Exercise 8 - Introduction to Bayesian Inference

12433732 - Stefan Merdian

2024-12-22

Task 1

1.1

We first build the Beta prior distribution for the German study.

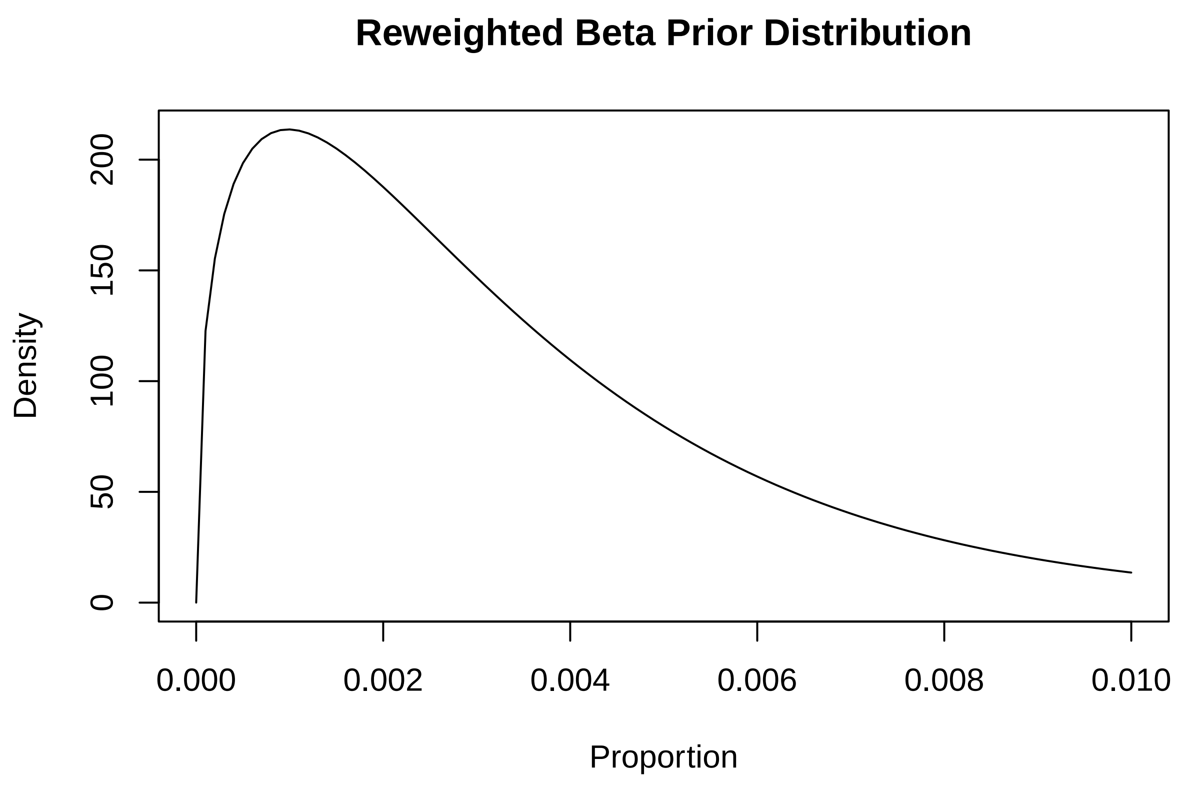
```r
germany_positive <- 4
germany_total <- 4068

# Reweighting the prior with a factor of 1/10
a_prior <- 1 + (germany_positive / 10)
b_prior <- 1 + ((germany_total - germany_positive) / 10)

cat("Reweighted Alpha:", a_prior, "\n")
```

```
## Reweighted Alpha: 1.4
```

```r
cat("Reweighted Beta:", b_prior, "\n")
```

```
## Reweighted Beta: 407.4
```

```r
curve(dbeta(x, a_prior, b_prior), from = 0, to = 0.01,
      ylab = "Density", xlab = "Proportion",
      main = "Reweighted Beta Prior Distribution")
```



1.2

Now we calculate a beta posterior for the Austrian study, with the German beta prior to update the update the Austrain study.

