Homework 2

Stephen R. Proulx

12/27/2020

Bayesian Statistical Modeling Winter 2021

Homework, Week 2

When is homework due? Homework is due on Mondays (Tuesday if Monday is a holiday). This assignment is due on Monday, 1/14/2021. Submit homework to gradescope: *Link here*

Problems from the end of chapter 2.

2E1

These expressions say
(1) the total probability of rain on any given day
(2) the probability of rain given it is monday. So in english, this would be the "probability of rain on monday"
(3) This is hte probablity it is Monday given that it is raining. Note that if rain does not depend on the day of the week then this is really just the probability it is a Monday.
(4) This is the joint probability of "rain" and "monday" divided by the probability it is monday. Try drawing the Venn diagram to help visualize this.

2E2

Pr(Monday|rain) means the probability it is Monday given that it is raining, so the answer is (3). Note that it is not the joint probability that it is raining and monday, which would be Pr(Monday, rain).

2M1

Recall that to compute the posterior distribution we do:
1. Copmute the likelihood of the data given the parameters
2. Compute the prior of the parameters
3. Divide by the normalizing factor In this example, the likelihood will always be the binomial probability of observing the data given the parameters.

(1) data is {W,W,W}, so `data = c(1,1,1)` where we code trials that show water as 1 and land as 0.

```
data = c(1,1,1,0,0,0,0,0)
```

First we start by making our grid of parameters. Here we have one parameter, p, which is the probability of water. We'll make a grid that is spaced out in 0.05 intervals. There are many ways to use the seq function to do this, I prefer using `seq` where we specify `by`. A convenient way to do this is to set up our dataframe right from the start with the grid of parameters and later use mutate to add columns for the rest of the computations we wish to perform.

```
post_approx <- tibble(p = seq( from=0 , to=1 , by=0.05 ) )
```

We will use the mutate function to do the calculations necessary to fill in each of these columns. You can write one mutate function to do it all in one line, but here we will spread it out to make it more clear.

First we calculate the likelihood of the data given the parameter, for each value of p in the dataframe. The data are summarised by the number of 1's in the dataset and the length of the dataset.

```
post_approx <- mutate(post_approx , likelihood = dbinom( sum(data) , size=length(data) , prob=p  )  )
```

Next we calculate the prior for each value of p. Here we are assuming that the prior is uniform, so we just put 1 in for each value.

```
post_approx <- mutate(post_approx, prior= 1)
```

Now we combine the likelihood and prior by multiplying them together and call this the `raw.posterior`.

```
post_approx <- mutate(post_approx, raw.posterior=likelihood*prior)
```

To caclulate the normalizing constant we just add up each of the `raw.posterior` values.

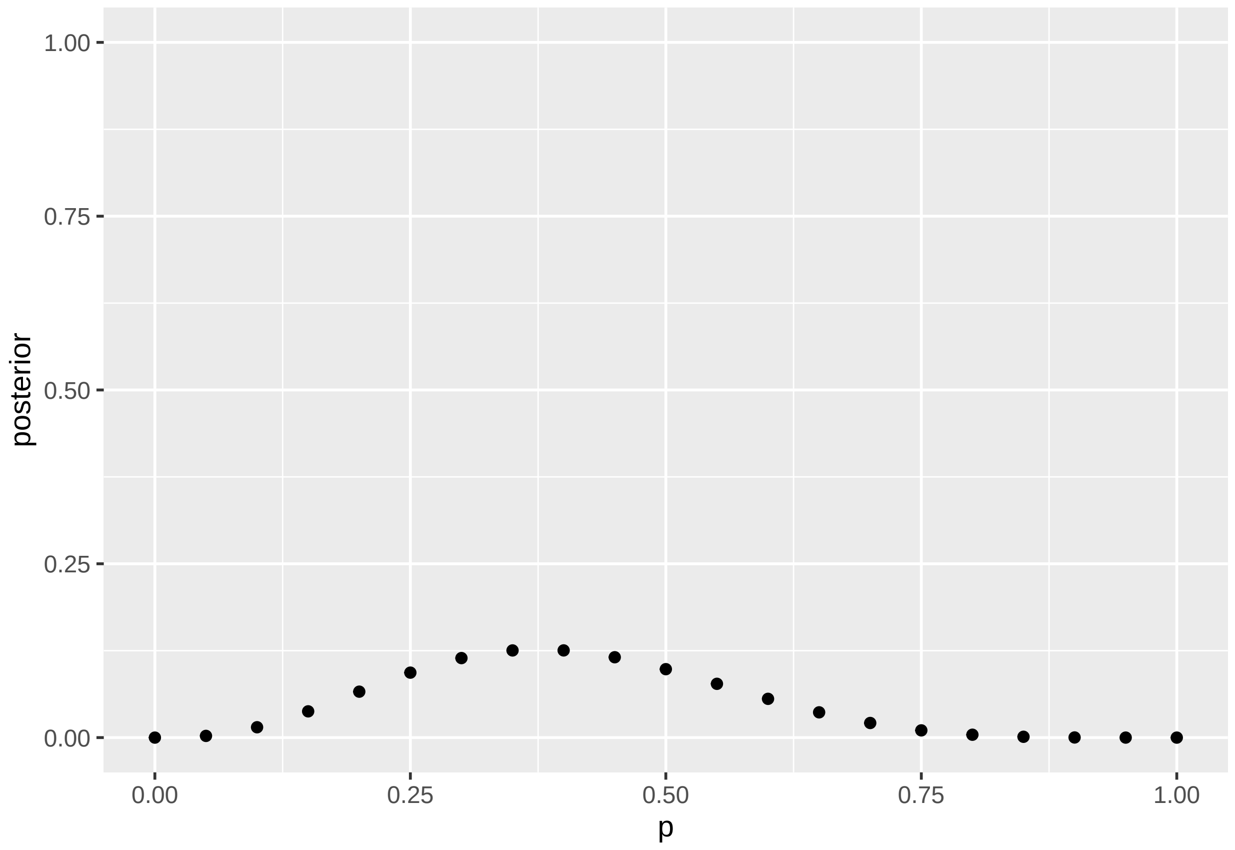
```
norm.constant <- sum(post_approx$raw.posterior)
```

