Bayesian Learning - Lab 01

Lakshidaa Saigiridharan (laksa656) and Maximilian Pfundstein (maxpf364)

2019-03-28

Contents

1 Bernoulli

1.1 Drawing from the Posterior

First we define the parameters as we need them later.

```
################################################################################
# Exercise 1
################################################################################

# Parameters
n = 20
s = 14
f = n - s

# Prior
alpha_z = 2
beta_z = 2

# Posterior
alpha_post = alpha_z + s
beta_post = beta_z + f
```

The posterior is given as $\text{Beta}(\alpha_n, \beta_n)$ where $\alpha_n = \alpha_0 + s$ and $\beta_n = \beta_0 + f$. Therefore the theoretical mean is given by:

$$\text{E}[X] = \frac{\alpha_n}{\alpha_n + \beta_n}$$

And the standard deviation by:

$$\text{var}[X] = \sqrt{\frac{\alpha_n \beta_n}{(\alpha_n + \beta_n)^2(\alpha_n + \beta_n + 1)}}$$

So let's calculate this.

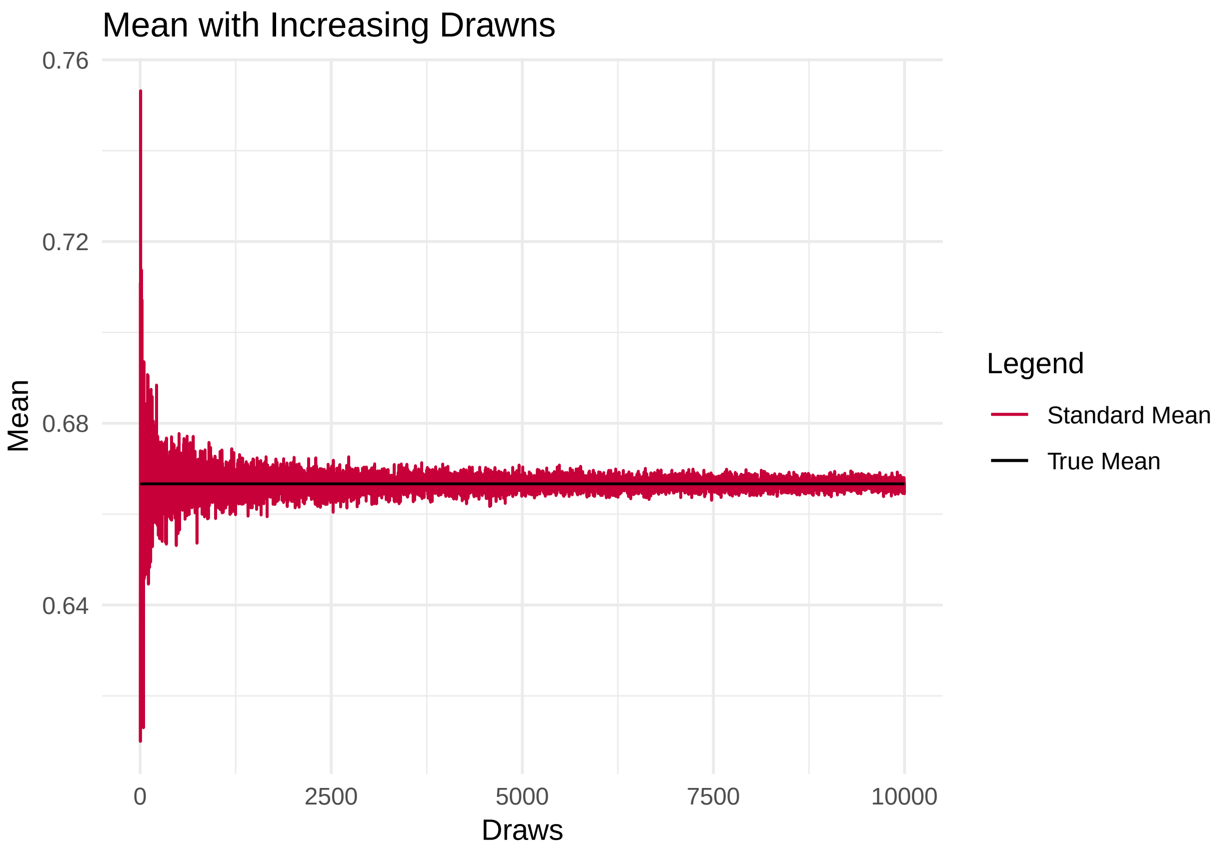
```
mean_posterior = alpha_post / (alpha_post + beta_post)
sd_posterior = sqrt((alpha_post * beta_post) /
                  ((alpha_post + beta_post)^2 * (alpha_post + beta_post + 1)))
```