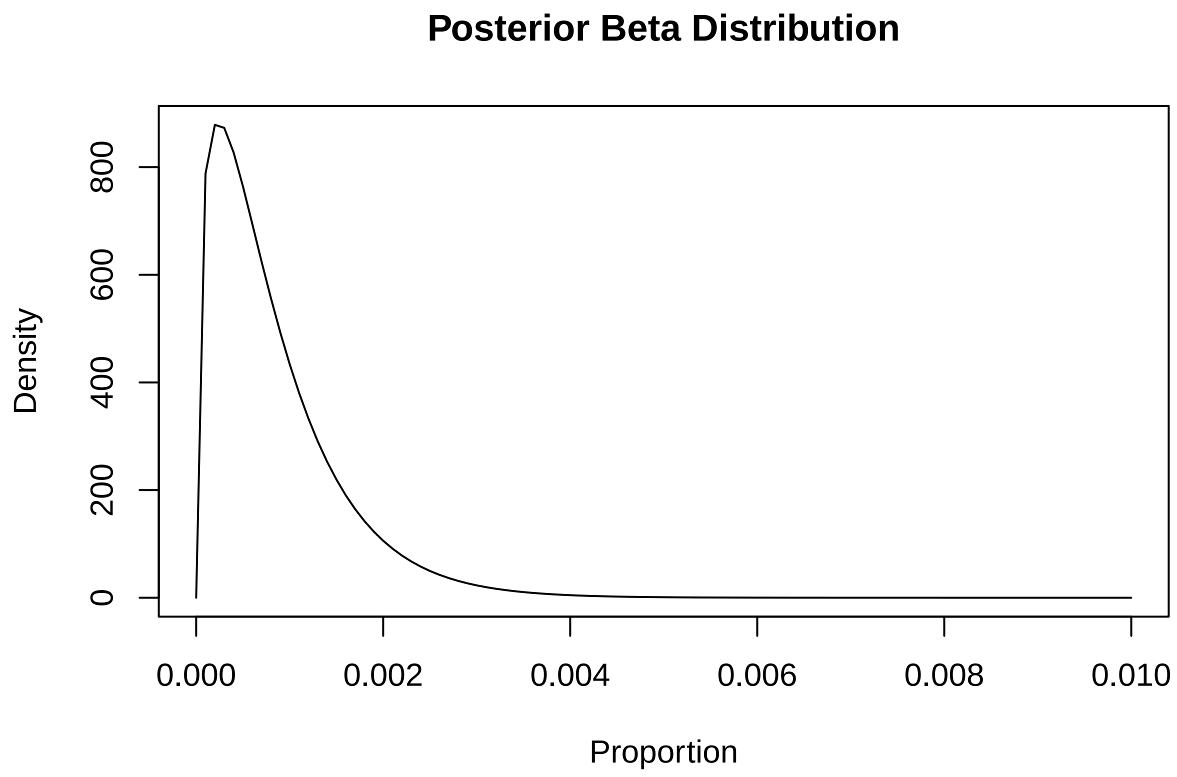
```
austria_positive <- 0
austria_total <- 1279

a_posterior <- a_prior + austria_positive
b_posterior <- b_prior + (austria_total - austria_positive)

cat("Posterior Alpha:", a_posterior, "\n")
```

```
## Posterior Alpha: 1.4
```

```
cat("Posterior Beta:", b_posterior, "\n")
```

```
## Posterior Beta: 1686.4
```

```
curve(dbeta(x, a_posterior, b_posterior), from = 0, to = 0.01,
      ylab = "Density", xlab = "Proportion",
      main = "Posterior Beta Distribution")
```



1.3 Plot the posterior density and 95% Highest posterior density

We calculate the mean and mode prevalence based on the posterior distribution. The mode shows the single value of the prevalence that's most consistent with the data.

