Here we calculate the posterior distribution given $n = 20$ and $y = 8$:

```r
#Entering data and the prior probabilities
priorvalues <- c(0, .1, .2, .3, .4, .5, .6, .7, .8, .9, 1)
priorprob <- c(1/23, 1/23, 7/23, 7/23, 3/23, 3/23, 1/23, 0/23, 0/23, 0/23, 0/23)

n <- 20
y <- 8

#vector for storing results
jointprob <- numeric(length = length(priorvalues))

for(i in 1:length(priorvalues))
{

  #compute Binomial probability given value of p - likelihood
  binomprob <- dbinom(y, n, p = priorvalues[i])

  #compute joint probability - posterior
  jointprob[i] <- binomprob * priorprob[i]

}

#compute marginal probability of y
pofy <- sum(jointprob)

#compute posterior probabilities
posteriorprob <- jointprob/pofy

#visualizing the posterior

#put posterior probabilities in one matrix object for easy viewing
allnighterposterior <- as.data.frame(cbind(priorvalues, priorprob, posteriorprob))
names(allnighterposterior) <- c("p", "prior", "posterior")

#list the final posterior distribution, based on our prior derived in class
allnighterposterior
```

```
##      p      prior    posterior
## 1  0.0 0.04347826 0.0000000000
## 2  0.1 0.04347826 0.0001881144
## 3  0.2 0.30434783 0.0820219042
## 4  0.3 0.30434783 0.4234055597
## 5  0.4 0.13043478 0.2850546884
## 6  0.5 0.13043478 0.1905607002
## 7  0.6 0.04347826 0.0187690330
## 8  0.7 0.00000000 0.0000000000
## 9  0.8 0.00000000 0.0000000000
## 10 0.9 0.00000000 0.0000000000
## 11 1.0 0.00000000 0.0000000000
```

