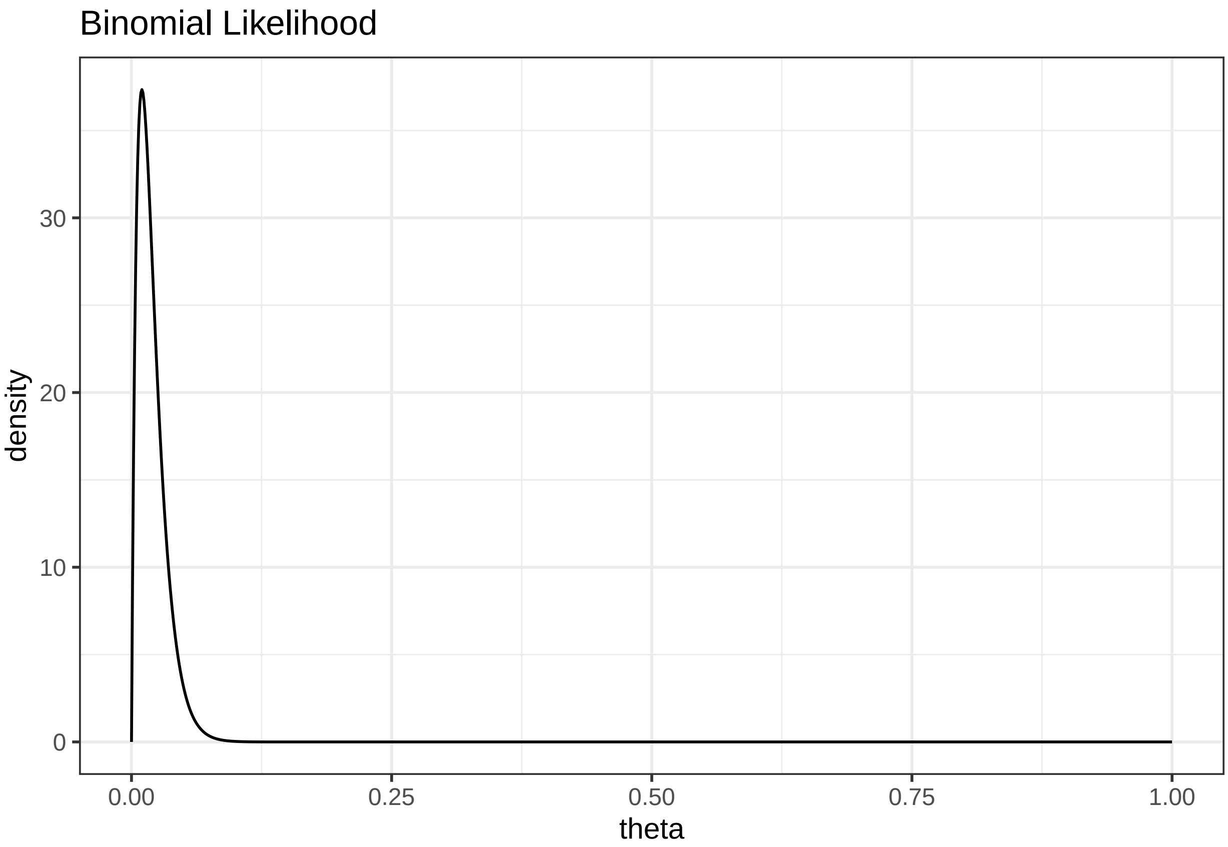
The likelihood function is given below. Since this is a probability and is only valid over the interval from $[0, 1]$ we generate a sequence over that interval of length 1000.

You have a rough sketch of what you should do for this part of the assignment. Try this out in lab on your own.

```
### Bernoulli LH Function ###
# Input: obs.data, theta
# Output: bernoulli likelihood

BernoulliLhFunction <- function(obs.data, theta) {
  n <-length((obs.data))
  x <- sum(obs.data)
  like <- dbeta(theta, x + 1, n -x +1)
  return((like))
  }
### Plot LH for a grid of theta values ###
# Create the grid #
# Store the LH values
# Create the Plot
theta <- seq(0,1,length = 1000)
like <- BernoulliLhFunction(obs.data, theta)
df <- data.frame(theta,like)

df %>%
  ggplot(aes(x = theta, y = like)) +
    geom_line() +
    labs(
      title = "Binomial Likelihood",
      x = "theta",
      y = "density") +
    theme_bw()
```