And now apply this normalizing constant by dividing the `raw.posterior` values by it.

```
post_approx <- mutate(post_approx, posterior = raw.posterior/norm.constant)
```

We can visualize this by plotting the posterior distribution.

```
ggplot(data=post_approx, aes(x=p,y=posterior)) +
  geom_point()+
  ylim(0,1)
```

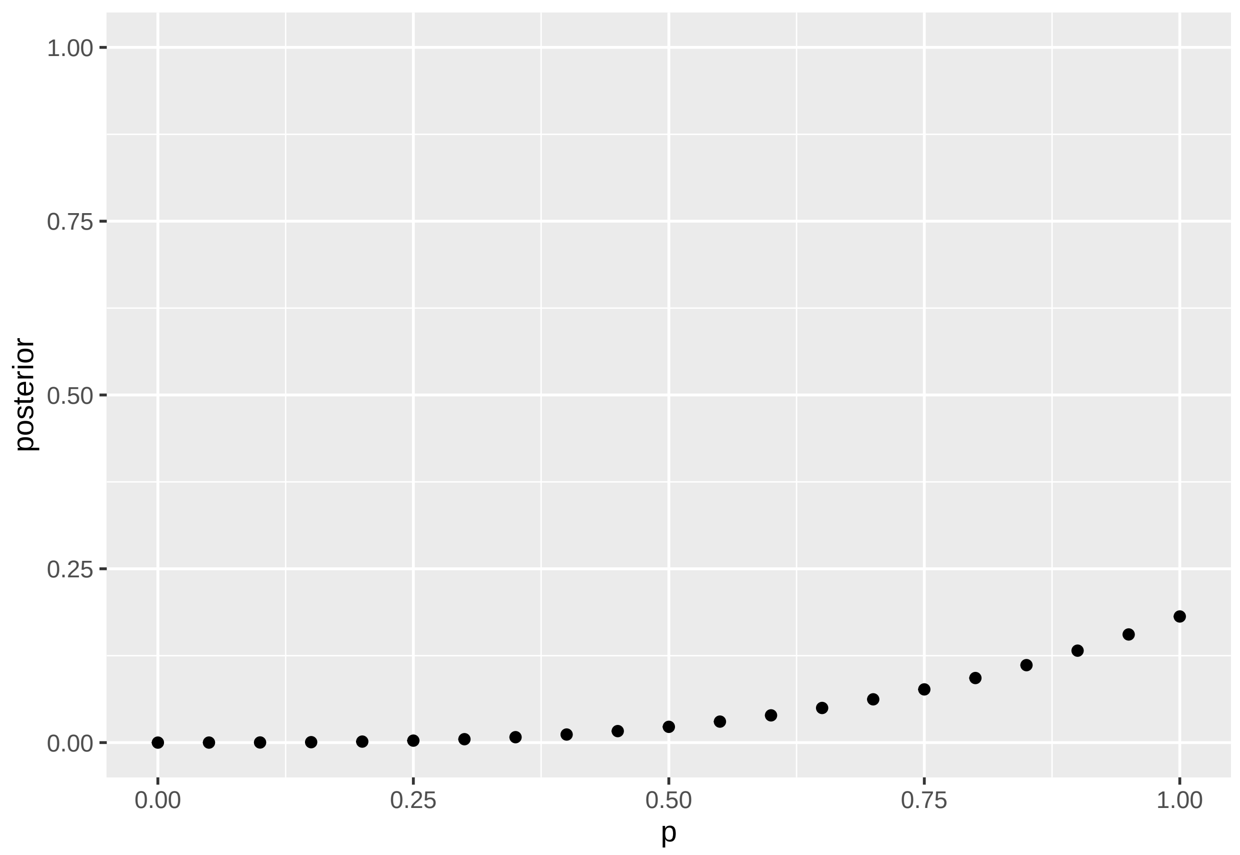


Could we do this all in one step? Yes, the `mutate` function can create multiple new columns in one function call and does this sequentially, as if you had done separate calls to `mutate`.