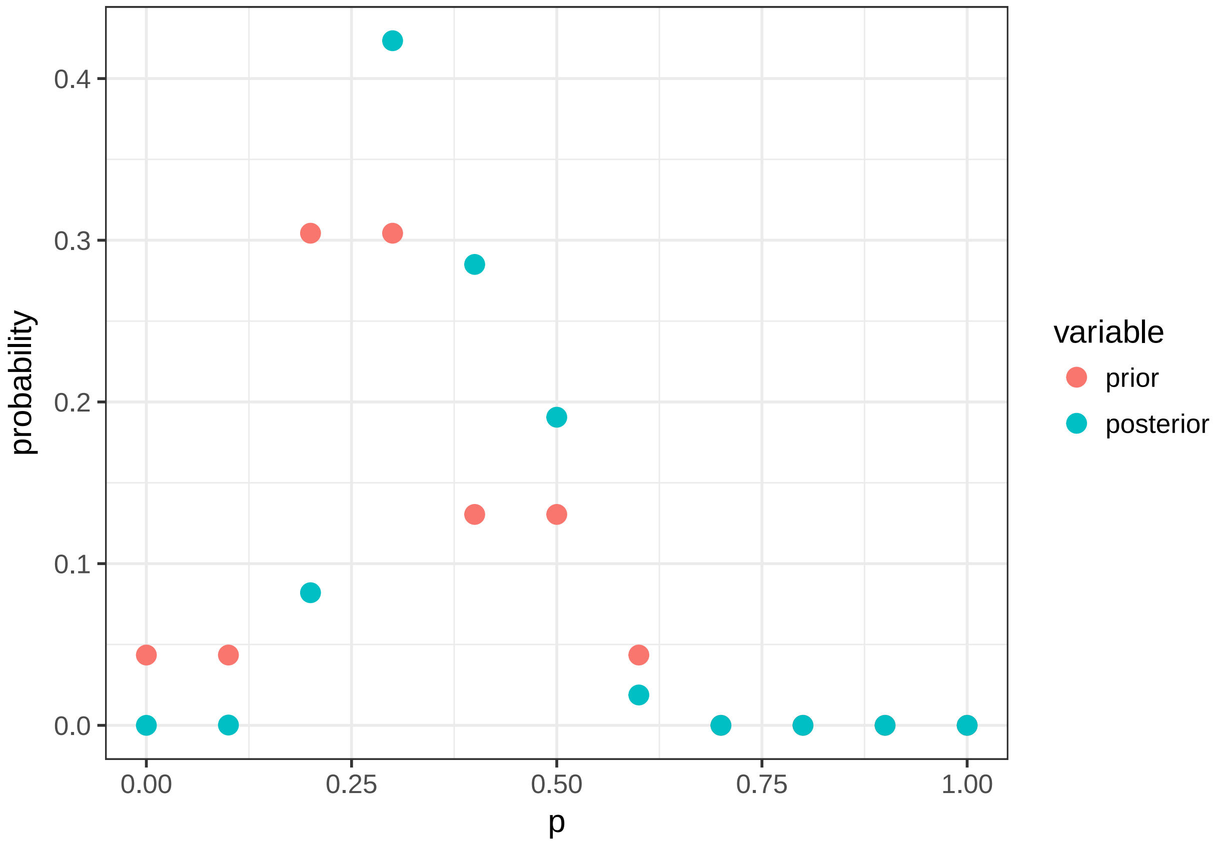
```r
#plot the prior and posterior probabilities
require(ggplot2)
```

```
## Loading required package: ggplot2
```

```r
require(reshape2)
```

```
## Loading required package: reshape2
```

```r
allnighterposterior_all <- melt(allnighterposterior, id = "p")

ggplot(allnighterposterior_all, aes(x = p, y = value, colour = variable)) +
  geom_point(size = 3) +
  xlab("p") + ylab("probability") +
 theme_bw(base_size = 12, base_family = "")
```




```r
allnighterposterior
```

```
##      p      prior    posterior
## 1  0.0 0.04347826 0.0000000000
## 2  0.1 0.04347826 0.0001881144
## 3  0.2 0.30434783 0.0820219042
## 4  0.3 0.30434783 0.4234055597
## 5  0.4 0.13043478 0.2850546884
## 6  0.5 0.13043478 0.1905607002
## 7  0.6 0.04347826 0.0187690330
## 8  0.7 0.00000000 0.0000000000
## 9  0.8 0.00000000 0.0000000000
## 10 0.9 0.00000000 0.0000000000
```

```
## 11 1.0 0.00000000 0.0000000000
```

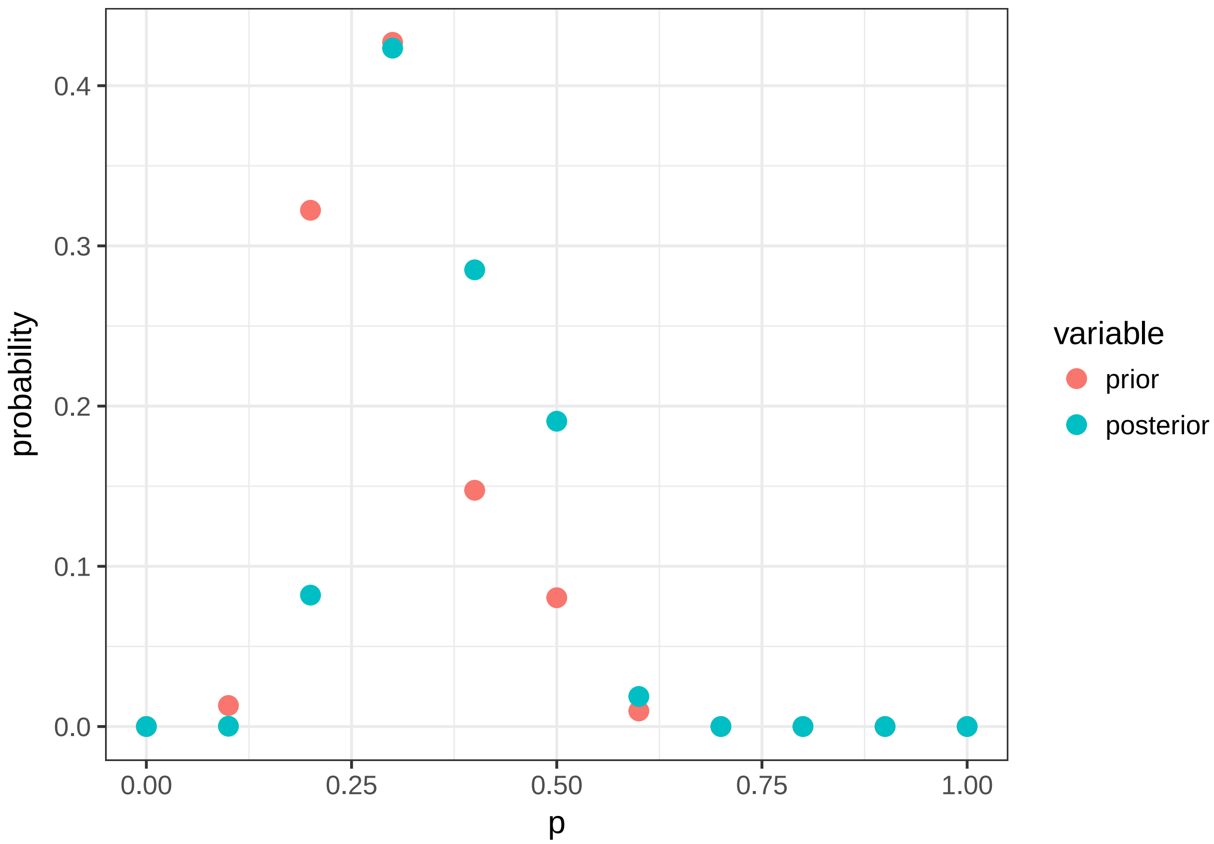
In the following code, we assign the posterior calculated in class as prior and compute the posterior for $n = 10$ and $y = 5$.