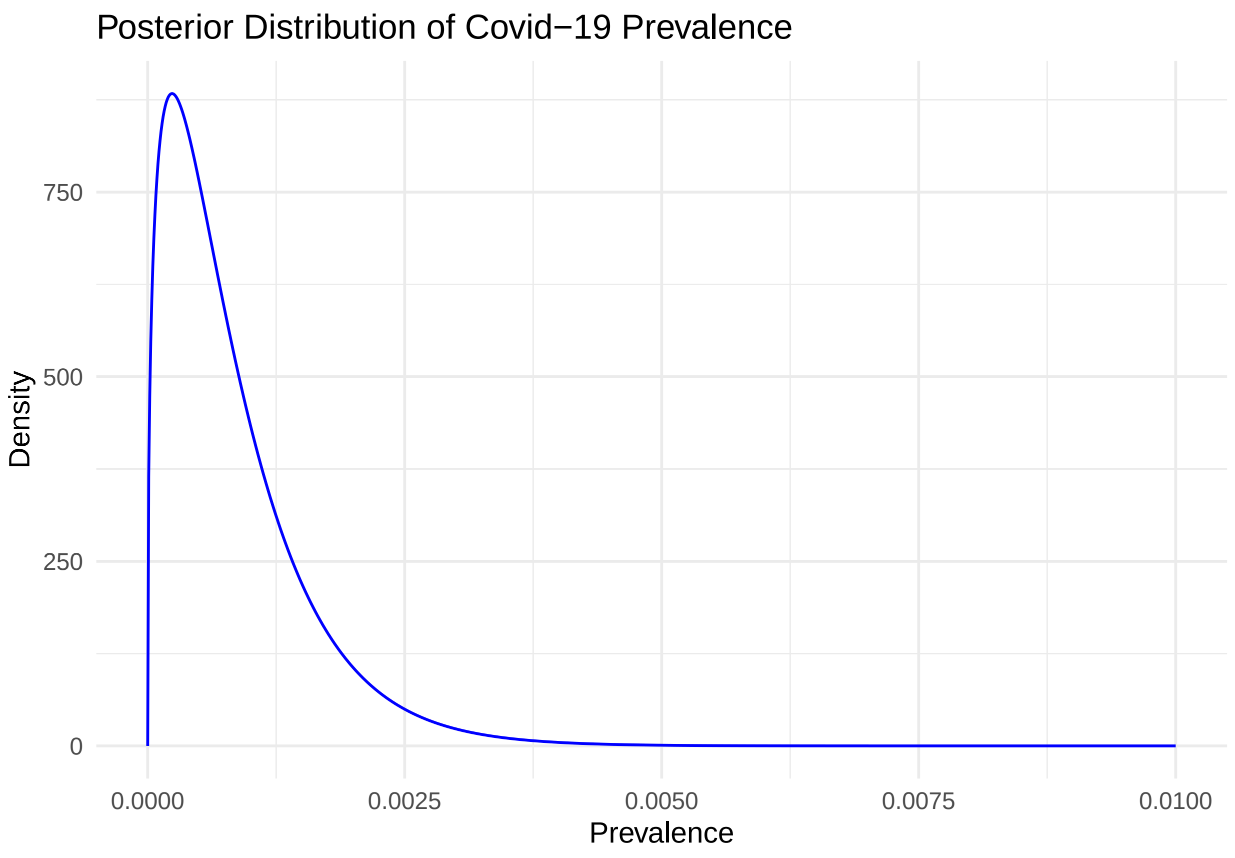
Also the 95% Highest Posterior Density Interval, it tells us the range of prevalence values that are most likely, covering 95% of the total probability.

We also plot the posterior density.

```r
library(ggplot2)
library(HDInterval)
```

```
## Warning: package 'HDInterval' was built under R version 4.4.2
```

```r
x <- seq(0, 0.01, length.out = 1000)
posterior_density <- dbeta(x, a_posterior, b_posterior)

posterior_plot <- data.frame(x, posterior_density)
ggplot(posterior_plot, aes(x, posterior_density)) +
  geom_line(color = "blue") +
  labs(
    title = "Posterior Distribution of Covid-19 Prevalence",
    x = "Prevalence",
    y = "Density"
  ) +
  theme_minimal()
```



```r
# Point estimators
mean_estimate <- a_posterior / (a_posterior + b_posterior)
mode_estimate <- (a_posterior - 1) / (a_posterior + b_posterior - 2)

# 95% HPDI
hpdi <- hdi(qbeta, credMass = 0.95, shape1 = a_posterior, shape2 = b_posterior)

cat("Mean estimate of prevalence:", mean_estimate, "\n")
```

```
## Mean estimate of prevalence: 0.0008294822
```

```r
cat("Mode estimate of prevalence:", mode_estimate, "\n")
```

```
## Mode estimate of prevalence: 0.0002372761
```

```r
cat("95% HPDI for prevalence:", hpdi, "\n")
```

```
## 95% HPDI for prevalence: 1.972584e-07 0.002210735
```

1.4

Frequentist methods typically rely solely on observed data, which, in this case, might lead to a prevalence estimate of exactly 0% because no positive cases were found in Austria. This would fail to account for the possibility of a small, nonzero prevalence. Bayesian methods allow Austria to balance the prior knowledge (from Germany) with the Austrian data to obtain a realistic estimate.

Simulation-based approaches can be computationally expensive and time-consuming, especially if you need high precision or repeat analyse. The Bayesian method with a Beta prior and a Binomial likelihood results in a conjugate posterior. This means Austria can directly compute the posterior distribution, avoiding the need for simulations. Also it offers an intuitive probabilistic interpretation (HPDI).