```
data = c(1,1,1)

post_approx <- tibble(p = seq( from=0 , to=1 , by=0.05 ) ) %>%
  mutate(  likelihood = dbinom( sum(data) , size=length(data) , prob=p),
           prior  = 1,
           raw.posterior=likelihood*prior,
           posterior = raw.posterior/sum(raw.posterior))


ggplot(data=post_approx, aes(x=p,y=posterior))+
  geom_point()+
  ylim(0,1)
```



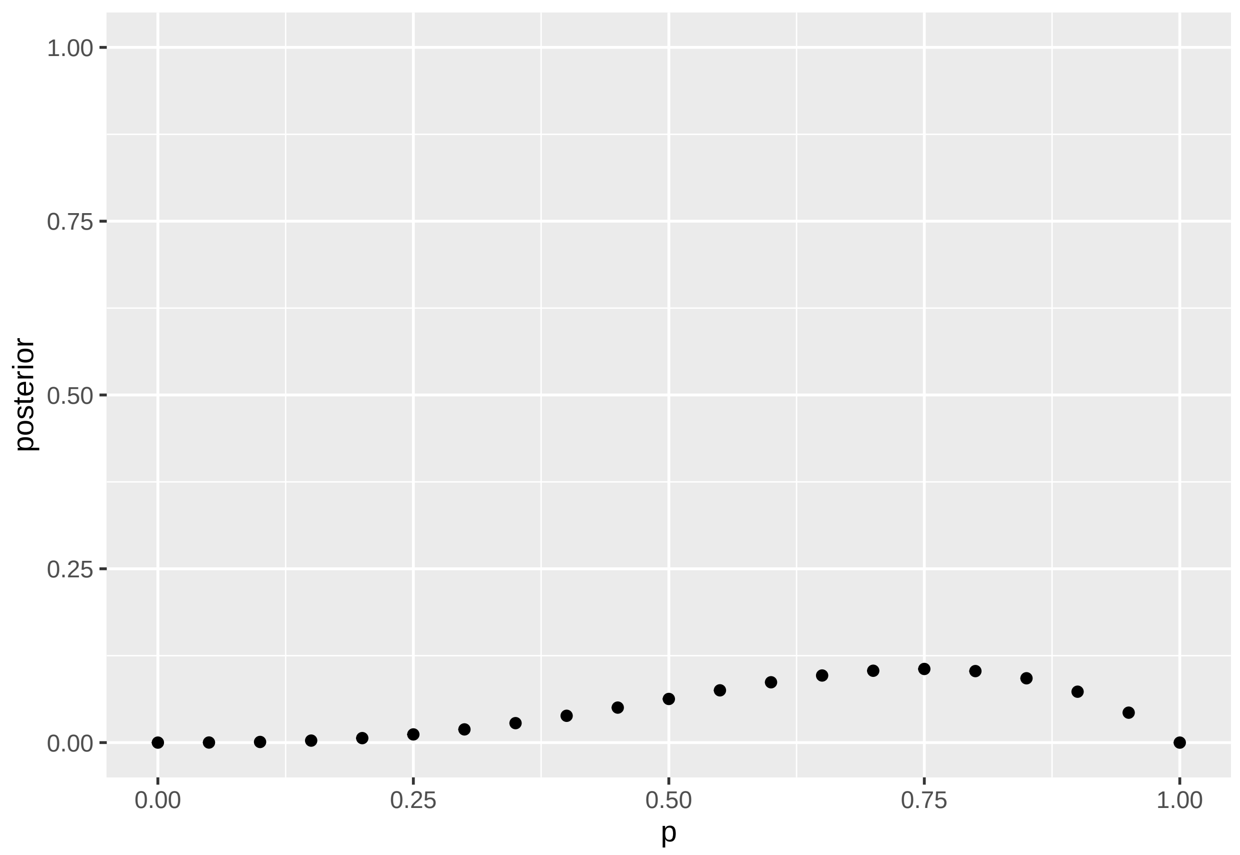
So now we can do this with the other datasets by simply changing the input for `data` and running the rest of the code without making any changes.

