Fundamentals of Bayesian Data Analysis in R

for Exercises

Eszter Katalin Bognar

16-06-2020

Fundamentals of Bayesian Data Analysis in R

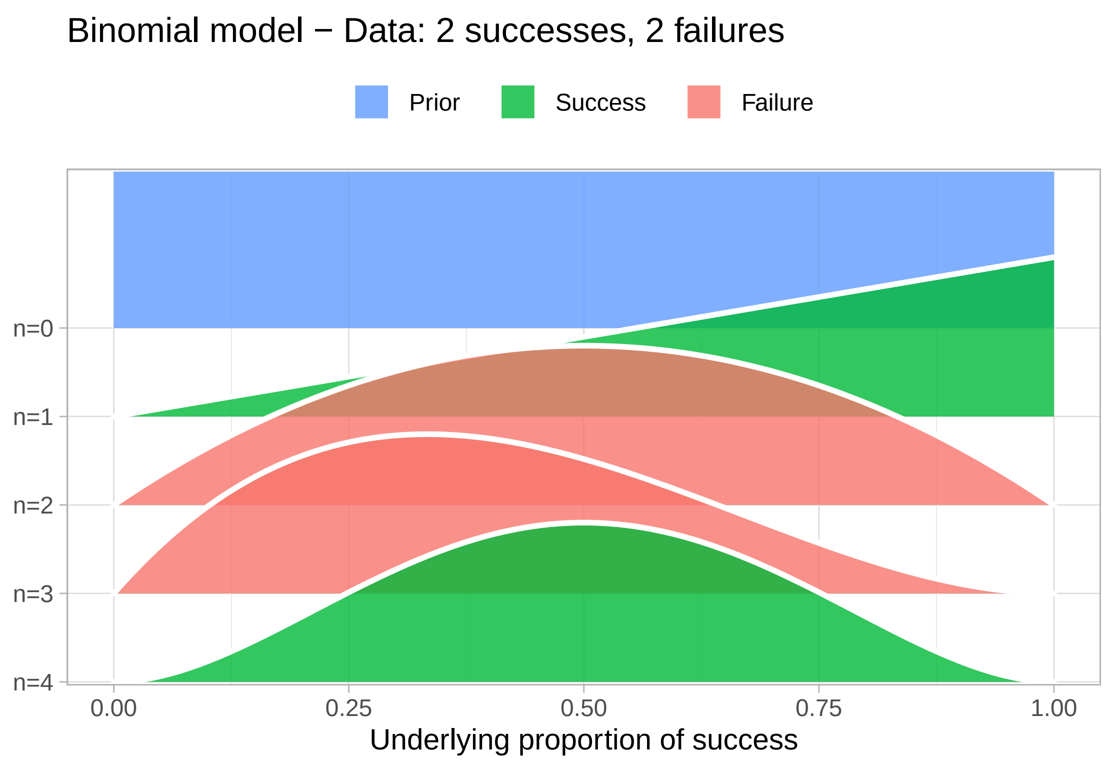
The prop_model function

```
prop_model <- function(data = c(), prior_prop = c(1, 1), n_draws = 10000) {
  library(tidyverse)
  data <- as.logical(data)
  # data_indices decides what densities to plot between the prior and the posterior
  # For 20 datapoints and less we're plotting all of them.
  data_indices <- round(seq(0, length(data),
                            length.out = min(length(data) + 1, 20)))

  # dens_curves will be a data frame with the x & y coordinates for the
  # denities to plot where x = proportion_success and y = probability
  proportion_success <- c(0, seq(0, 1, length.out = 100), 1)
  dens_curves <- map_dfr(data_indices, function(i) {
    value <- ifelse(i == 0, "Prior", ifelse(data[i], "Success", "Failure"))
    label <- paste0("n=", i)
    probability <- dbeta(proportion_success,
                         prior_prop[1] + sum(data[seq_len(i)]),
                         prior_prop[2] + sum(!data[seq_len(i)]))
    probability <- probability / max(probability)
    data_frame(value, label, proportion_success, probability)
  })
  # Turning label and value into factors with the right ordering for the plot
  dens_curves$label <- fct_rev(factor(dens_curves$label,
                                      levels =  paste0("n=", data_indices )))
  dens_curves$value <- factor(dens_curves$value,
                              levels = c("Prior", "Success", "Failure"))

  p <- ggplot(dens_curves, aes(x = proportion_success, y = label,
                               height = probability, fill = value)) +
    ggridges::geom_density_ridges(stat="identity", color = "white", alpha = 0.8,
                                  panel_scaling = TRUE, size = 1) +
    scale_y_discrete("", expand = c(0.01, 0)) +
    scale_x_continuous("Underlying proportion of success") +
    scale_fill_manual(values = hcl(120 * 2:0 + 15, 100, 65), name = "", drop = FALSE,
                      labels =  c("Prior   ", "Success   ", "Failure   ")) +
    ggtitle(paste0(
      "Binomial model - Data: ", sum(data),
      " successes, " , sum(!data), " failures")) +
    theme_light() +
    theme(legend.position = "top")
  print(p)

  # Returning a sample from the posterior distribution that can be further
  # manipulated and inspected
  posterior_sample <- rbeta(n_draws, prior_prop[1] +
                              sum(data), prior_prop[2] + sum(!data))
  invisible(posterior_sample)
}
```

Coin flips with prop_model

```
data <- c(1, 0, 0, 1)
prop_model(data)
```

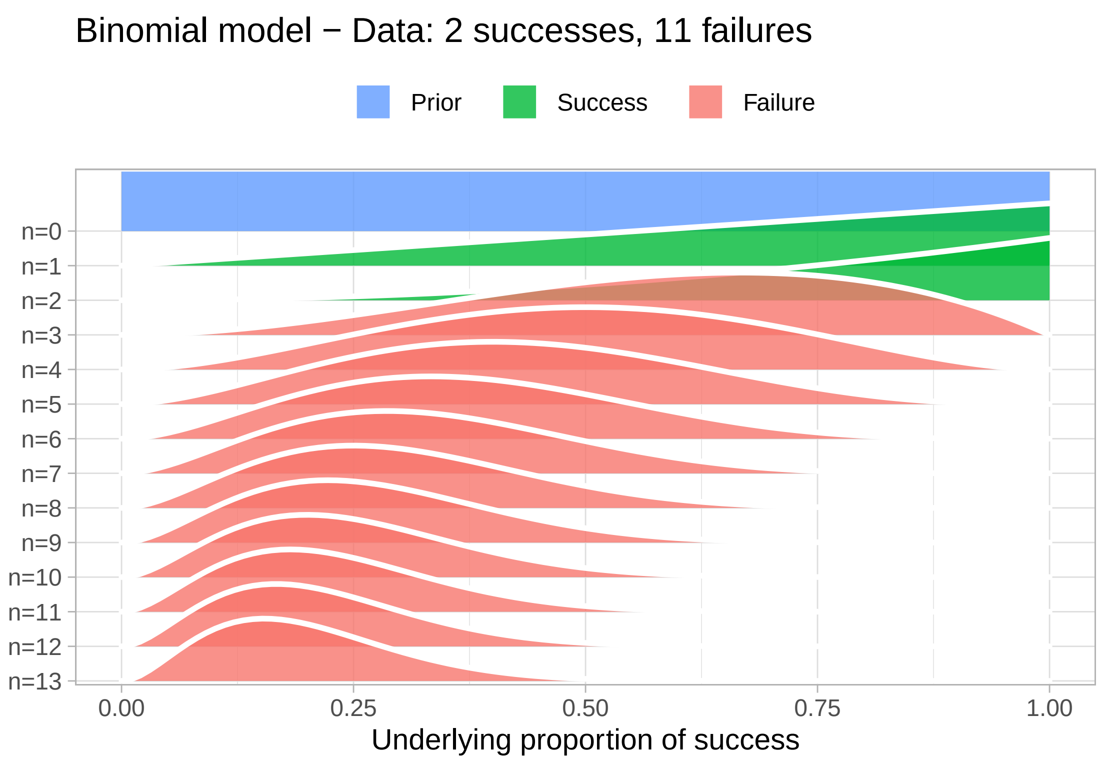
```
## -- Attaching packages -----------------------------------------------------------------------------------------

## v ggplot2 3.3.0     v purrr   0.3.4
## v tibble  3.0.0     v dplyr   0.8.5
## v tidyr   1.0.2     v stringr 1.4.0
## v readr   1.3.1     v forcats 0.5.0

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

## Warning: `data_frame()` is deprecated as of tibble 1.1.0.
## Please use `tibble()` instead.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_warnings()` to see where this warning was generated.
```



```
#the information that the model have regarding the underlying proportion of heads:
#It's most likely around 50%, but there is large uncertainty.
```

