Bayesian Learning Lab 1

Manu Jain [manja242] & Varun Gurupurandar [vargu125]

2025-04-19

QUESTION 1

Explanation of Problem

```
### Bayesian Inference using Beta-Binomial Model ###

### Concept and Working ###
# 1. We assume a Beta(alpha0, beta0) prior for theta (success probability).
# 2. Given observed data (s successes, f failures), we update our belief to get the posterior:
#       theta | y ~ Beta(alpha0 + s, beta0 + f)
# 3. We draw random samples from this posterior to estimate its properties.
# 4. We verify that the sample estimates converge to theoretical values.
# 5. We compute and compare posterior probability P(theta > 0.5 | y).
# 6. We analyze the posterior odds phi = theta / (1 - theta).

### Expected Plots and Their Interpretation ###
# 1. **Prior and Posterior Distributions**
#    - Prior shows initial belief before observing data.
#    - Posterior shows updated belief after observing successes and failures.
#
# 2. **Convergence Plots**
#    - Posterior Mean: Estimated mean from samples should converge to the true posterior mean.
#    - Posterior Std Dev: Estimated standard deviation should stabilize around the true value.
#
# 3. **Posterior Probability P(theta > 0.5 | y)**
#    - Monte Carlo estimate from samples should match the exact probability from the Beta distribution.
#
# 4. **Posterior Distribution of Odds phi = theta / (1 - theta)**
#    - Histogram of odds gives a discrete view of the odds distribution.
#    - Density plot provides a smoothed view of the distribution.




##EXPLANANTION OF THE PROBLEM TASK 1
#We have been given failure value we can get success value using the equation given in the slide .
# and even we are given alpha and beta values ,
#1.We need to use the above values and use Montecarlo(MC) and draw 10000 values
#and check weather posterior mean and SD have same  true values as and when the draw increases
#2.Compute Posterior probability using 2 different methods (a)pbeta(),(b)MC approximation .
#both methods will be having the same threshold values 0.5 .
```

```
#3.Since we have previosly have parameter theta ,now we need to  understand the odd of success instead
# odd is bacically the ration of success by failure
```

Part A)

```
### Variables ###
f = 35  # Number of failures
n = 78  # Total trials
a0 = 7  # Prior alpha
b0 = 7  # Prior beta
s = n - f  # Number of successes
#HELPER FUNCTIONS WHICH WILL BE USED REPETITIVELY
meanBeta <- function(a, b) {
  return(a / (a + b))
}

varBeta <- function(a, b) {
  return((a * b) / (((a + b)^2) * (a + b + 1)))
}

BetaPlot <- function(a, b) {
  xGrid <- seq(0.001, 0.999, by = 0.001)
  prior = dbeta(xGrid, a, b)
  maxDensity <- max(prior)
  plot(xGrid, prior, type = 'l', lwd = 3, col = "blue", xlim = c(0,1), ylim = c(0, maxDensity),
       xlab = "theta", ylab = 'Density', main = paste('Beta(', a, ',', b, ') density'))
}
#Draw 10,000 samples, show posterior mean and SD converge correctly.
# & Visualize convergence of Beta posterior estimates to theoretical true values.

# Prior and Posterior Plots
BetaPlot(a0, b0)  # Prior density
BetaPlot(a0 + s, b0 + f)  # Posterior density

# Posterior Mean and Variance
meanPost = meanBeta(a0 + s, b0 + f)
varPost = varBeta(a0 + s, b0 + f)
stdPost = sqrt(varPost)

# Sampling from Posterior
Ndraws = 10000
samples = rbeta(Ndraws, shape1 = a0 + s, shape2 = b0 + f)

# Convergence of Mean and Std Dev
mean_vals = rep(0, Ndraws)
std_vals = rep(0, Ndraws)

for (i in 1:Ndraws) {
  mean_vals[i] = mean(samples[1:i])
  std_vals[i] = sd(samples[1:i])
}
```

```r
plot(mean_vals, type = "l", col = "red", main = "Convergence of Posterior Mean",
     xlab = "Number of Draws", ylab = "Mean Estimate")
abline(h = meanPost, col = "blue", lty = 2)

plot(std_vals, type = "l", col = "red", main = "Convergence of Posterior Std Dev",
     xlab = "Number of Draws", ylab = "Std Dev Estimate", ylim = c(0, 0.1))
abline(h = stdPost, col = "blue", lty = 2)
```