(2)

```
data=c(1,1,1,0)

post_approx <- tibble(p = seq( from=0 , to=1 , by=0.05 ) ) %>%
  mutate(  likelihood = dbinom( sum(data) , size=length(data) , prob=p),
           prior  = 1,
           raw.posterior=likelihood*prior,
           posterior = raw.posterior/sum(raw.posterior))


ggplot(data=post_approx, aes(x=p,y=posterior)) +
  geom_point()+
  ylim(0,1)
```

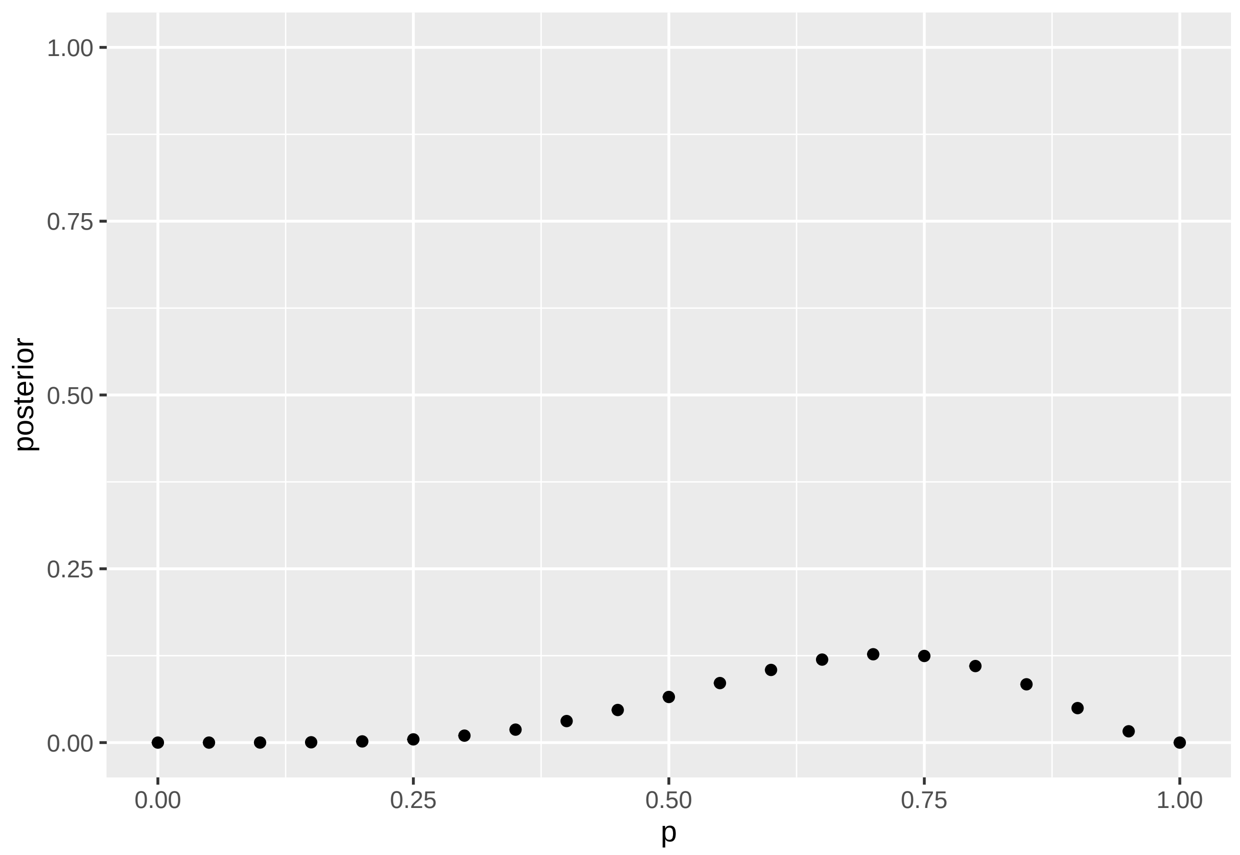


(3)

```
data=c(0,1,1,0,1,1,1)

post_approx <- tibble(p = seq( from=0 , to=1 , by=0.05 ) ) %>%
  mutate(  likelihood = dbinom( sum(data) , size=length(data) , prob=p),
           prior  = 1,
           raw.posterior=likelihood*prior,
           posterior = raw.posterior/sum(raw.posterior))


ggplot(data=post_approx, aes(x=p,y=posterior)) +
  geom_point()+
  ylim(0,1)
```