Zombie drugs with prop_model

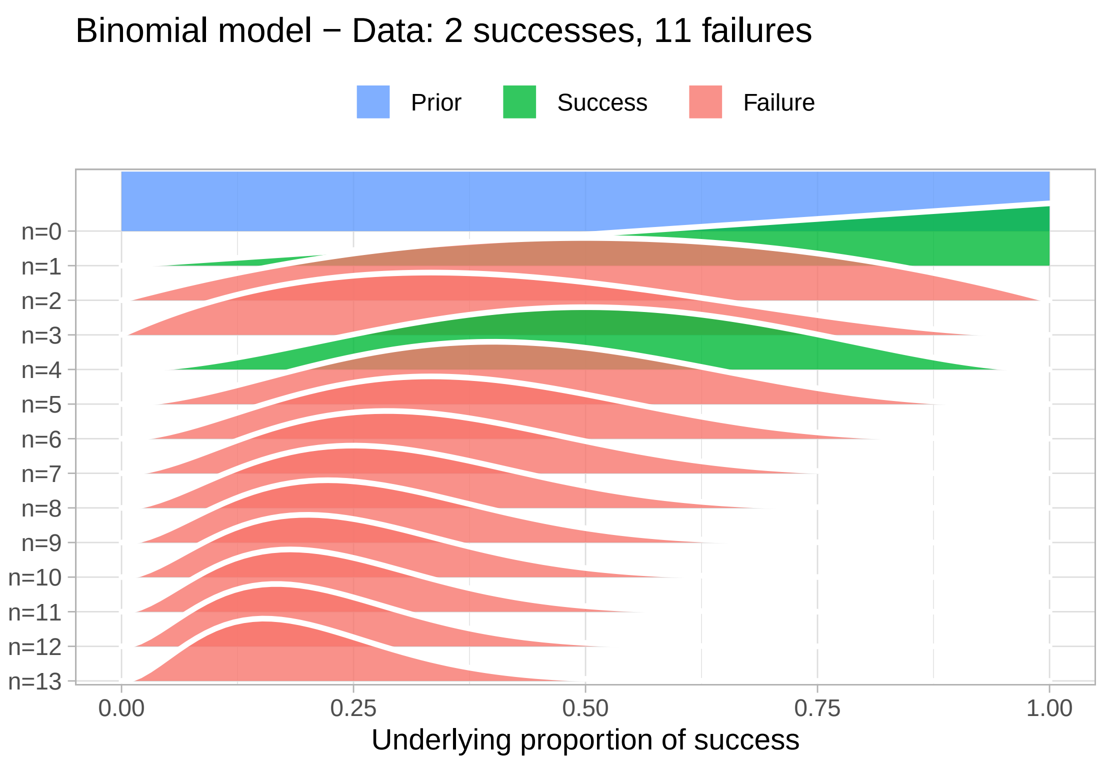
```
# Update the data and rerun prop_model
data = c(1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)
prop_model(data)
```



```
#the information that the model have regarding the underlying proportion of heads:
#It's most likely Between 5% to 40%.
```

Samples and posterior summaries

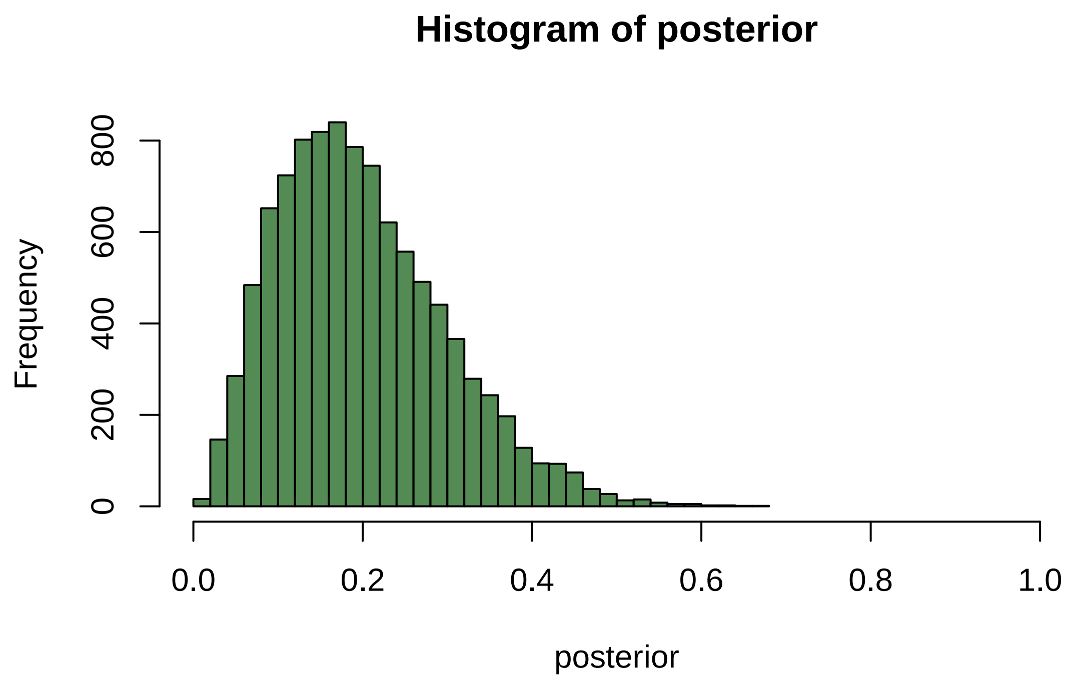
```
data = c(1, 0, 0, 1, 0, 0,
         0, 0, 0, 0, 0, 0, 0)

# Extract and explore the posterior
posterior <- prop_model(data)
```



```r
head(posterior)
```

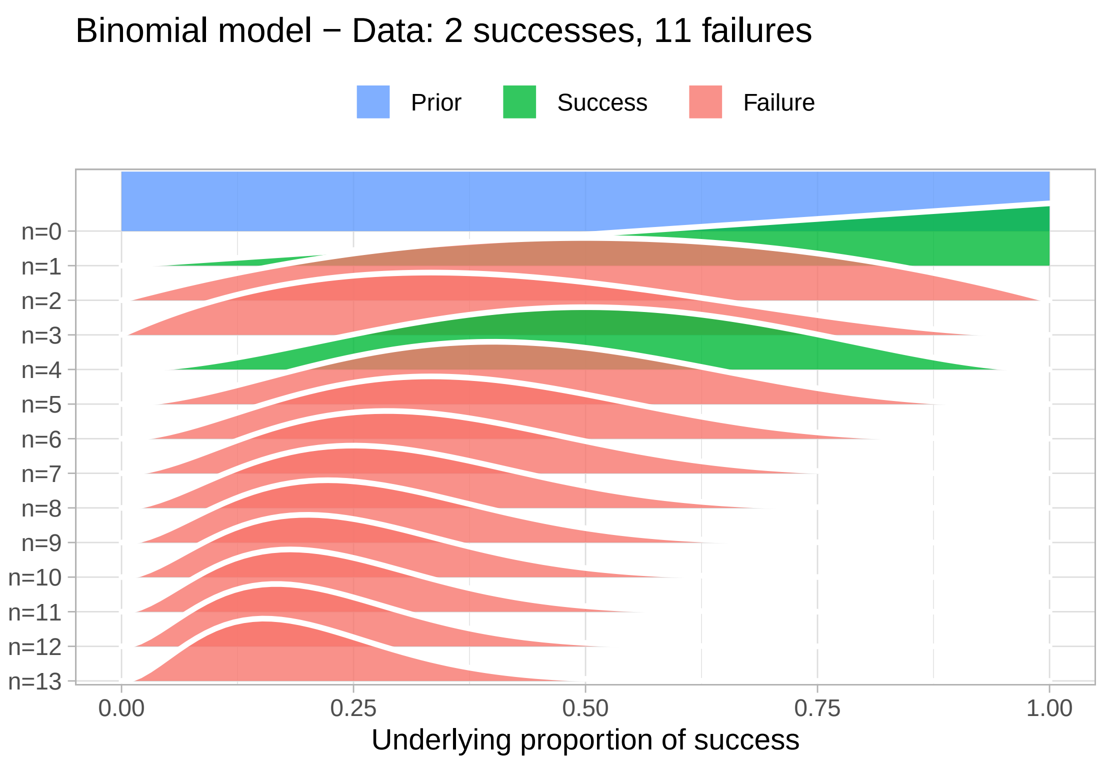
```
## [1] 0.08879642 0.34965419 0.37395627 0.11864239 0.05067127 0.02754691
```

```r
# Edit the histogram
hist(posterior,breaks = 30,xlim = c(0, 1),col = "palegreen4")
```