Task 4 (To be completed for homework)

Write a function that takes as its inputs prior parameters a and b for the Beta-Bernoulli model and the observed data, and produces the posterior parameters you need for the model. **Generate and print** the posterior parameters for a non-informative prior i.e. (a,b) = (1,1) and for an informative case (a,b) = (3,1)}.

```
posteriorGenerate <- function(a,b, obs.data) {
  n = length(obs.data)
  x = sum(obs.data)
  a.post = x + a
  b.post = n - x + b
  return(c(a.post,b.post))
}

noninform_param <- posteriorGenerate(1,1,obs.data)
inform_param <- posteriorGenerate(3,1,obs.data)
```

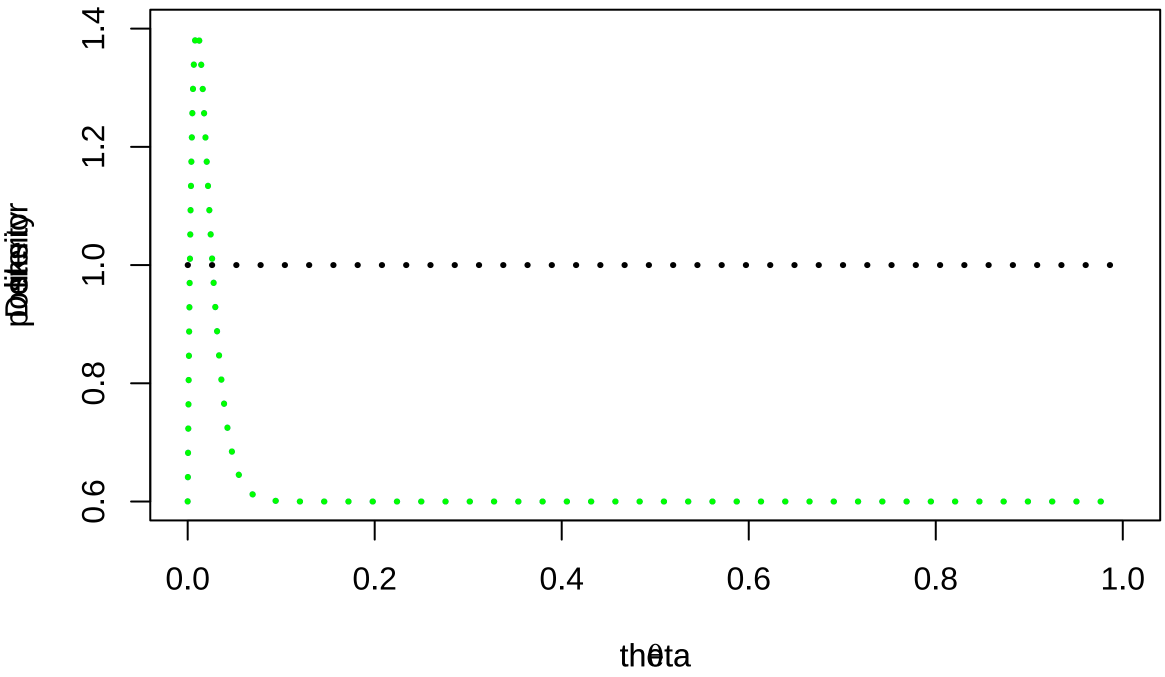
Task 5 (To be completed for homework)

Create two plots, one for the informative and one for the non-informative case to show the posterior distribution and superimpose the prior distributions on each along with the likelihood. What do you see? Remember to turn the y-axis ticks off since superimposing may make the scale non-sense.

```
prior = dbeta(theta, 1, 1)
like = BernoulliLhFunction(obs.data, theta)
posterior = dbeta(theta, noninform_param[1],noninform_param[2])
```

```
non_informdf = data.frame(theta,prior,like,posterior)

plot(theta, prior,type = "l", ylab = "Density",
     lty = 3, lwd = 3, xlab = expression(theta))
par(new = TRUE)
plot(theta, like, type = "l",
     lty = 3, lwd = 3, axes = FALSE, col = "blue")
par(new = TRUE)
plot(theta, posterior, type = "l",
     lty = 3, lwd = 3, axes = FALSE, col = "green")
```


Density
like
posterior
theta
θ