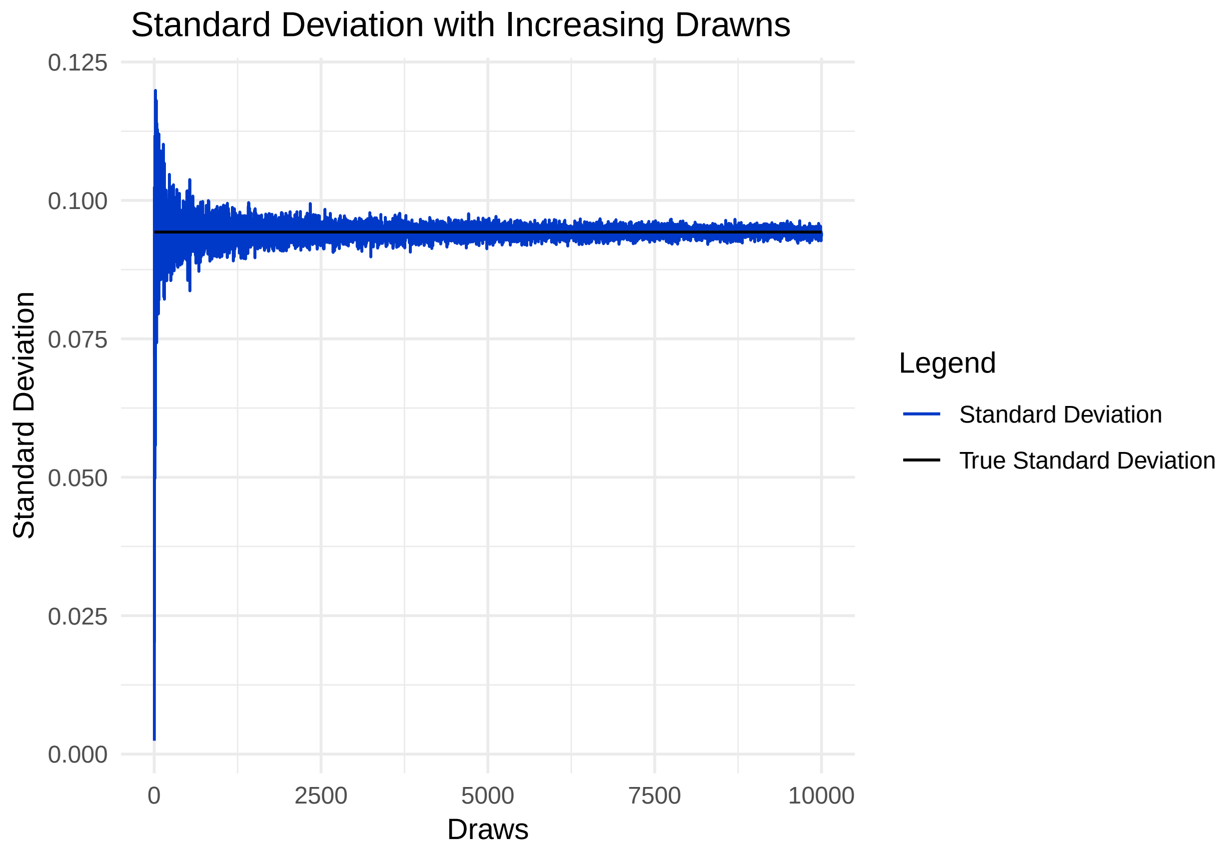
Therefore the mean of the prior is given by 0.6666667 and the standard deviation by 0.0942809.

Now we will create a function that calculates the mean and standard deviation for a given number of trials to plot it later on.

```
get_stats = function(n, alpha, beta) {
  samples = rbeta(n, alpha, beta)
  return(c(count = n, sample_mean = mean(samples), sample_sd = sd(samples)))
}

df = data.frame(t(sapply(2:10000, get_stats, alpha_post, beta_post)))
```





1.2 $Pr(\theta < 0.4|y)$

The true probability is given by `pbeta(0.4, alpha_post, beta_post)` which is 0.0039727.

We will simulate by taking samples and counting how many of them are < 0.4.

```
mean(rbeta(100000, alpha_post, beta_post) < 0.4)
```

```
## [1] 0.00412
```

As we can see both values are quite close to each other.