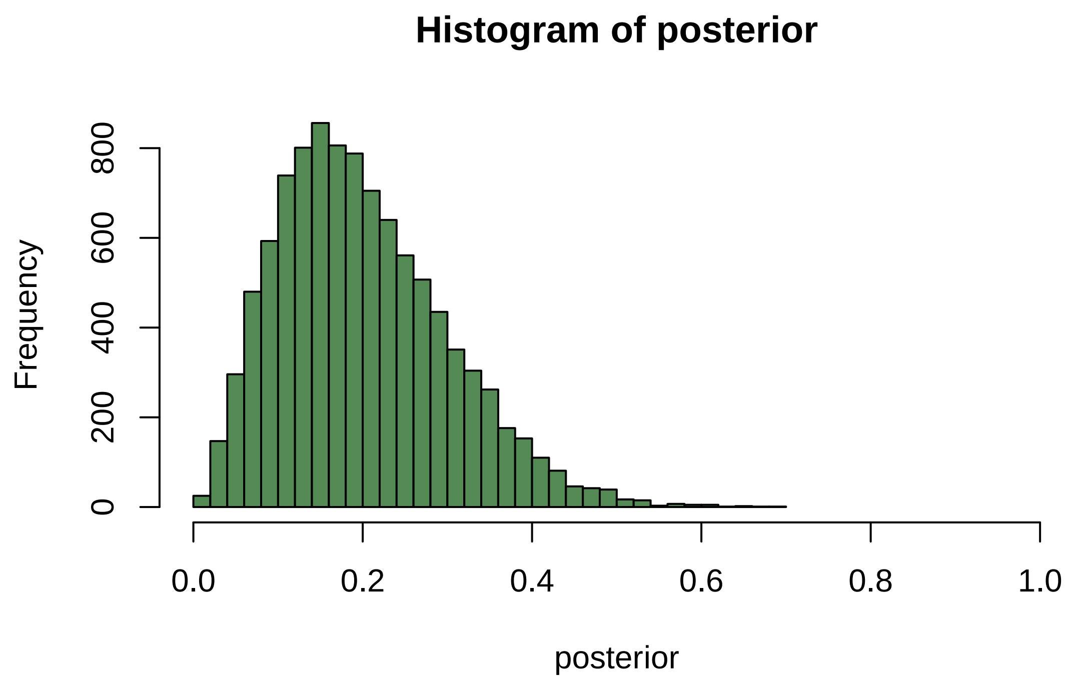


Summarizing the zombie drug experiment

```
data = c(1, 0, 0, 1, 0, 0,
         0, 0, 0, 0, 0, 0, 0)
posterior <- prop_model(data)
```



```r
#drawing histogram
hist(posterior, breaks = 30, xlim = c(0, 1), col = "palegreen4")
```



```
# Calculate the median
median(posterior)
```

```
## [1] 0.186541
```

```
# Calculate the credible interval
quantile(posterior, c(0.05, 0.95))
```

```
##         5%        95%
## 0.06204288 0.38312000
```

```
# Calculate the probability
sum(posterior > 0.07) / length(posterior)
```

```
## [1] 0.9321
```

How does Bayesian inference work?

Take a generative model for a spin

```
# The generative zombie drug model
# Set parameters
prop_success <- sum(data)
n_zombies <- 100
# Simulating data
data <- c()
for(zombie in 1:n_zombies) {
  data[zombie] <- runif(1, min = 0, max = 1) < prop_success
}
# Count cured
data <- prop_success
data
```

```
## [1] 2
```

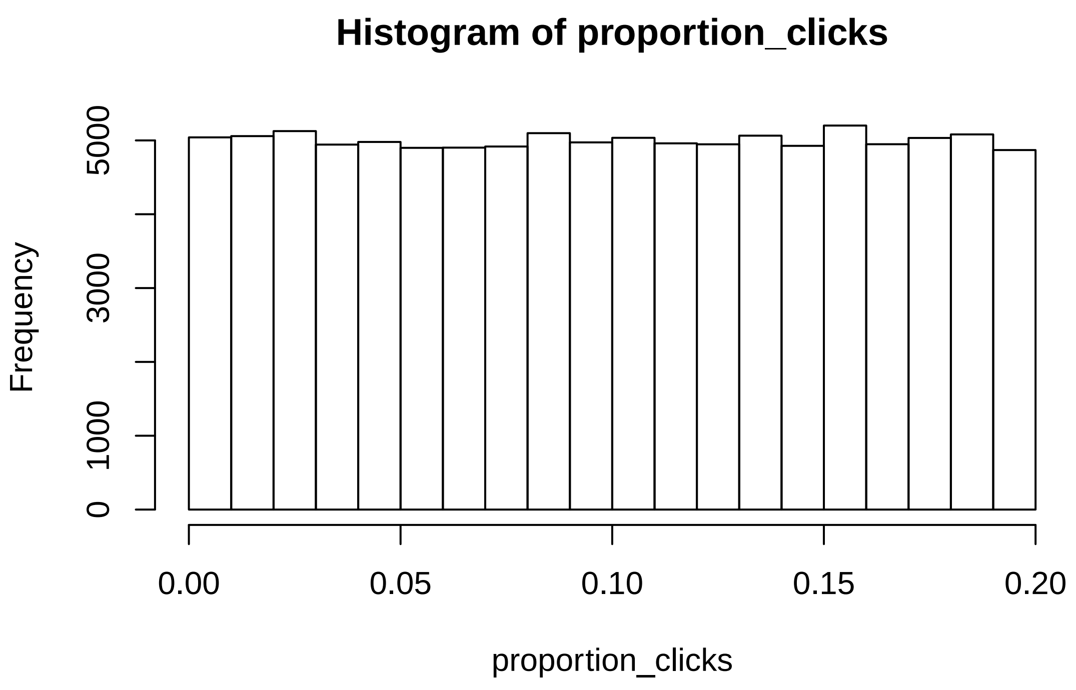
Take the binomial distribution for a spin

```
# n The number of times you want to run the generative model
#  size The number of trials. (For example, the number of zombies you're giving the drug.)
#  prop The underlying proportion of success as a number between 0.0 and 1.0.
rbinom(n = 200, size = 100, prob = 0.42)
```

