MS2505: Bayesian Statistics

Course Project

December 17, 2024

Name	Samuel Jonsson
E-Mail	sajs19@student.bth.se
Person Nr.	19990415-5596
Program	DVAMI19h

1 Setup

- Describe the data and the analysis problem.
- Choose and describe the modeling approach (e.g., non-hierarchical or hierarchical model).
- Justify your prior choice.
- Perform posterior predictive checks.

1.1 Analysis problem

1.2 Data Selection

Describe the data and the analysis problem.

The dataset selected is a datasets containing a list of emails, as well as a label marking each email as "spam" or "ham" (spam or not spam). The first 10 rows of the dataset looks as follows:

Table 1
`mail_data.csv` dataset first 10 rows

Category	Message
ham	"Go until jurong point, crazy.. Available only in bugis n great world la e buffet... Cine there got amore wat..."
ham	Ok lar... Joking wif u oni...
spam	Free entry in 2 a wkly comp to win FA Cup final tkts 21st May 2005. Text FA to 87121 to receive entry question(std txt rate)T&C’s apply 08452810075over18’s
ham	U dun say so early hor... U c already then say...
ham	"Nah I don’t think he goes to usf, he lives around here though"
spam	"FreeMsg Hey there darling it’s been 3 week’s now and no word back! I’d like some fun you up for it still? Tb ok! XxX std chgs to send, £1.50 to rcv"
ham	Even my brother is not like to speak with me. They treat me like aids patent.
ham	As per your request ’Melle Melle (Oru Minnaminunginte Nurungu Vettam)’ has been set as your callertune for all Callers. Press *9 to copy your friends Callertune
...	...

Then, using a Python script, the labels were converted to 1 if it was "spam" and 0 if it was "ham", for easier analysis.

1.3 Model

- Choose and describe the modeling approach (e.g., non-hierarchical or hierarchical model).
- Justify your prior choice.

The model chosen was a binomial likelihood model with a beta prior. As the goal is to analyse the probability of an email being spam, the fallout will be binary (either it is spam or it is not). Hence, a binomial likelihood, where I want to find the parameter θ in a dataset of fixed size with a set number of "successes" and "fails" (spam and ham), is appropriate.

Additionally, as I do not have any prior knowledge in regards to this distribution, a non-informative prior is the most suited option, and as $Beta(1,1)$ is a common prior used with binomial likelihood functions, I chose it for this problem.

1.4 Prior checks

,

Perform posterior predictive checks.

2 Results

Include diagnostics to assess model convergence and adequacy.

3 Discussion

Discuss results, problems encountered, and possible improvements.