Part B)

```r
# Part (b) - Compute Probability P(theta > 0.5 | y)
threshold = 0.5
true_probability = 1 - pbeta(threshold, shape1 = a0 + s, shape2 = b0 + f)
simulated_probability = mean(samples > threshold)

print(paste("True Probability: ", round(true_probability, 4)))
print(paste("Simulated Probability: ", round(simulated_probability, 4)))
```

Part C)

```r
# Part (c) - Posterior Distribution of Odds phi
phi = samples / (1 - samples)
hist(phi, breaks = 50, main = "Histogram of Posterior Odds", xlab = "phi", col = "lightblue")
plot(density(phi), main = "Density Plot of Posterior Odds", col = "red", lwd = 2)
```

QUESTION 2

Explanation of Problem

```r
# Step 1: Define Variables
# income      -> Monthly income data (in 1000 SEK)
# my ( )      -> Given log-mean income (3.65)
# Ndraw       -> Number of posterior samples (10,000)
# ============================
# Step 2: Compute   (Tao)
# Tao ( ) measures the variance of the log-transformed data.
# Formula:   = (1/n) * Σ (log(y) -  )^2
# This helps us estimate the posterior distribution of variance ( ²).
# ============================
# Step 3: Draw Samples from Chi-Squared Distribution
# The posterior for  ² follows an inverse-chi-squared distribution.
# We generate 10,000 random samples from a chi-squared distribution with 'n' degrees of freedom.
# ============================
# Step 4: Compute Posterior Samples of  ²: Formula:  ² = (n *  ) /  ²
# This gives us the posterior distribution of variance.
# ============================
# Step 5: Plot Posterior Distribution of  ²: This plot visualizes the posterior uncertainty in  ².
# A wider distribution indicates higher uncertainty.
# ==============================
# Step 6: Compute Posterior Distribution of Gini Coefficient
# Gini coefficient measures income inequality.
# Formula: G = 2 * Φ( / sqrt(2)) - 1
# Φ(z) is the standard normal CDF.
# ==============================
# Step 7: Plot Gini Distribution:This plot shows the probability density of the Gini coefficient.
# A lower Gini value (~0.03) suggests an equal income distribution.
# ==============================
# Step 8: Compute 95% Equal-Tail Credible Interval:The 95% equal-tail credible interval retains the mid
# It removes the lower 2.5% and upper 2.5% of probability mass.
# ==============================
# Step 9: Compute 95% HPDI (Highest Posterior Density Interval):The HPDI retains the shortest interval
# This is different from the equal-tail interval because it excludes low-density regions.
# ==============================
# Step 10: Compare Equal-Tail and HPDI Intervals
# Equal-Tail Interval:
# - Retains the middle 95% of posterior samples.
# - Includes low-probability regions.
## HPDI:
# - Keeps the most probable 95% region.
# - More precise but computationally complex.

##We have been given income values and we apply logarthmic to those values .
#They have given probability density function for logarthmic distribution,
#mean is known but varience is unknown
#1:We need to calculate posterior distribution .the posterior has different distribution
#called scaled inverse chi square.
#2:once we have 10000 possible sigma square values we will take square root of each sampled sigma squar
#to get sigma and then plug into gini formula mentioned .That will give 10000 gini values ,
#the main observation is to see distribution of gini which measures income inequality
#3:next we sort them from small to large and then trim 2.5 % from highest and lowest part.
#we get the rest 95% credibility interval.
#4:showing where the densest 95% is present .
#finding the chunk of distributuion with most probable values
```

Part A)

```r
# Load required libraries
library(dplyr)
library(bayestestR)  # Needed for HPDI calculation

# Given income data
income <- c(22, 33, 31, 49, 65, 78, 17, 24)

# (a) Compute tao^2 and draw 10,000 samples from the posterior of sigma^2
mu <- 3.65          # Known mean mu
Ndraw <- 10000      # Number of posterior samples
n <- length(income) # Sample size


# Function to compute tao² = (1/n) * Σ (log(y_i) - mu)^2
```

```r
taoFunc <- function(y) {
  return(sum((log(y) - mu)^2) / length(y))
}
tao <- taoFunc(income)

# Draw from chi-squared distribution and compute sigma² posterior samples
set.seed(12345)
chi_vals <- rchisq(Ndraw, n)
sigmaSquared <- (n * tao) / chi_vals  # Posterior samples of sigma²

# Plot posterior distribution of sigma²
plot(density(sigmaSquared), main = expression(paste("Posterior Distribution of ", sigma^2)))
```