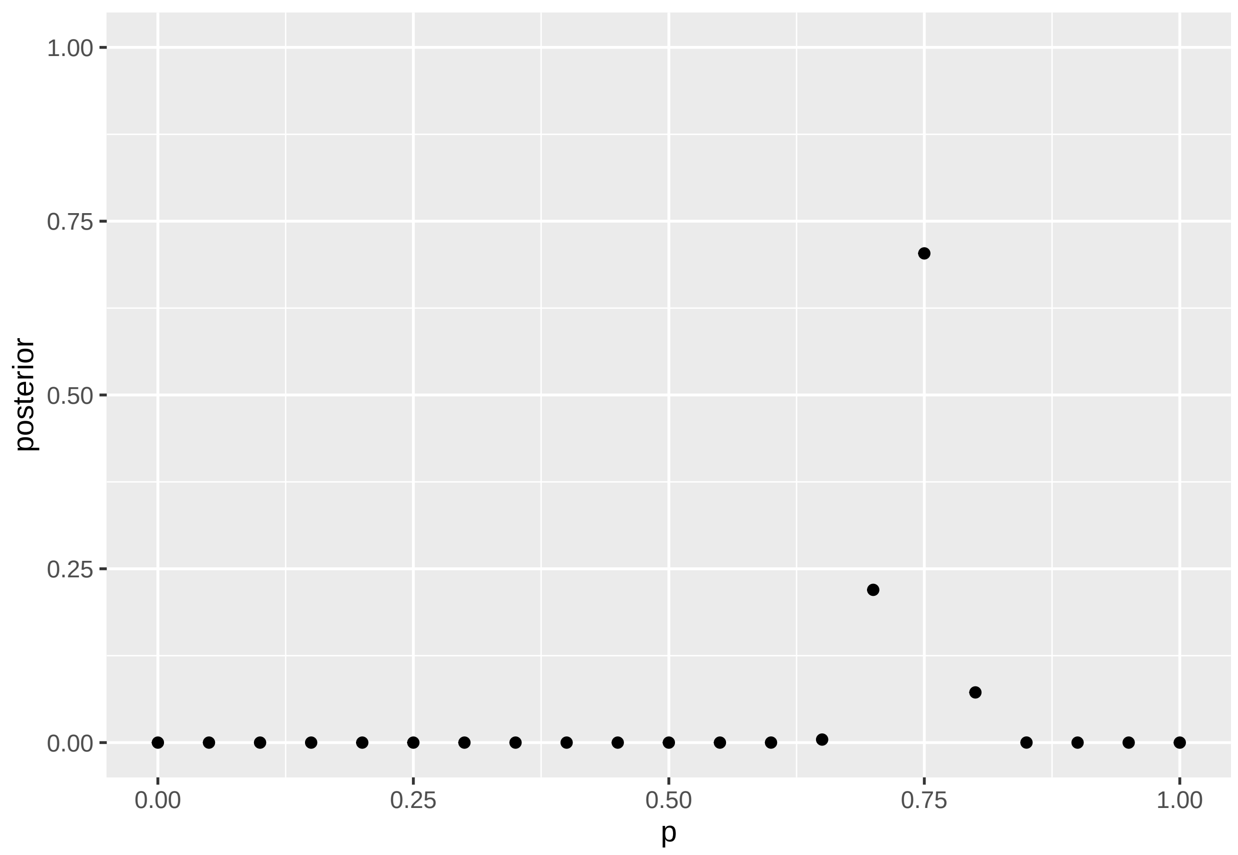


This wasn't assigned, but what if we have much more data? So far, our posterior has fairly broad support, because we only had small samples. Here we'll draw 250 samples from a globe with 0.7 probability of water and plot the posterior. It is much more peaked around the value 0.7.

```
data=rbinom(n=250,size=1,prob=0.7)

post_approx <- tibble(p = seq( from=0 , to=1 , by=0.05 ) ) %>%
  mutate(  likelihood = dbinom( sum(data) , size=length(data) , prob=p),
           prior  = 1,
           raw.posterior=likelihood*prior,
           posterior = raw.posterior/sum(raw.posterior))


ggplot(data=post_approx, aes(x=p,y=posterior)) +
  geom_point()+
  ylim(0,1)
```



2M2

Fortunately we get to re-use most of our code from problem 2M1. Now we use a conditional statement in the `mutate` call for the prior. In R, an inequality that is "true" returns a 1, and "false" returns 0. In other programming languages you might use an `if` statement here, but in R you just use the 1/0 return values to create conditional outputs.