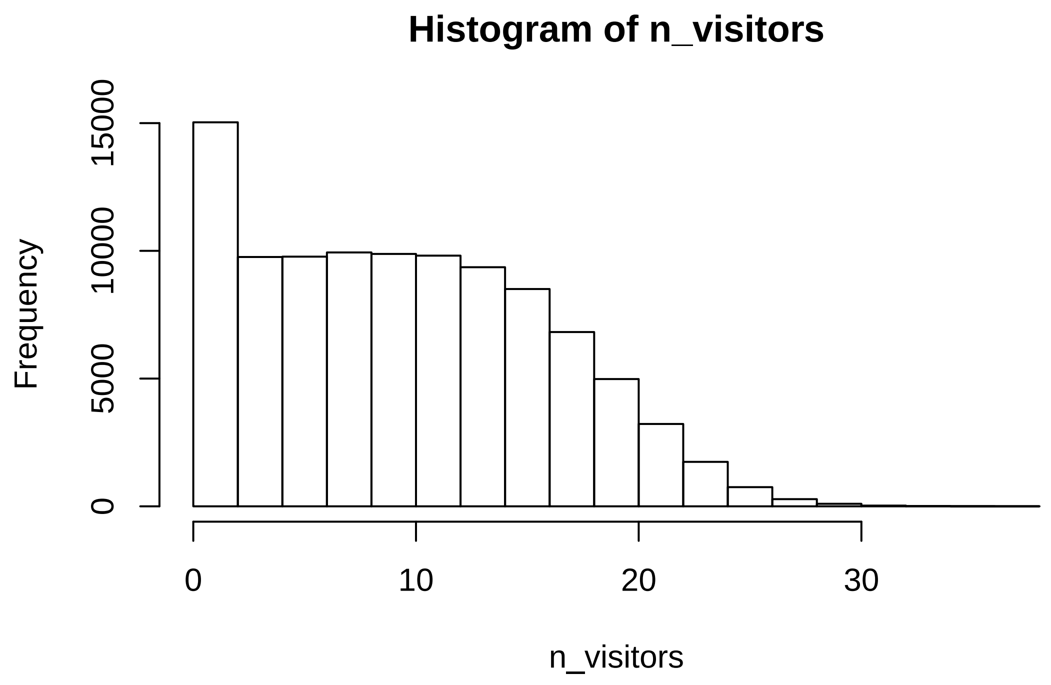
```
##   [1] 44 44 48 39 43 44 33 44 40 47 48 37 38 41 43 44 48 43 39 35 42 41 49 43 40
##  [26] 44 46 39 41 55 41 41 36 44 41 36 46 38 46 41 49 44 43 38 49 53 39 47 40 45
##  [51] 39 40 41 47 45 47 49 46 42 45 41 48 41 44 42 42 46 47 44 38 45 36 50 42 40
##  [76] 48 48 41 46 35 47 38 55 40 45 46 47 39 41 38 41 34 44 51 36 44 36 39 38 46
## [101] 48 44 52 38 38 37 44 39 39 50 41 39 38 49 38 43 38 42 38 40 45 40 43 45 49
## [126] 36 41 44 36 42 44 39 33 43 48 38 46 41 46 39 39 42 42 44 43 50 39 43 41 42
## [151] 45 40 38 47 47 42 42 41 32 39 44 35 35 36 39 49 38 34 37 40 38 32 43 42 49
## [176] 47 44 42 51 42 36 44 34 43 44 49 37 41 39 41 30 53 34 43 45 36 49 46 55 34
```

Adding a prior to the model

```r
n_samples <- 100000
n_ads_shown <- 100
proportion_clicks <- runif(n_samples, min = 0.0, max = 0.2)
n_visitors <- rbinom(n = n_samples, size = n_ads_shown, prob = proportion_clicks)

# Visualize proportion clicks
hist(proportion_clicks)
```



```r
# Visualize n_visitors
hist(n_visitors)
```



Bayesian models and conditioning

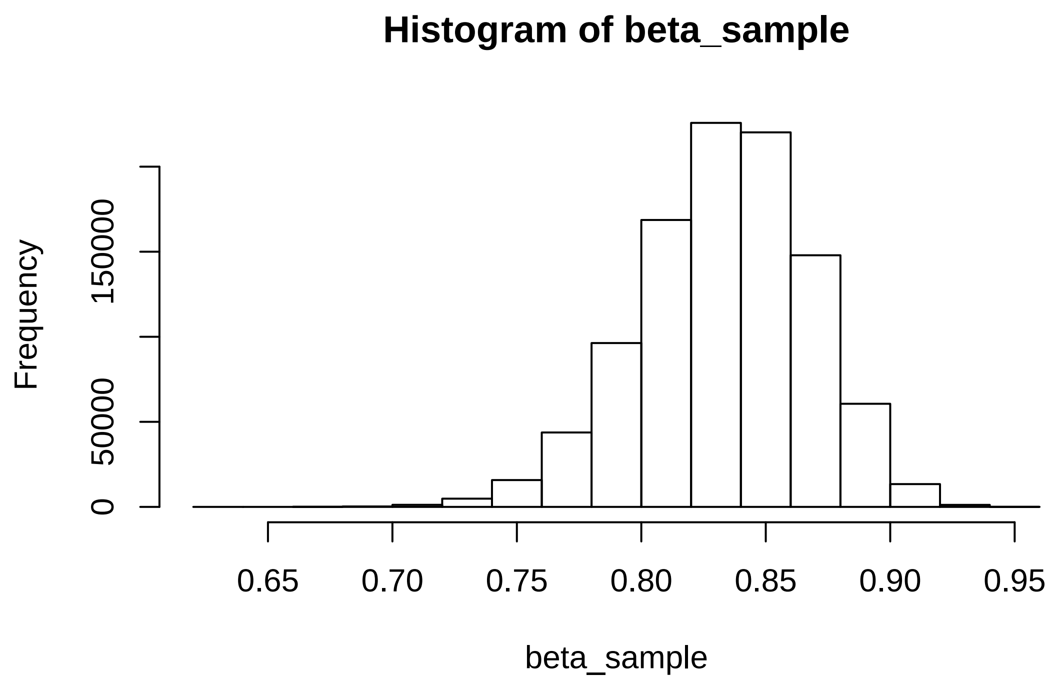
```r
# Assign posterior to a new variable called prior
prior <- posterior

# Take a look at the first rows in prior
head(prior)

n_samples <-  nrow(prior)
n_ads_shown <- 100

prior$n_visitors <- rbinom(n_samples, size = n_ads_shown, prob = prior$proportion_clicks)

hist(prior$n_visitors)

# Calculate the probability that you will get 5 or more visitors
sum(prior$n_visitors >= 5) / length(prior$n_visitors)
```

Why use Bayesian Data Analysis?

Explore using the Beta distribution as a prior

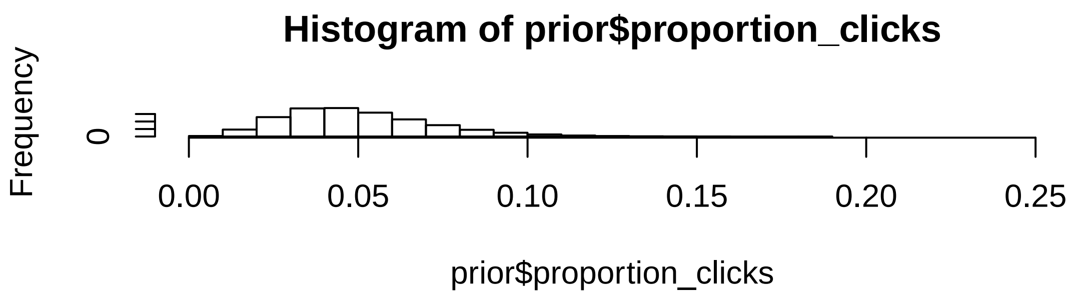
```r
# Modify the parameters
beta_sample <- rbeta(n = 1000000, shape1 = 100, shape2 = 20)
```

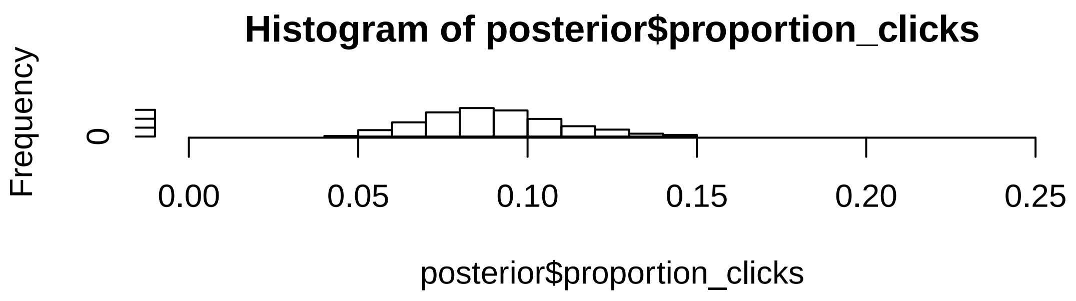
```r
# Visualize the results
hist(beta_sample)
```



