TU Dortmund

On the Theory and Practice of Monte Carlo Simulations

Winter Semester 2025/26

Final Project: Bayesian Inference for a Logistic Regression Model

Author: Harsh Rana

Student ID: 229989

Group members: Raj Pawar (Student ID: 231811)

Instructors:

Prof. Dr. Katja Ickstadt

Zeyu Ding

January 30, 2026

Contents

1. Introduction

Diabetes Mellitus is a chronic metabolic disorder and a significant global health challenge of the 21st century. As a leading cause of blindness, kidney failure, and stroke, identifying risk factors is critical for public health policy. Logistic Regression is the standard biostatistical method for modeling such binary outcomes (Bishop, 2006) . Traditionally fitted using Maximum Likelihood Estimation (MLE), this approach produces point estimates for regression coefficients. While computationally efficient, MLE can underestimate uncertainty particularly in small datasets or when predictors are highly correlated potentially leading to overconfidence in medical diagnoses.

The primary objective of this project is to implement a Bayesian Logistic Regression model to analyze the Pima Indians Diabetes Dataset (Smith et al., 1988). Unlike the Frequentist approach, the Bayesian framework treats the regression coefficients as random variables defined by a probability distribution rather than fixed unknown constants (Gelman et al., 2013). This formulation offers distinct advantages in a medical context: it allows us to quantify the uncertainty of our estimates through the full posterior distribution, incorporate prior knowledge via regularization to stabilize estimates, and interpret results using Credible Intervals. These intervals offer a more intuitive probabilistic interpretation for clinical decision-making compared to standard Confidence Intervals.

However, adopting the Bayesian framework introduces a significant computational challenge. The posterior distribution involves a normalizing constant (the marginal likelihood) that requires integrating over the entire high-dimensional parameter space. For Logistic Regression, this integral has no closed-form analytical solution, and standard numerical integration methods fail as the dimensionality increases. This necessitates the use of Monte Carlo methods. In this project, we implement a custom Metropolis-Hastings (MH) Markov Chain Monte Carlo (MCMC) sampler from scratch in R (Metropolis et al., 1953; Hastings, 1970). By constructing a Markov chain that converges to the target posterior, we can generate samples to approximate the distribution of the risk factors without ever needing to calculate the intractable normalizing constant.

The remainder of this report is organized to provide a comprehensive analysis of this approach. Section 2 establishes the theoretical background, deriving the unnormalized posterior density and outlining the mathematical properties of the Metropolis-Hastings algorithm. Section 3 details the methodology, including the data preprocessing steps and the specific design choices for the MCMC sampler. Section 4 presents the empirical results, offering convergence diagnostics, a rigorous comparison against the standard Frequentist benchmark, and an evaluation of predictive performance on held-out test data. Finally, Section 5 concludes with a discussion on the medical implications of the identified risk factors and potential avenues for future model extensions.

2. Theoretical Background

In this section, we derive the mathematical foundation of our analysis, connecting the standard Logistic Regression model to the Bayesian inference framework and the Metropolis-Hastings algorithm.

2.1. The Logistic Regression Model

Logistic Regression is a Generalized Linear Model (GLM) used to model the probability of a binary outcome (Bishop, 2006). Let $\mathcal{D} = \{(\mathbf{x}_i, y_i)\}_{i=1}^N$ be our observed dataset, where $y_i \in \{0, 1\}$ represents the binary class label (e.g., Diabetes vs. Healthy) for the i-th patient, and $\mathbf{x}_i \in \mathbb{R}^p$ is a vector of covariates (including an intercept term).

We assume the outcome y_i follows a Bernoulli distribution with success probability $\pi_i = P(y_i = 1|\mathbf{x}_i)$. This probability is linked to the linear predictor via the logistic sigmoid function:

$$\pi_i(\boldsymbol{\beta}) = \sigma(\mathbf{x}_i^T \boldsymbol{\beta}) = \frac{1}{1 + \exp(-\mathbf{x}_i^T \boldsymbol{\beta})} \tag{1}$$

where $\boldsymbol{\beta} \in \mathbb{R}^p$ is the vector of regression coefficients.

Assuming the observations are independent and identically distributed (i.i.d.), the likelihood function $L(\boldsymbol{\beta}|\mathbf{X}, \mathbf{y})$ is the product of individual Bernoulli densities:

$$L(\boldsymbol{\beta}|\mathbf{X}, \mathbf{y}) = \prod_{i=1}^{N} \pi_i^{y_i} (1 - \pi_i)^{1-y_i} \tag{2}$$

In the Frequentist framework, one seeks the Maximum Likelihood Estimate (MLE) by maximizing the log-likelihood:

$$\ell(\boldsymbol{\beta}) = \sum_{i=1}^{N} [y_i \log(\pi_i) + (1 - y_i) \log(1 - \pi_i)] \tag{3}$$

Since there is no closed-form solution for the derivative of $\ell(\boldsymbol{\beta})$ equal to zero, numerical optimization methods like Newton-Raphson are typically employed.