A R Code

Listing 1
Project R code

```r
# =====================================================
# Bayesian Analysis for Email Spam Classification
# =====================================================

# Load Required Libraries
required_packages <- c("ggplot2", "dplyr", "MCMCpack", "coda", "tidyr")
installed_packages <- rownames(installed.packages())
for (pkg in required_packages) {
  if (!pkg %in% installed_packages) install.packages(pkg)
  library(pkg, character.only = TRUE)
}

# Set a global random seed for reproducibility
set.seed(123)

# Output Logs
if (!dir.exists("logs")) dir.create("logs")
results_log_file <- "logs/combined_results.log"

# =====================================================
# Load Data
# =====================================================

data_path <- "data/mail_data_bin.csv"
```

```
if (!file.exists(data_path)) stop("Data file not found!")
mail_data <- read.csv(data_path)

# Ensure the response variable is binary and properly coded
if (!all(mail_data$Category %in% c(0, 1))) {
  stop("The response variable ’Category’ must be binary (0 or 1).")
}

# Metadata
spam_count <- sum(mail_data$Category == 1)
ham_count <- sum(mail_data$Category == 0)
total_emails <- nrow(mail_data)


sink(results_log_file)
cat("--- Data Metadata ---\n")
cat("Spam Count:", spam_count, "\n")
cat("Ham Count:", ham_count, "\n")
cat("Total Emails:", total_emails, "\n")
sink()

# =====================================================
# Beta Posterior Analysis
# =====================================================

# Prior parameters
prior_alpha <- 1
prior_beta <- 1

# Posterior parameters
posterior_alpha <- prior_alpha + spam_count
posterior_beta <- prior_beta + ham_count

# Monte Carlo Sampling
n_samples <- 10000
beta_samples <- rbeta(n_samples, posterior_alpha, posterior_beta)

# Summary statistics
beta_mean <- mean(beta_samples)
beta_sd <- sd(beta_samples)
beta_ci <- quantile(beta_samples, c(0.025, 0.975))

sink(results_log_file, append = TRUE)
cat("\n--- Beta Posterior Analysis ---\n")
cat("Posterior Mean:", beta_mean, "\n")
cat("Posterior SD:", beta_sd, "\n")
cat("95% Credible Interval:", beta_ci, "\n")
sink()

# =====================================================
# MCMC Sampling
# ==================================================

# Define log-posterior function
log_posterior <- function(params) {
  theta <- params[1]
  log_prior <- dbeta(theta, 1, 1, log = TRUE)
  log_likelihood <- sum(dbinom(mail_data$Category,
                             size = 1,
                             prob = theta,
                             log = TRUE))
  return(log_prior + log_likelihood)
}

# Run MCMC sampling
mcmc_results <- MCMCmetrop1R(
  fun = log_posterior,
  theta.init = 0.5,
  burnin = 1000,
  mcmc = n_samples,
  thin = 1,
  verbose = 0
)

# Extract posterior samples
mcmc_samples <- as.vector(mcmc_results)
mcmc_mean <- mean(mcmc_samples)
mcmc_sd <- sd(mcmc_samples)
mcmc_ci <- quantile(mcmc_samples, c(0.025, 0.975))

sink(results_log_file, append = TRUE)
cat("\n--- MCMC Sampling Analysis ---\n")
cat("Posterior Mean:", mcmc_mean, "\n")
cat("Posterior SD:", mcmc_sd, "\n")
cat("95% Credible Interval:", mcmc_ci, "\n")
sink()

# ==================================================
# Diagnostics and Visualization
# ==================================================

if (!dir.exists("figures")) dir.create("figures")

# 1. Beta Density Plot
pdf("figures/beta_posterior_density_plot.pdf")
ggplot(data = data.frame(samples = beta_samples), aes(x = samples)) +
  geom_density(fill = "lightblue", alpha = 0.7) +
  geom_vline(xintercept = beta_mean, color = "red", linetype = "
     dashed") +
  geom_vline(xintercept = beta_ci, color = "blue", linetype = "dotted
     ") +
```

```r
  labs(title = "Beta Posterior Density", x = "Probability", y = "Density") +
  theme_minimal()
dev.off()

# 2. MCMC Density Plot
pdf("figures/mcmc_posterior_density_plot.pdf")
ggplot(data = data.frame(samples = mcmc_samples), aes(x = samples)) +
  geom_density(fill = "lightblue", alpha = 0.7) +
  geom_vline(xintercept = mcmc_mean, color = "red", linetype = "dashed") +
  geom_vline(xintercept = mcmc_ci, color = "blue", linetype = "dotted") +
  labs(title = "MCMC Posterior Density", x = "Probability", y = "Density") +
  theme_minimal()
dev.off()

# 3. Beta Posterior Histogram
pdf("figures/beta_posterior_histogram.pdf")
hist(beta_samples,
  breaks = 30, col = "lightgreen", border = "black",
  xlab = "Probability", main = "Beta Posterior Histogram"
)
dev.off()

# 4. MCMC Posterior Histogram
pdf("figures/mcmc_posterior_histogram.pdf")
hist(mcmc_samples,
  breaks = 30, col = "lightblue", border = "black",
  xlab = "Probability", main = "MCMC Posterior Histogram"
)
dev.off()

# 5. Trace Plot for MCMC
pdf("figures/mcmc_trace_plot.pdf")
ggplot(
  data.frame(Iteration = 1:n_samples, Sample = mcmc_samples),
  aes(x = Iteration, y = Sample)
) +
  geom_line(alpha = 0.2, color = "gray") +
  geom_smooth(color = "blue", method = "loess", se = FALSE) +
  labs(title = "Trace Plot of MCMC Samples",
       x = "Iteration",
       y = "Sampled Probability") +
  theme_minimal()
dev.off()

# =====================================================
# Completion
# =====================================================

cat("Analysis complete. Check ’logs’ and ’figures’ directories for results.\n")
```