## Using a prior with Beta distribution

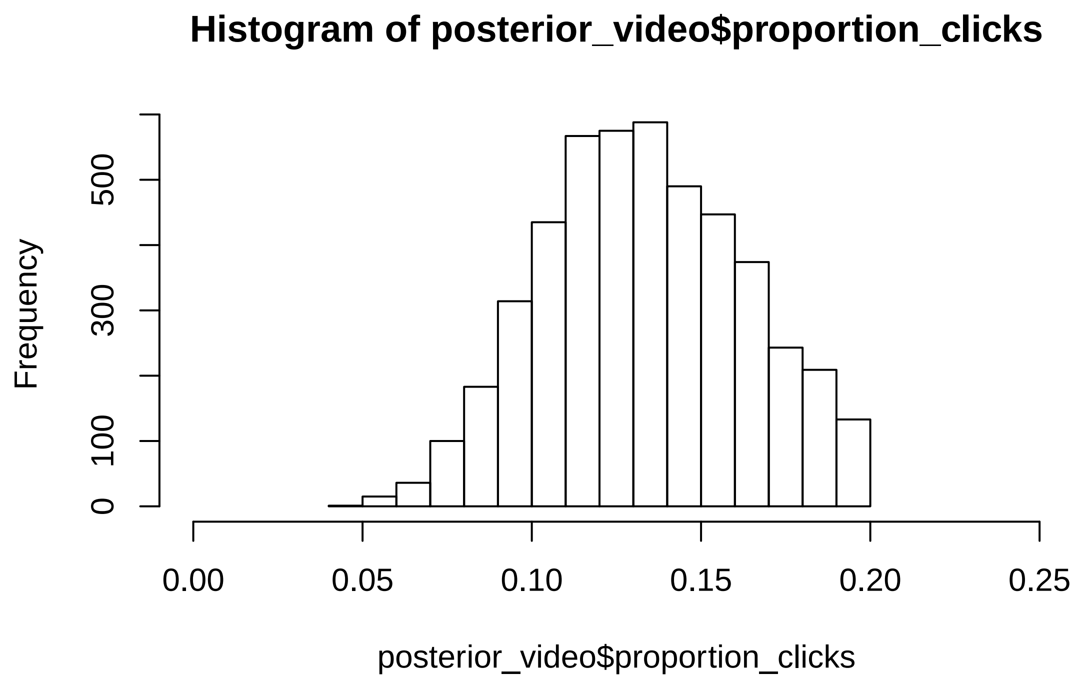
```r
n_draws <- 100000
n_ads_shown <- 100

# Change the prior on proportion_clicks
proportion_clicks <-
  rbeta(n_draws, shape1 = 5, shape2 = 95)
n_visitors <-
  rbinom(n_draws, size = n_ads_shown,
         prob = proportion_clicks)
prior <-
  data.frame(proportion_clicks, n_visitors)
posterior <-
  prior[prior$n_visitors == 13, ]

# This plots the prior and the posterior in the same plot
par(mfcol = c(2, 1))
hist(prior$proportion_clicks,
     xlim = c(0, 0.25))
hist(posterior$proportion_clicks,
     xlim = c(0, 0.25))
```



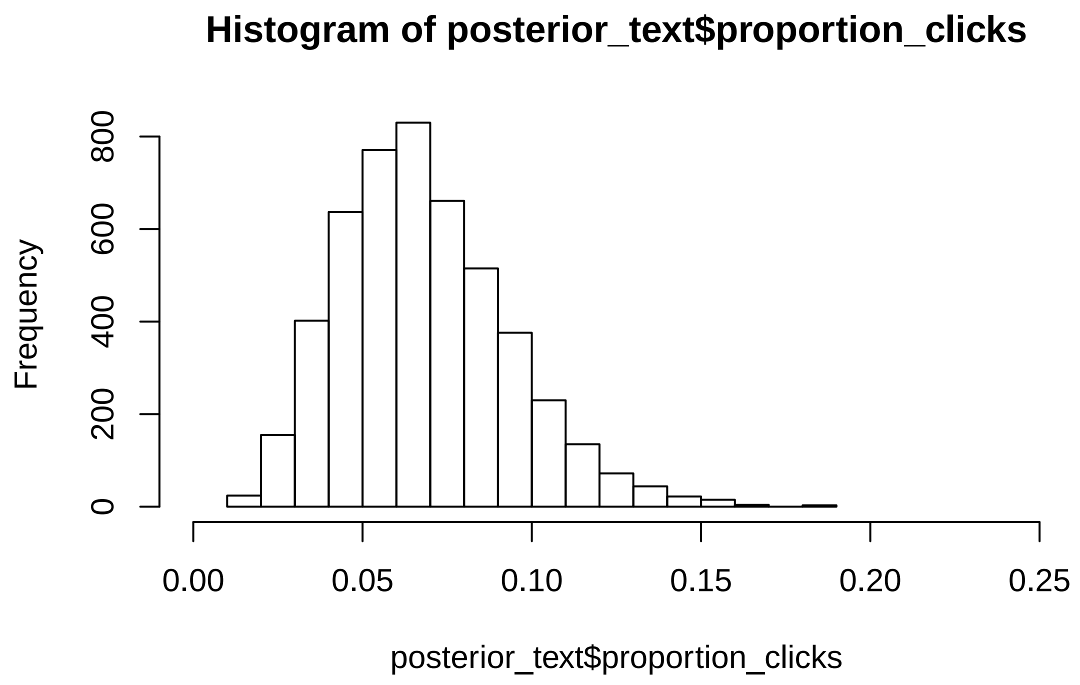


Visualial comparing of posteriors

```r
# Define parameters
n_draws <- 100000
n_ads_shown <- 100
proportion_clicks <- runif(n_draws, min = 0.0, max = 0.2)
n_visitors <- rbinom(n = n_draws, size = n_ads_shown,
                     prob = proportion_clicks)
prior <- data.frame(proportion_clicks, n_visitors)

# Create the posteriors for video and text ads
posterior_video <- prior[prior$n_visitors == 13, ]
posterior_text <- prior[prior$n_visitors == 6, ]

# Visualize the posteriors
hist(posterior_video$proportion_clicks, xlim = c(0, 0.25))
```



```r
hist(posterior_text$proportion_clicks, xlim = c(0, 0.25))
```



Calculating the posterior difference

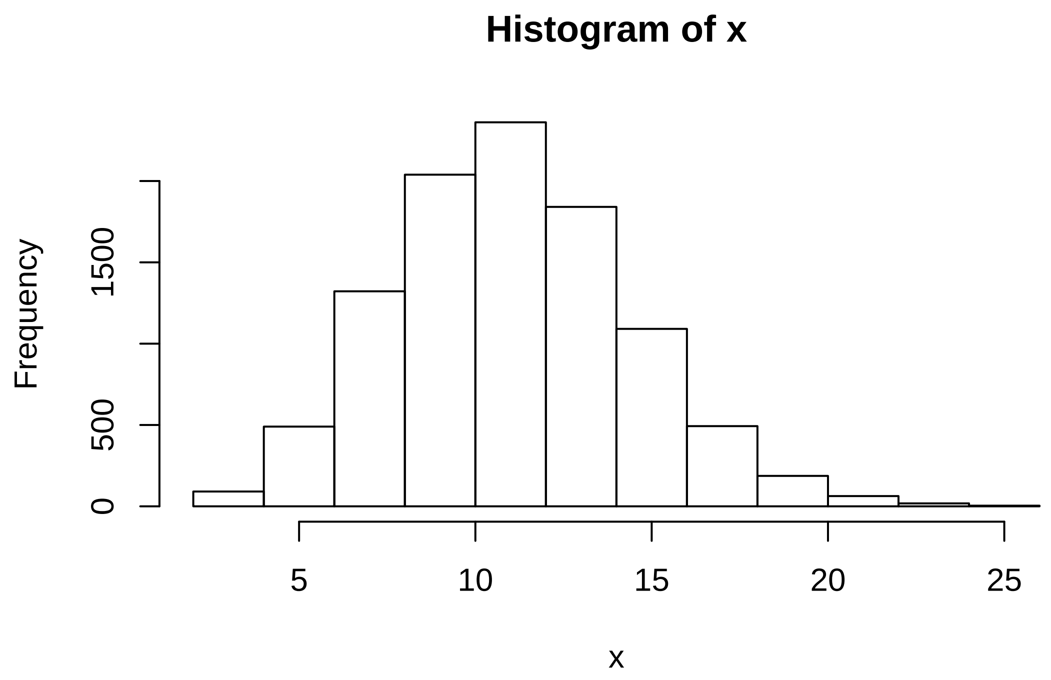
```
posterior <- data.frame(video_prop = posterior_video$proportion_clicks[1:4000],
                        text_prop = posterior_text$proportion_click[1:4000])

# Calculate the posterior difference: video_prop - text_prop
posterior$prop_diff <- posterior$video_prop - posterior$text_prop

# Visualize prop_diff
hist(posterior$prop_diff)

# Calculate the median of prop_diff
median(posterior$prop_diff)

# Calculate the proportion
sum(posterior$prop_diff>0)/length(posterior$prop_diff)
```