Task 2

2.1

```r
mu_beta <- 0
tau_beta2 <- 1e6
prior_beta <- function(beta) {
  dnorm(beta, mean = mu_beta, sd = sqrt(tau_beta2))
}

alpha <- 0.01
beta <- 0.01

prior_sigma2 <- function(sigma2) {
  if (sigma2 > 0) {
    dgamma(1 / sigma2, shape = alpha, rate = beta) / sigma2^2
  } else {
    0
  }
}

example_beta <- prior_beta(1)
example_sigma2 <- prior_sigma2(0.1)
cat("Prior density of   at 1:", example_beta, "\n")
```

```
## Prior density of   at 1: 0.0003989421
```

```r
cat("Prior density of  ² at 0.1:", example_sigma2, "\n")
```

```
## Prior density of  ² at 0.1: 0.08892867
```

To make the priors uninformative, the parameters of the priors should be set such that they have minimal influence on the posterior distribution. The goal of uninformative priors is to let the data dominate the inference rather than imposing strong assumptions.

- to make the prior uninformative, we choose a very large variance ($2 \to \infty$)

With these settings, the prior essentially says: "I have no strong belief about , so I am letting the data determine the estimate."

The prior for 2 is an Inverse-Gamma distribution:

- : Shape parameter (controls the weight given to prior beliefs).
- : Scale parameter (controls the expected size of 2).

To make the prior uninformative, we set:

- →0: Makes the prior flat.
- →0: Minimizes influence on the scale.

Compare different prior parameters:

For β: Normal Prior ($N(\mu_\beta, \tau_\beta^2)$)

Parameter Choice	Effect
Small τ_β^2	Strong prior belief about β. Constrains estimates to values close to μ_β.
Large τ_β^2	Weak or uninformative prior. Allows β to take on a wide range of values.

For σ^2: Inverse-Gamma Prior (IG(α, β))

Parameter Choice	Effect
Large α, β	Strong prior belief about σ^2. Constrains residual variance around specific values.
Small α, β	Weak or uninformative prior. Allows σ^2 to take on a wide range of positive values.

Key Differences

- **Informative Priors**: Constrain the parameters based on prior knowledge, reducing the influence of the data. Useful when you have strong prior information. In fact, lasso and ridge regression are equivalent to using specific informative priors on the coefficients in a Bayesian framework. Ridge imposes a Gaussian prior on the coefficients: N(0, −1). And Lasso imposes a Laplace prior on the coefficients: Laplace(0,b).
- **Uninformative Priors**: Let the data dominate, leading to more flexible and unbiased inference. Best when prior knowledge is minimal.

2.2

```r
set.seed(123)
n <- 100
x <- rnorm(n, mean = 0, sd = 1)
true_beta <- 2
```

```r
true_sigma2 <- 1
y <- rnorm(n, mean = true_beta * x, sd = sqrt(true_sigma2))

sigma2_known <- true_sigma2

beta_known <- true_beta

mu_beta <- 0
tau_beta2 <- 1e6
alpha_prior <- 0.01
beta_prior <- 0.01

posterior_beta <- function(sigma2) {
  tau_beta_star2 <- 1 / (sum(x^2) / sigma2 + 1 / tau_beta2)
  mu_beta_star <- tau_beta_star2 * (sum(x * y) / sigma2 + mu_beta / tau_beta2)
  list(mean = mu_beta_star, variance = tau_beta_star2)
}

posterior_beta_result <- posterior_beta(sigma2_known)
cat("Posterior for beta (assuming sigma2 is known):\n")
```

```
## Posterior for beta (assuming sigma2 is known):
```

```r
cat("Mean:", posterior_beta_result$mean, "\n")
```

```
## Mean: 1.936372
```

```r
cat("Variance:", posterior_beta_result$variance, "\n\n")
```

```
## Variance: 0.01200374
```

```r
posterior_sigma2 <- function(beta) {
  alpha_star <- alpha_prior + n / 2
  beta_star <- beta_prior + 0.5 * sum((y - beta * x)^2)
  list(shape = alpha_star, scale = beta_star)
}

posterior_sigma2_result <- posterior_sigma2(beta_known)
cat("Posterior for sigma2 (assuming beta is known):\n")
```

```
## Posterior for sigma2 (assuming beta is known):
```

```r
cat("Shape:", posterior_sigma2_result$shape, "\n")
```

```
## Shape: 50.01
```

```r
cat("Scale:", posterior_sigma2_result$scale, "\n")
```

```
## Scale: 46.87394
```