Part B)

```r
sigma <- sqrt(sigmaSquared)  # Get standard deviation from variance
G <- 2 * pnorm(sigma / sqrt(2)) - 1  # Compute Gini for each draw

# Plot the posterior distribution of Gini coefficient
plot(density(G), main = "Posterior Distribution of Gini Coefficient")
```

Part C)

```r
# (c) Compute 95% Equal-Tail Credible Interval for Gini
sorted_G <- sort(G)
lower_ET <- sorted_G[round(Ndraw * 0.025)]
upper_ET <- sorted_G[round(Ndraw * 0.975)]
credible_interval_ET <- c(lower_ET, upper_ET)

# Plot the credible interval on Gini density
plot(density(G), main = "95% Equal-Tail Credible Interval for G")
abline(v = lower_ET, col = "red", lty = 2, lwd = 2)
abline(v = upper_ET, col = "blue", lty = 2, lwd = 2)

# Print Equal-Tail Credible Interval
cat("95% Equal-Tail Credible Interval for Gini:\n")
print(credible_interval_ET)
```

Part D)

```r
# (d) Compute 95% Highest Posterior Density Interval (HPDI) using bayestestR
hpdi_interval <- hdi(G, ci = 0.95)

# Plot HPDI on top of Gini density
plot(density(G), main = "95% HPDI for Gini Coefficient")
abline(v = hpdi_interval$CI_low, col = "green", lty = 2, lwd = 2)
abline(v = hpdi_interval$CI_high, col = "purple", lty = 2, lwd = 2)
```

```r
# Print HPDI Interval
cat("95% Highest Posterior Density Interval (HPDI) for Gini:\n")
print(hpdi_interval)

# Comparison of intervals
cat("\nComparison:\n")
cat("Equal-Tail Interval: ", round(credible_interval_ET[1], 4), "-", round(credible_interval_ET[2], 4),
cat("HPDI Interval      : ", round(hpdi_interval$CI_low, 4), "-", round(hpdi_interval$CI_high, 4), "\n"
```

QUESTION 3

Part A)

```r
############ Question 3 ##############

### Part A
# Data: Number of goals per match
y <- c(0, 2, 5, 5, 7, 1, 4)

# Parameters
sigma <- 5  # scale of half-normal prior
lambda_grid <- seq(0, 15, length.out = 1000)  # Grid of lambda values

# Prior: PDF of Half-normal distribution
prior <- function(lambda, sigma) {
  sqrt(2) / (sigma * sqrt(pi)) * exp(-lambda^2 / (2 * sigma^2))
}

# Likelihood: Product of Poisson
likelihood <- function(lambda, y) {
  sapply(lambda, function(l) prod(dpois(y, lambda = l)))
}

# Derivation of unnormalized posterior over the grid
unnormalized_posterior <- likelihood(lambda_grid, y) * prior(lambda_grid, sigma)

# Normalize the posterior
posterior <- unnormalized_posterior / sum(unnormalized_posterior * diff(lambda_grid)[1])
# Here, In normalization we multiply by 'diff(lambda_grid)[1]' because it is the width
# between grid points. Instead of integrating, we approximate the integral as a sum over
# small intervals.

# Plot posterior
plot(lambda_grid, posterior, type = "l", lwd = 2, col = "lightblue",
     xlab = expression(lambda), ylab = "Posterior density",
     main = expression(paste("Posterior Distribution of ", lambda)))
```

Part B)

```r
### Part B
# Find posterior mode (MAP estimate)
posterior_mode <- lambda_grid[which.max(posterior)]
cat("Posterior mode of lambda:", posterior_mode, "\n")

# Commands to show posterior mode in plot-
#abline(v = lambda_map, col = "red", lty = 2)
#text(lambda_map, max(posterior), labels = paste("MAP =", round(lambda_map, 2)),
#     pos = 4, col = "red")
```