Decision analysis

```
# Add the column posterior$profit_diff
posterior$profit_diff <- posterior$video_profit - posterior$text_profit

# Visualize posterior$profit_diff
hist(posterior$profit_diff)

# Calculate a "best guess" for the difference in profits
median(posterior$profit_diff)

# Calculate the probability that text ads are better than video ads
sum(posterior$profit_diff < 0) / length(posterior$profit_diff)
```

The Poisson distribution

```
# Simulate from a Poisson distribution and visualize the result
x <- rpois(n = 10000, lambda = 11.5)
hist(x)
```



```r
# Calculate the probability of break-even
sum(x >= 15)/ length(x)
```

```
## [1] 0.1856
```

Bayesian inference with Bayes' theorem

Probability rules

```r
#Calculate the probability of drawing any of the four aces
prob_to_draw_ace <- 4/52
#Calculate the probability of picking four aces in a row
prob_to_draw_four_aces <- (4/52) * (3/51) * (2/50) * (1/49)
```

Calculating likelihoods

```r
n_ads_shown <- 100
proportion_clicks <- 0.1
prob_13_visitors <- dbinom(13,
    size = n_ads_shown, prob = proportion_clicks)
prob_13_visitors
```

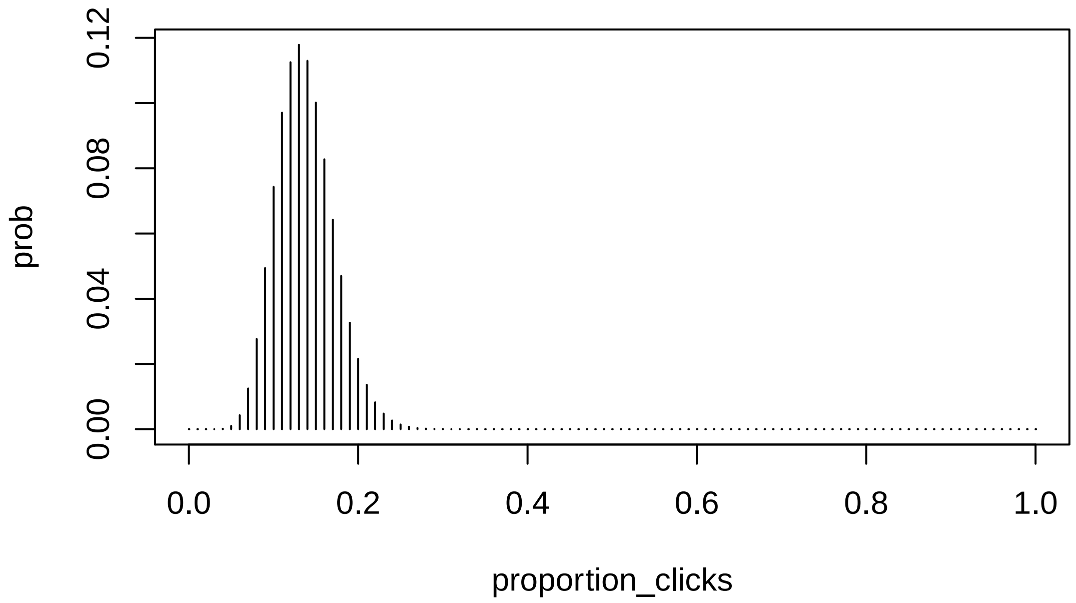
```
## [1] 0.07430209
```

Calculating probabilities with dbinom

```
# Change the code according to the instructions
n_ads_shown <- 100
proportion_clicks <- seq(0, 1, by = 0.01)
n_visitors <- 13
prob <- dbinom(n_visitors,
    size = n_ads_shown, prob = seq(0, 1, by = 0.01))
prob
```

```
##   [1]  0.000000e+00  2.965956e-11  1.004526e-07  8.009768e-06  1.368611e-04
##   [6]  1.001075e-03  4.265719e-03  1.247940e-02  2.764481e-02  4.939199e-02
##  [11]  7.430209e-02  9.703719e-02  1.125256e-01  1.178532e-01  1.129620e-01
##  [16]  1.001234e-01  8.274855e-02  6.419966e-02  4.701652e-02  3.265098e-02
##  [21]  2.158348e-02  1.362418e-02  8.234325e-03  4.775927e-03  2.663369e-03
##  [26]  1.430384e-03  7.408254e-04  3.704422e-04  1.790129e-04  8.366678e-05
##  [31]  3.784500e-05  1.657584e-05  7.032793e-06  2.891291e-06  1.151996e-06
##  [36]  4.448866e-07  1.665302e-07  6.041614e-08  2.124059e-08  7.234996e-09
##  [41]  2.386939e-09  7.624614e-10  2.357105e-10  7.048636e-11  2.037726e-11
##  [46]  5.691404e-12  1.534658e-12  3.991862e-13  1.000759e-13  2.415778e-14
##  [51]  5.609229e-15  1.251336e-15  2.678760e-16  5.495443e-17  1.078830e-17
##  [56]  2.023515e-18  3.620178e-19  6.166397e-20  9.980560e-21  1.531703e-21
##  [61]  2.223762e-22  3.046572e-23  3.927965e-24  4.752038e-25  5.377247e-26
##  [66]  5.671478e-27  5.554432e-28  5.030231e-29  4.193404e-30  3.201904e-31
##  [71]  2.227032e-32  1.402449e-33  7.942805e-35  4.015572e-36  1.797200e-37
##  [76]  7.054722e-39  2.403574e-40  7.024314e-42  1.737424e-43  3.582066e-45
##  [81]  6.048981e-47  8.199196e-49  8.713462e-51  7.062754e-53  4.226413e-55
##  [86]  1.795925e-57  5.170371e-60  9.521923e-63  1.044590e-65  6.239308e-69
##  [91]  1.807405e-72  2.180415e-76  8.911963e-81  9.240821e-86  1.591196e-91
##  [96]  2.358848e-98 1.001493e-106 1.546979e-117 8.461578e-133 6.239651e-159
## [101]  0.000000e+00
```

```
plot(proportion_clicks, prob, type = "h")
```



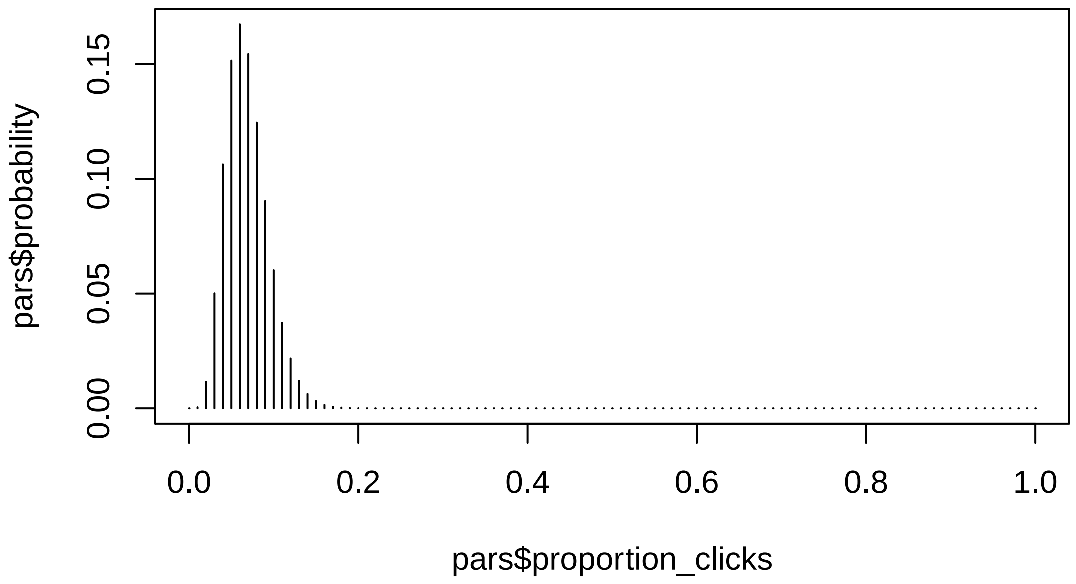
Calculating a joint distribution

```r
n_ads_shown <- 100
proportion_clicks <- seq(0, 1, by = 0.01)
n_visitors <- seq(0, 100, by = 1)
pars <- expand.grid(proportion_clicks = proportion_clicks,
                    n_visitors = n_visitors)
pars$prior <- dunif(pars$proportion_clicks, min = 0, max = 0.2)
pars$likelihood <- dbinom(pars$n_visitors,
    size = n_ads_shown, prob = pars$proportion_clicks)

# Add the column pars$probability and normalize it
pars$probability <- pars$likelihood * pars$prior
pars$probability <- pars$probability / sum(pars$probability)
```