```
#Calculating posterior (n=10, y=3) from the prior distribution given in class

priorvalues <- c(0, .1, .2, .3, .4, .5, .6, .7, .8, .9, 1)
priorprob <- c(1/23, 1/23, 7/23, 7/23, 3/23, 3/23, 1/23, 0/23, 0/23, 0/23, 0/23)

n <- 10
y <- 3

#vector for storing results
jointprob <- numeric(length = length(priorvalues))

for(i in 1:length(priorvalues))
{

  #compute Binomial probability given value of p - likelihood
  binomprob <- dbinom(y, n, p = priorvalues[i])

  #compute joint probability - posterior
  jointprob[i] <- binomprob * priorprob[i]

}

#compute marginal probability of y
pofy <- sum(jointprob)

#compute posterior probabilities
posteriorprob <- jointprob/pofy


#Now we will do a sequential update by settting the prior probabilities to the computed posterior proba

priorprob <- posteriorprob
n <- 10
y <- 5


jointprob <- numeric(length = length(priorvalues))

for(i in 1:length(priorvalues))
{

  #compute Binomial probability given value of p - likelihood
  binomprob <- dbinom(y, n, p = priorvalues[i])

  #compute joint probability - posterior
  jointprob[i] <- binomprob * priorprob[i]

}

#compute marginal probability of y
```

```r
pofy <- sum(jointprob)

#compute posterior probabilities
posteriorprob <- jointprob/pofy
allnighterposterior <- as.data.frame(cbind(priorvalues, priorprob, posteriorprob))
names(allnighterposterior) <- c("p", "prior", "posterior")

#list the final posterior distribution, based on our prior derived in class
allnighterposterior
```

```
##      p       prior    posterior
## 1  0.0 0.000000000 0.0000000000
## 2  0.1 0.013123561 0.0001881144
## 3  0.2 0.322234525 0.0820219042
## 4  0.3 0.427073101 0.4234055597
## 5  0.4 0.147473543 0.2850546884
## 6  0.5 0.080385077 0.1905607002
## 7  0.6 0.009710192 0.0187690330
## 8  0.7 0.000000000 0.0000000000
## 9  0.8 0.000000000 0.0000000000
## 10 0.9 0.000000000 0.0000000000
## 11 1.0 0.000000000 0.0000000000
```

```r
#plot the prior and posterior probabilities
require(ggplot2)
require(reshape2)

allnighterposterior_all <- melt(allnighterposterior, id = "p")

ggplot(allnighterposterior_all, aes(x = p, y = value, colour = variable)) +
  geom_point(size = 3) +
  xlab("p") + ylab("probability") +
 theme_bw(base_size = 12, base_family = "")
```




```
allnighterposterior
```

```
##      p       prior    posterior
## 1  0.0 0.000000000 0.0000000000
## 2  0.1 0.013123561 0.0001881144
## 3  0.2 0.322234525 0.0820219042
## 4  0.3 0.427073101 0.4234055597
## 5  0.4 0.147473543 0.2850546884
## 6  0.5 0.080385077 0.1905607002
## 7  0.6 0.009710192 0.0187690330
## 8  0.7 0.000000000 0.0000000000
## 9  0.8 0.000000000 0.0000000000
## 10 0.9 0.000000000 0.0000000000
## 11 1.0 0.000000000 0.0000000000
```

Thus we see that the posterior with the bigger data set calculated in one update is the same as the posterior calculated by sequential updates of 10 at a time. This is because we assume that the trials are independent so the fact that out of 10 people 3 stayed up last year does not affect the fact that out of 10 other people, 5 stayed up last year.