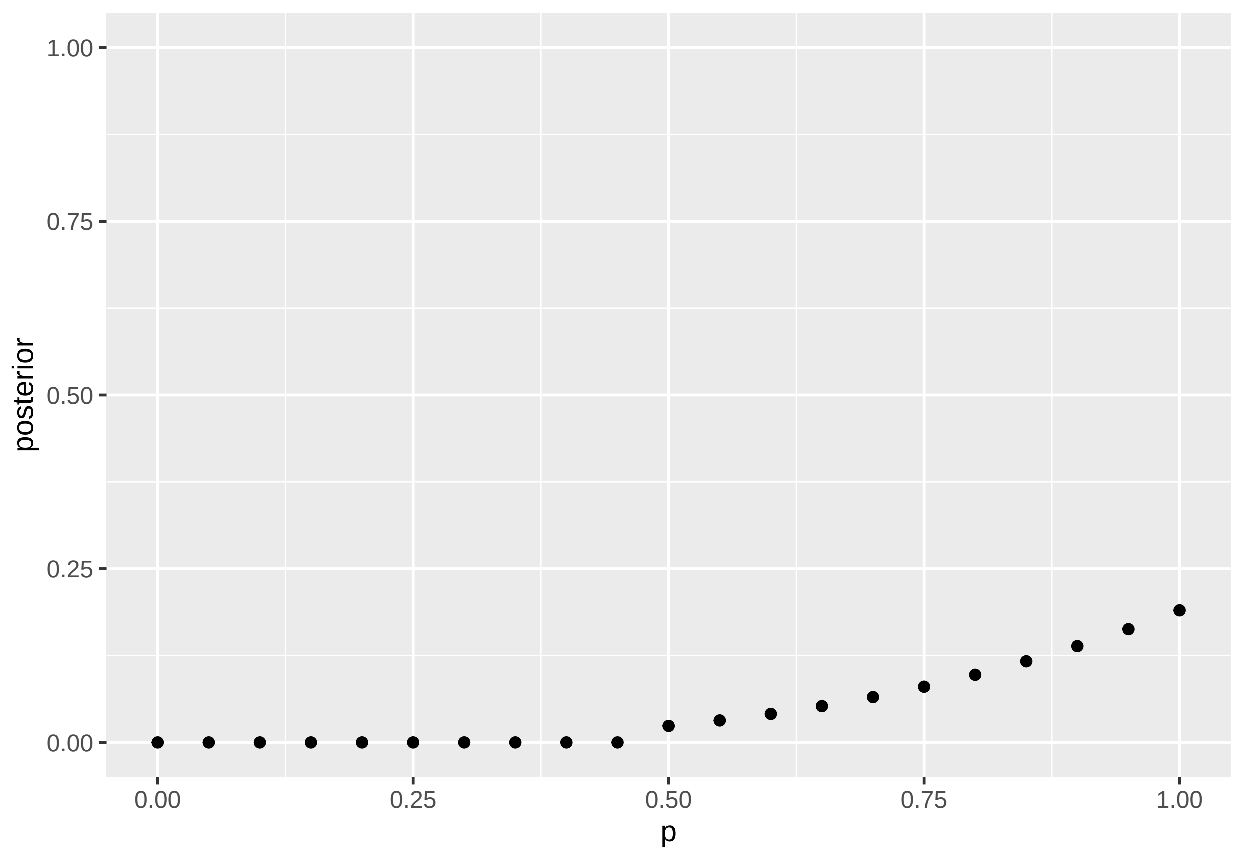
(1)

```r
data = c(1,1,1)


post_approx <- tibble(p = seq( from=0 , to=1 , by=0.05 ) )

post_approx <- tibble(p = seq( from=0 , to=1 , by=0.05 ) ) %>%
  mutate(  likelihood = dbinom( sum(data) , size=length(data) , prob=p),
           prior = (p>=0.5)*1,
           raw.posterior=likelihood*prior,
           posterior = raw.posterior/sum(raw.posterior))


ggplot(data=post_approx, aes(x=p,y=posterior))+
  geom_point()+
  ylim(0,1)
```