Conditioning on the data

```r
n_ads_shown <- 100
proportion_clicks <- seq(0, 1, by = 0.01)
n_visitors <- seq(0, 100, by = 1)
pars <- expand.grid(proportion_clicks = proportion_clicks,
                    n_visitors = n_visitors)
pars$prior <- dunif(pars$proportion_clicks, min = 0, max = 0.2)
pars$likelihood <- dbinom(pars$n_visitors,
    size = n_ads_shown, prob = pars$proportion_clicks)
```

```r
pars$probability <- pars$likelihood * pars$prior
pars$probability <- pars$probability / sum(pars$probability)
# Condition on the data
pars <- pars[pars$n_visitors == 6, ]
# Normalize again
pars$probability <- pars$probability / sum(pars$probability)
# Plot the posterior pars$probability
plot(pars$proportion_clicks, pars$probability, type = "h")
```



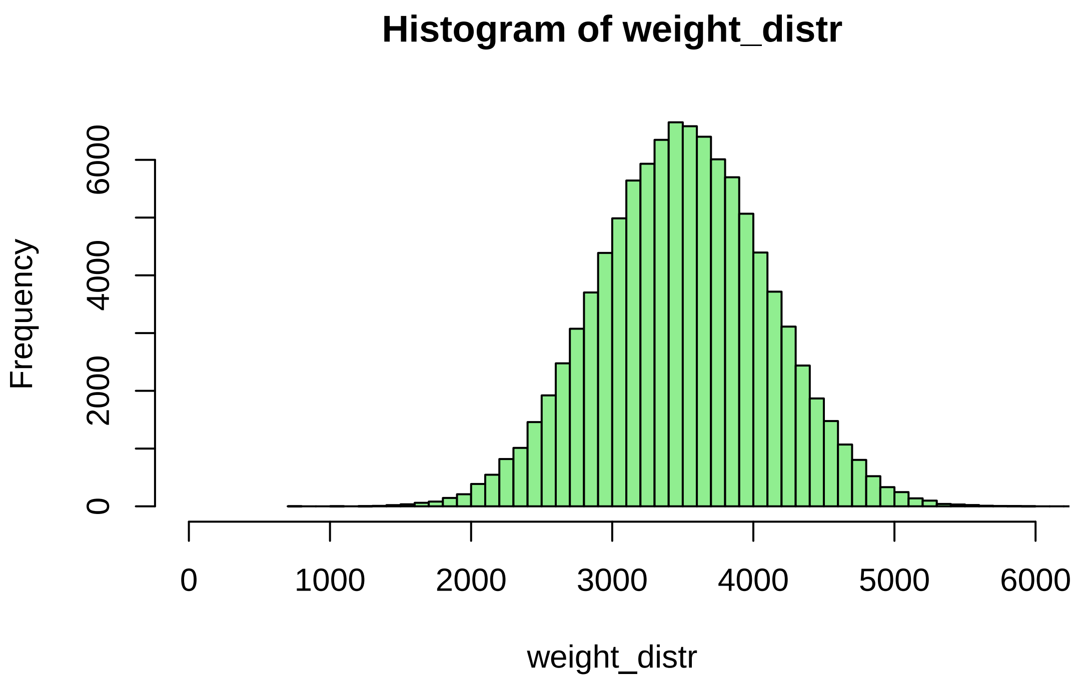
Bayes' theorem

```r
pars$probability <- pars$likelihood * pars$prior
pars$probability <- pars$probability / sum(pars$probability)
```

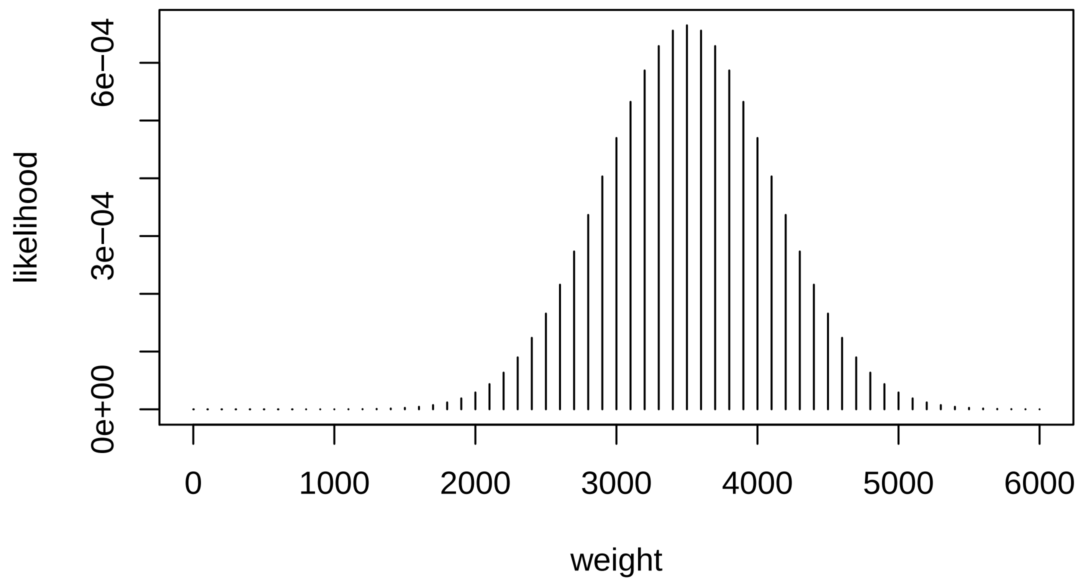
More parameters, more data, and more Bayes

Normal distribution

```r
# Assign mu and sigma
mu <- 3500
sigma <- 600

weight_distr <- rnorm(n = 100000, mean = mu, sd = sigma)
hist(weight_distr, 60, xlim = c(0, 6000), col = "lightgreen")
```



```
# Create weight
weight <- seq(0, 6000, by = 100)

# Calculate likelihood
likelihood <- dnorm(weight, mu, sigma)

# Plot the distribution of weight
plot(weight,likelihood,type="h")
```



A Bayesian model of Zombie IQ

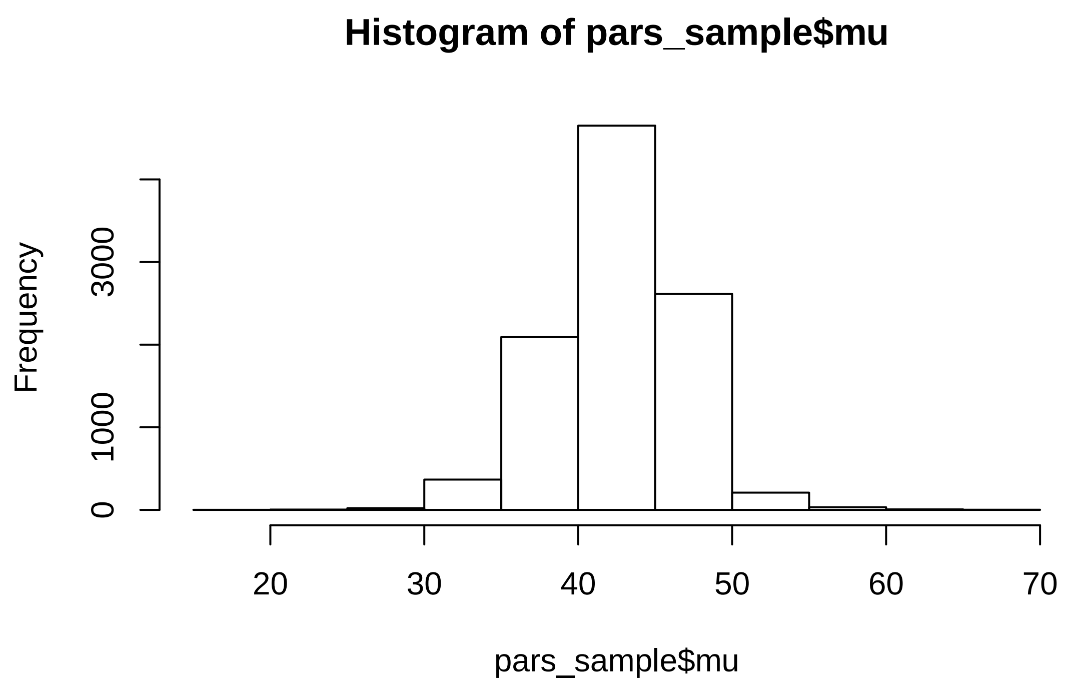
```r
# The IQ of a bunch of zombies
iq <- c(55, 44, 34, 18, 51, 40, 40, 49, 48, 46)
# Defining the parameter grid
pars <- expand.grid(mu = seq(0, 150, length.out = 100),
                    sigma = seq(0.1, 50, length.out = 100))
# Defining and calculating the prior density for each parameter combination
pars$mu_prior <- dnorm(pars$mu, mean = 100, sd = 100)
pars$sigma_prior <- dunif(pars$sigma, min = 0.1, max = 50)
pars$prior <- pars$mu_prior * pars$sigma_prior
# Calculating the likelihood for each parameter combination
for(i in 1:nrow(pars)) {
  likelihoods <- dnorm(iq, pars$mu[i], pars$sigma[i])
  pars$likelihood[i] <- prod(likelihoods)
}
# Calculate the probability of each parameter combination
pars$probability <- pars$likelihood * pars$prior
pars$probability <- pars$probability/sum(pars$probability)
```