1.3 Log-Odds

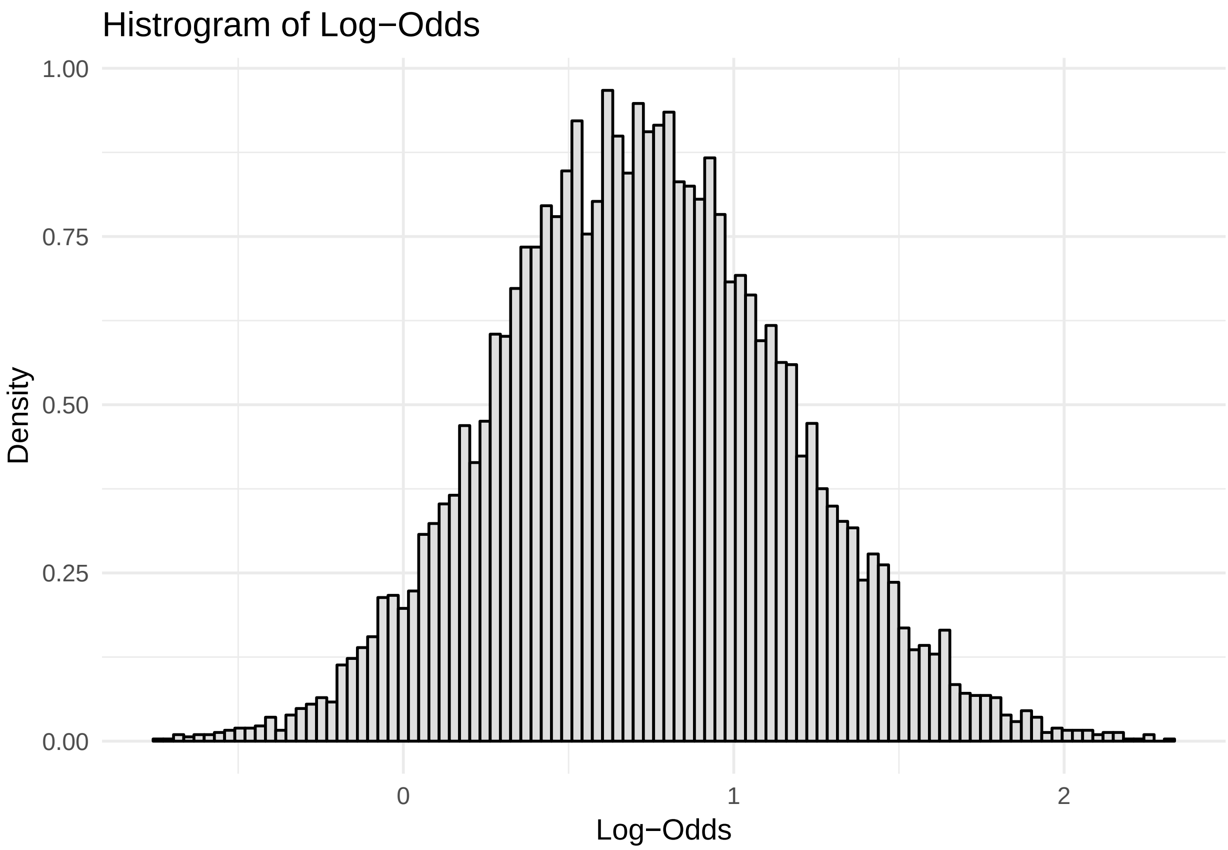
The log-odds are given by

$$\Phi = \log\left(\frac{\theta}{1-\theta}\right)$$

where θ are samples drawn from the posterior. We can therefore easily calculate the value by:

```
draws = 10000

samples = rbeta(draws, alpha_post, beta_post)
phi = data.frame(log(samples / (1 - samples)))
colnames(phi) = "phi"
```



2 Log-Normal Distribution and the Gini Coefficient

3 Bayesian Inference

4 Source Code

```
knitr::opts_chunk$set(echo = TRUE)
library(ggplot2)

################################################################################
# Exercise 1
################################################################################

# Parameters
n = 20
s = 14
f = n - s

# Prior
alpha_z = 2
beta_z = 2

# Posterior
```

```r
alpha_post = alpha_z + s
beta_post = beta_z + f


mean_posterior = alpha_post / (alpha_post + beta_post)
sd_posterior = sqrt((alpha_post * beta_post) /
                  ((alpha_post + beta_post)^2 * (alpha_post + beta_post + 1)))


get_stats = function(n, alpha, beta) {
  samples = rbeta(n, alpha, beta)
  return(c(count = n, sample_mean = mean(samples), sample_sd = sd(samples)))
}

df = data.frame(t(sapply(2:10000, get_stats, alpha_post, beta_post)))


ggplot(df) +
  geom_line(aes(x = count, y = sample_mean, color = "Standard Mean")) +
  geom_line(aes(x = count, y = mean_posterior, color = "True Mean")) +
  labs(title = "Mean with Increasing Drawns", y = "Mean", x = "Draws") +
  scale_color_manual("Legend", values = c("#C70039", "#000000")) +
  theme_minimal()


ggplot(df) +
  geom_line(aes(x = count, y = sample_sd, colour = "Standard Deviation")) +
  geom_line(aes(x = count, y = sd_posterior, colour = "True Standard Deviation")) +
  labs(title = " Standard Deviation with Increasing Drawns", y = "Standard Deviation", x = "Draws") +
  scale_color_manual("Legend", values = c("#0039C7", "#000000")) +
  theme_minimal()


mean(rbeta(100000, alpha_post, beta_post) < 0.4)


draws = 10000

samples = rbeta(draws, alpha_post, beta_post)
phi = data.frame(log(samples / (1 - samples)))
colnames(phi) = "phi"


ggplot(phi) +
  geom_histogram(aes(x = phi, y=..density..), color = "black",
                 fill = "#dedede", bins = sqrt(draws)) +
  labs(title = "Histrogram of Log-Odds",
  y = "Density",
  x = "Log-Odds", color = "Legend") +
  theme_minimal()
```