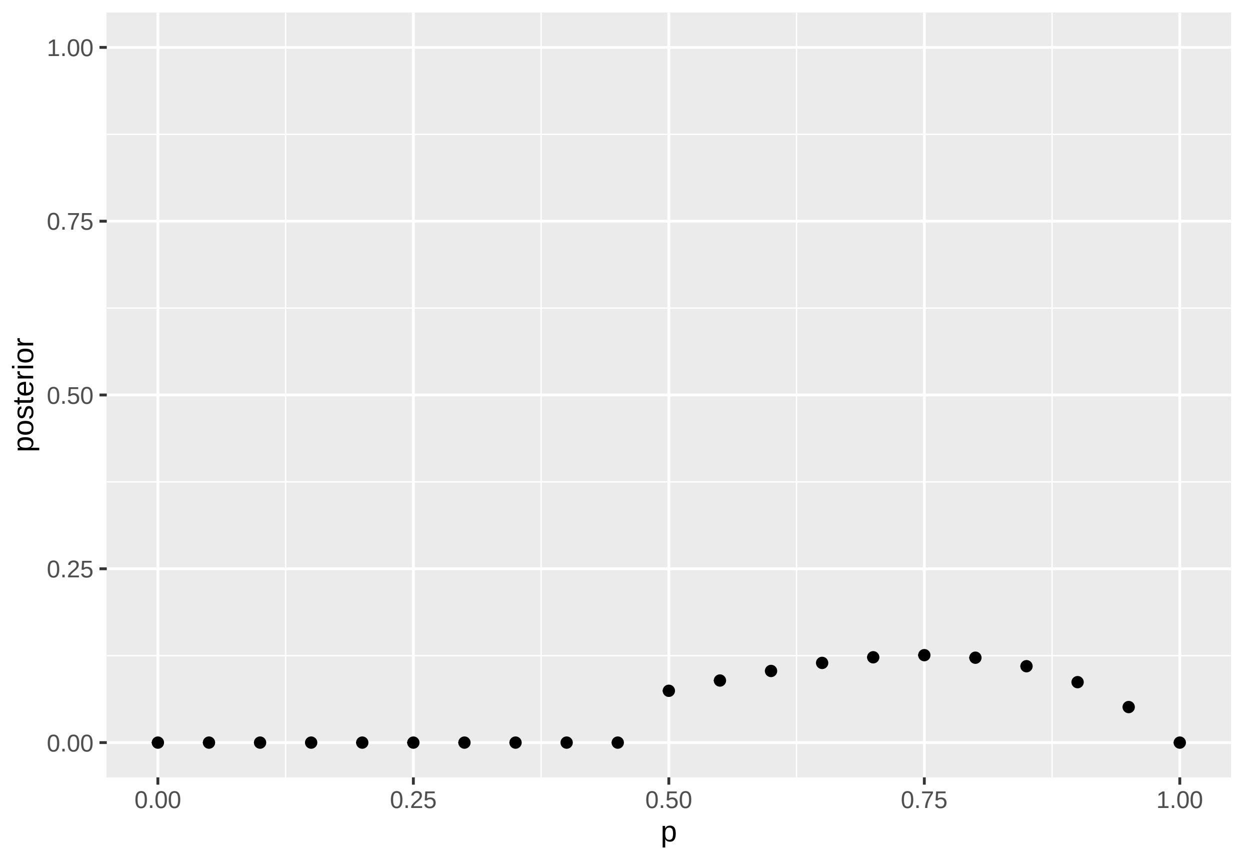
(2)

```
data = c(1,1,1,0)


post_approx <- tibble(p = seq( from=0 , to=1 , by=0.05 ) )

post_approx <- tibble(p = seq( from=0 , to=1 , by=0.05 ) ) %>%
  mutate(  likelihood = dbinom( sum(data) , size=length(data) , prob=p),
           prior = (p>=0.5)*1,
           raw.posterior=likelihood*prior,
           posterior = raw.posterior/sum(raw.posterior))


ggplot(data=post_approx, aes(x=p,y=posterior))+
  geom_point()+
  ylim(0,1)
```