Sampling from the zombie posterior

```r
sample_indices <- sample( 1:nrow(pars), size = 10000,
    replace = TRUE, prob = pars$probability)
head(sample_indices)
```

```
## [1] 2431 2630 3031 1331 2731 1629
```

```r
# Sample from pars to calculate some new measures
pars_sample <- pars[sample_indices, c("mu", "sigma")]

# Visualize pars_sample
hist(pars_sample$mu)
```

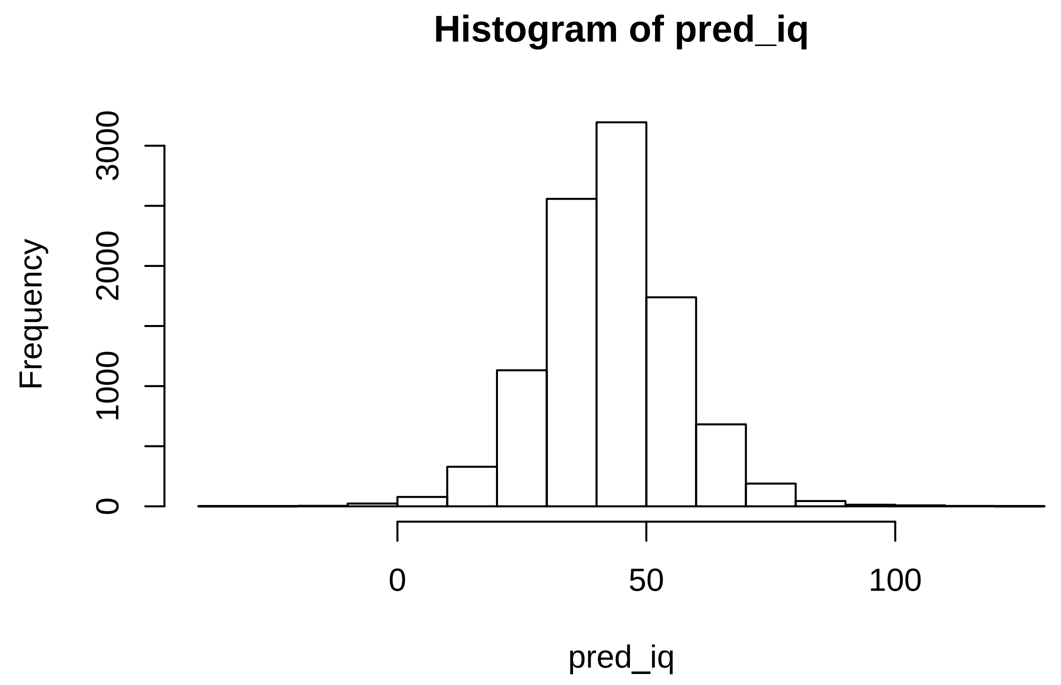


```r
# Calculate the 0.025, 0.5 and 0.975 quantiles of pars_sample$mu

quantile(pars_sample$mu, c(0.025, 0.5, .975))
```

```
##     2.5%      50%    97.5%
## 34.84848 42.42424 50.00000
```

what range of zombie IQs should we expect?

```r
pred_iq <- rnorm(10000, mean = pars_sample$mu,
                sd = pars_sample$sigma)
# Visualize pred_iq
hist(pred_iq)
```



```r
# Calculate the probability of a zombie being "smart" (+60 IQ)
pred_iq <- rnorm(10000, mean=pars_sample$mu, sd=pars_sample$sigma)

# the Pr that the next zombie you'll meet will have an IQ of >=60
sum(pred_iq >= 60)/length(pred_iq)
```

```
## [1] 0.0921
```

BEST package

```r
# The IQ of zombies on a regular diet and a brain based diet.
iq_brains <- c(44, 52, 42, 66, 53, 42, 55, 57, 56, 51)
iq_regular <- c(55, 44, 34, 18, 51, 40, 40, 49, 48, 46)

# Calculate the mean difference in IQ between the two groups
mean(iq_brains) - mean(iq_regular)
```

```
## [1] 9.3
```

```r
# Fit the BEST model to the data from both groups
library(BEST)
```

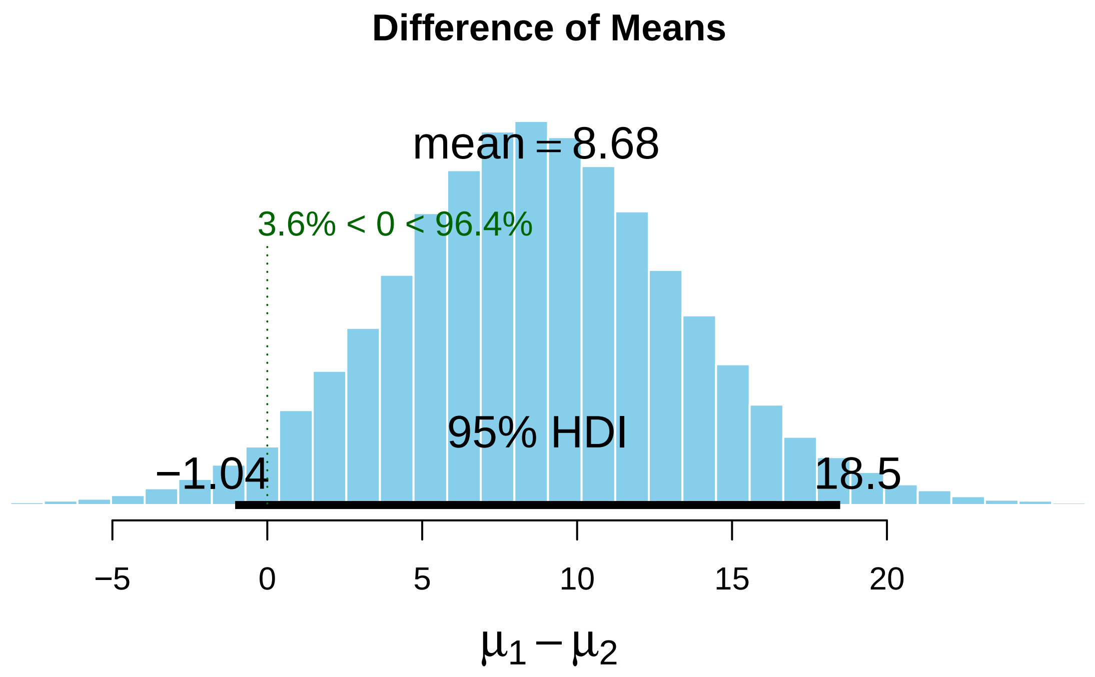
```
## Loading required package: HDInterval
```

```
best_posterior <- BESTmcmc(iq_brains, iq_regular)
```

```
## Waiting for parallel processing to complete...
```

```
## done.
```

```
# Plot the model result
plot(best_posterior)
```


Difference of Means
mean = 8.68
3.6% < 0 < 96.4%
95% HDI
$\mu_1 - \mu_2$