2.3

```r
posterior_sigma2_result <- posterior_sigma2(beta_known)
sigma2_mean <- posterior_sigma2_result$scale / (posterior_sigma2_result$shape - 1)
sigma2_hpdi <- c(
  1 / qgamma(0.975, shape = posterior_sigma2_result$shape, rate = posterior_sigma2_result$scale),
  1 / qgamma(0.025, shape = posterior_sigma2_result$shape, rate = posterior_sigma2_result$scale)
)

cat("Posterior for sigma2 (assuming beta is known):\n")
```

```
## Posterior for sigma2 (assuming beta is known):
```

```r
cat("Point Estimator (mean):", sigma2_mean, "\n")
```

```
## Point Estimator (mean): 0.9564158
```

```r
cat("95% HPDI:", sigma2_hpdi, "\n")
```

```
## 95% HPDI: 0.7234527 1.262782
```

2.4 Testwith dataset Auto

First we calculate the parameters for the frequentist model.

```r
library(ISLR)
data("Auto")

y <- Auto$mpg
x <- Auto$horsepower
n <- length(y)


frequentist_model <- lm(mpg ~ horsepower, data = Auto)
frequentist_beta <- coef(frequentist_model)
frequentist_beta_ci <- confint(frequentist_model)
frequentist_sigma2 <- summary(frequentist_model)$sigma^2

cat("Frequentist Results:\n")
```

```
## Frequentist Results:
```

```r
cat("Coefficient (beta):", frequentist_beta[2], "\n")
```

```
## Coefficient (beta): -0.1578447
```

```r
cat("95% Confidence Interval for beta:", frequentist_beta_ci[2, ], "\n")
```

```
## 95% Confidence Interval for beta: -0.170517 -0.1451725
```

```r
cat("Residual Variance (sigma^2):", frequentist_sigma2, "\n\n")
```

```
## Residual Variance (sigma^2): 24.06645
```

Now we calculate the parameters for the Bayesian model, if sigma2 is known.

```r
mu_beta <- 0
tau_beta2 <- 1e6

alpha_prior <- 0.01
beta_prior <- 0.01

sigma2_known <- frequentist_sigma2

posterior_beta <- function(sigma2) {
  tau_beta_star2 <- 1 / (sum(x^2) / sigma2 + 1 / tau_beta2)
  mu_beta_star <- tau_beta_star2 * (sum(x * y) / sigma2 + mu_beta / tau_beta2)
  list(mean = mu_beta_star, variance = tau_beta_star2)
}

posterior_beta_result <- posterior_beta(sigma2_known)
beta_mean <- posterior_beta_result$mean
beta_variance <- posterior_beta_result$variance
beta_hpdi <- c(
  beta_mean - 1.96 * sqrt(beta_variance),
  beta_mean + 1.96 * sqrt(beta_variance)
)

cat("Bayesian Results for beta (assuming sigma2 is known):\n")
```

```
## Bayesian Results for beta (assuming sigma2 is known):
```

```r
cat("Point Estimator (mean):", beta_mean, "\n")
```

```
## Point Estimator (mean): 0.1788398
```

```r
cat("95% HPDI:", beta_hpdi, "\n\n")
```

```
## 95% HPDI: 0.1744771 0.1832025
```

And also the same for the parameters, if beta is known.

```r
beta_known <- frequentist_beta[2]

posterior_sigma2 <- function(beta) {
  alpha_star <- alpha_prior + n / 2
  beta_star <- beta_prior + 0.5 * sum((y - beta * x)^2)
  list(shape = alpha_star, scale = beta_star)
}

posterior_sigma2_result <- posterior_sigma2(beta_known)
sigma2_mean <- posterior_sigma2_result$scale / (posterior_sigma2_result$shape - 1)
sigma2_hpdi <- c(
  1 / qgamma(0.975, shape = posterior_sigma2_result$shape, rate = posterior_sigma2_result$scale),
  1 / qgamma(0.025, shape = posterior_sigma2_result$shape, rate = posterior_sigma2_result$scale)
)

cat("Bayesian Results for sigma2 (assuming beta is known):\n")
```

```
## Bayesian Results for sigma2 (assuming beta is known):
```

```r
cat("Point Estimator (mean):", sigma2_mean, "\n")
```

```
## Point Estimator (mean): 1627.035
```

```r
cat("95% HPDI:", sigma2_hpdi, "\n\n")
```

```
## 95% HPDI: 1414.036 1871.587
```