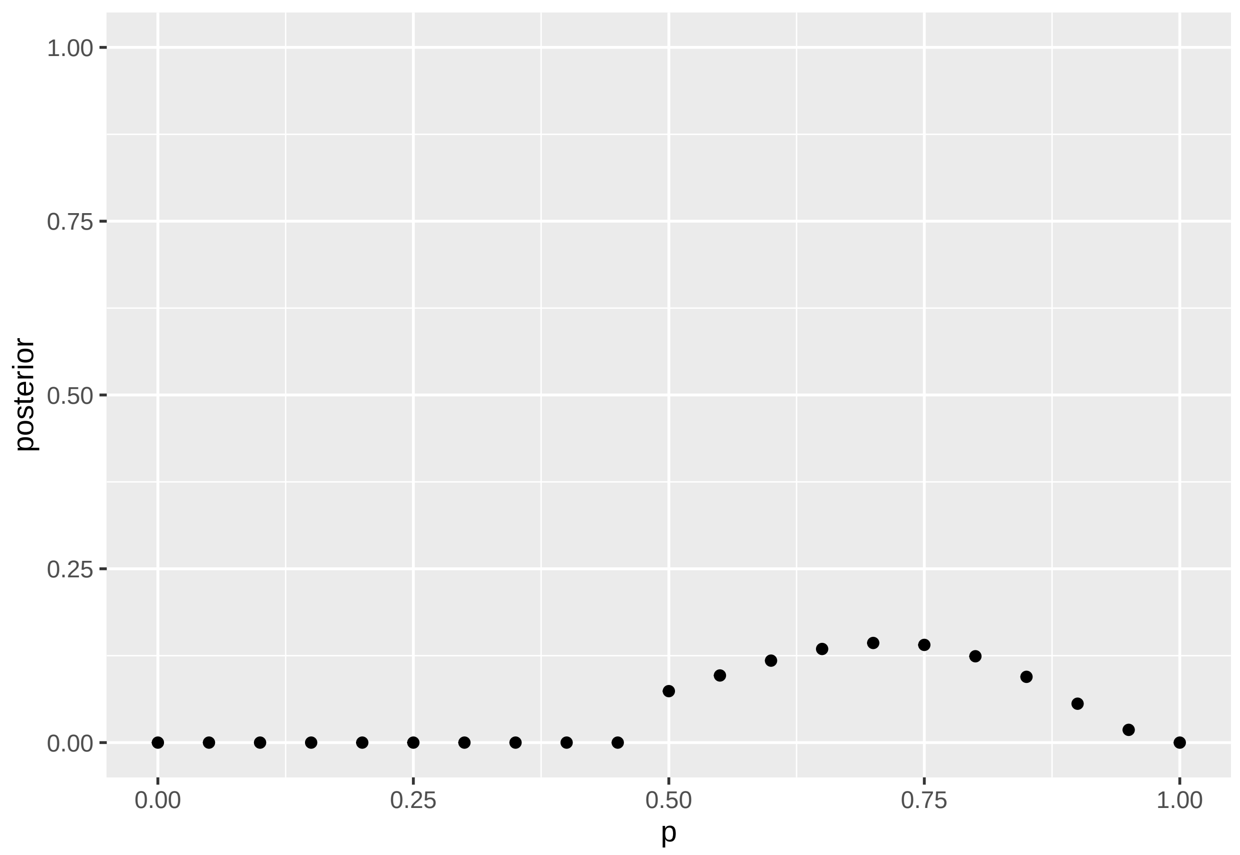
(3)

```
data=c(0,1,1,0,1,1,1)


post_approx <- tibble(p = seq( from=0 , to=1 , by=0.05 ) )

post_approx <- tibble(p = seq( from=0 , to=1 , by=0.05 ) ) %>%
  mutate(  likelihood = dbinom( sum(data) , size=length(data) , prob=p),
           prior = (p>=0.5)*1,
           raw.posterior=likelihood*prior,
           posterior = raw.posterior/sum(raw.posterior))


ggplot(data=post_approx, aes(x=p,y=posterior))+
  geom_point()+
  ylim(0,1)
```



2M3

We can do this using our same updating scheme. Here we have a parameter which is the earth/mars globe. We assign a prior of 0.5 for each globe. The likelihood function is the probability of getting water given a specific globe. The same steps as before can be used to create our posterior table, where we manually code the likelihood as the probability of a 0 (i.e. land) from a binomial with one draw and the probability given by the earth or mars globe.

```
post_tab<- tibble(likelihood=c( dbinom(0,size=1,prob=0.7)  , dbinom(0,size=1,prob=0.0)  ), prior=0.5) %
  mutate(raw.posterior=likelihood*prior, posterior=raw.posterior/sum(raw.posterior))
```

2H1

This problem seems much more complex, but really it just has one additional step from the last problem. Our strategy is :
1. Calculate the posterior probability that we have species A or B.
2. Calculate the probability of twins given the posterior probability of species A and B.

Let's code twins as 1.

```
post_tab<- tibble(species = c("A","B"),p=c(0.1,0.2), prior=0.5) %>%
  mutate(likelihood = dbinom(1,size=1,prob=p),raw.posterior=likelihood*prior, posterior=raw.posterior/s
```

And let's look at our table:

```r
post_tab
```

```
## # A tibble: 2 x 6
##   species     p prior likelihood raw.posterior posterior
##   <chr>   <dbl> <dbl>      <dbl>         <dbl>     <dbl>
## 1 A         0.1   0.5        0.1          0.05     0.333
## 2 B         0.2   0.5        0.2          0.1      0.667
```

We see that there is a 1/3 chance it is species A and a 2/3 chance it is species B.

Now we use the probability of twinning for each respective species and sum them.

```r
post_tab <- mutate(post_tab,p.twins = posterior*p)

sum(post_tab$p.twins)
```

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