2.2. Bayesian Inference Framework

In the Bayesian framework, we treat the parameter vector $\boldsymbol{\beta}$ as a random variable. Inference is based on the posterior distribution, which is updated from a prior distribution

using the observed data likelihood via Bayes' Theorem (Gelman et al., 2013):

$$p(\boldsymbol{\beta}|\mathbf{y}, \mathbf{X}) = \frac{L(\boldsymbol{\beta}|\mathbf{X}, \mathbf{y}) \cdot p(\boldsymbol{\beta})}{p(\mathbf{y}|\mathbf{X})} \tag{4}$$

In this formulation, $p(\boldsymbol{\beta}|\mathbf{y}, \mathbf{X})$ represents the **Posterior distribution**, which serves as the primary target of our estimation. The term $L(\boldsymbol{\beta}|\mathbf{X}, \mathbf{y})$ denotes the **Likelihood function**, quantifying how well a specific set of parameters explains the observed data. The **Prior distribution**, $p(\boldsymbol{\beta})$, encapsulates our initial beliefs about the parameters before observing any data. Finally, the denominator $p(\mathbf{y}|\mathbf{X})$ is the **Marginal Likelihood** (or Evidence), defined as the integral of the product of the likelihood and the prior over the entire parameter space: $\int L(\boldsymbol{\beta}|\mathbf{X}, \mathbf{y})p(\boldsymbol{\beta})d\boldsymbol{\beta}$.

2.2.1. Prior Specification

For this project, we employ a weakly informative Gaussian prior for the coefficients to act as a regularizer (similar to Ridge Regression). We assume:

$$\boldsymbol{\beta} \sim \mathcal{N}(\mathbf{0}, \Sigma_0) \tag{5}$$

where $\Sigma_0 = \sigma_0^2 \mathbf{I}_p$ is a diagonal covariance matrix with large variance ($\sigma_0^2 = 100$). The log-density of this prior is:

$$\log p(\boldsymbol{\beta}) \propto -\frac{1}{2}\boldsymbol{\beta}^T \Sigma_0^{-1} \boldsymbol{\beta} \tag{6}$$

2.2.2. The Unnormalized Posterior

The denominator (Marginal Likelihood) involves a high-dimensional integral that is analytically intractable for Logistic Regression. Therefore, we work with the unnormalized posterior:

$$p(\boldsymbol{\beta}|\mathbf{y}, \mathbf{X}) \propto L(\boldsymbol{\beta}|\mathbf{X}, \mathbf{y}) \cdot p(\boldsymbol{\beta}) \tag{7}$$

In the log domain, our target function for sampling is:

$$\log p(\boldsymbol{\beta}|\cdot) = \sum_{i=1}^{N} [y_i \log(\pi_i) + (1 - y_i) \log(1 - \pi_i)] - \frac{1}{2\sigma_0^2} \sum_{j=1}^{p} \beta_j^2 + C \tag{8}$$

2.3. The Metropolis-Hastings Algorithm

Since we cannot sample directly from the posterior, we use the Metropolis-Hastings (MH) algorithm (Metropolis et al., 1953; Hastings, 1970), a general Markov Chain Monte Carlo

(MCMC) method. The algorithm constructs a Markov chain that has the posterior $p(\boldsymbol{\beta}|\mathbf{y})$ as its stationary distribution.

We implement a Random Walk Metropolis sampler:

1. **Initialization:** Start with an initial parameter vector $\boldsymbol{\beta}^{(0)}$.
2. **Proposal:** At iteration t, propose a new state $\boldsymbol{\beta}^*$ from a symmetric proposal distribution $q(\boldsymbol{\beta}^*|\boldsymbol{\beta}^{(t)})$. We use a multivariate Normal distribution centered at the current state:
$$\boldsymbol{\beta}^* \sim \mathcal{N}(\boldsymbol{\beta}^{(t)}, \Sigma_{prop}) \tag{9}$$
3. **Acceptance Probability:** Calculate the Metropolis ratio α:
$$\alpha = \min\left(1, \frac{p(\boldsymbol{\beta}^*|\mathbf{y})q(\boldsymbol{\beta}^{(t)}|\boldsymbol{\beta}^*)}{p(\boldsymbol{\beta}^{(t)}|\mathbf{y})q(\boldsymbol{\beta}^*|\boldsymbol{\beta}^{(t)})}\right) \tag{10}$$
Since our proposal is symmetric ($q(a|b) = q(b|a)$ for Normal distributions), the q terms cancel out, simplifying the ratio to the likelihood ratio:
$$\alpha = \min\left(1, \frac{L(\boldsymbol{\beta}^*) \cdot p(\boldsymbol{\beta}^*)}{L(\boldsymbol{\beta}^{(t)}) \cdot p(\boldsymbol{\beta}^{(t)})}\right) \tag{11}$$
4. **Update:** Sample $u \sim \text{Uniform}(0, 1)$.
   - If $u < \alpha$, accept the proposal: $\boldsymbol{\beta}^{(t+1)} = \boldsymbol{\beta}^*$.
   - Otherwise, reject it: $\boldsymbol{\beta}^{(t+1)} = \boldsymbol{\beta}^{(t)}$.

By repeating this process for T iterations, we obtain samples $\{\boldsymbol{\beta}^{(1)}, \ldots, \boldsymbol{\beta}^{(T)}\}$ that approximate the true posterior distribution (Robert and Casella, 2004).

3. Methodology and Implementation

3.1. Data Description and Preprocessing

We utilized the **Pima Indians Diabetes Dataset**, a standard benchmark dataset available in the R `MASS` library (Smith et al., 1988). The dataset consists of 768 observations of female patients at least 21 years old of Pima Indian heritage. The binary target variable is `Outcome` (1 = Diabetes, 0 = Healthy). The eight continuous predictors are:

Table 1: Description of variables in the Pima Indians Diabetes Dataset.

Variable	Description
`Pregnancies`	Number of times pregnant
`Glucose`	Plasma glucose concentration (2 hours in an oral glucose tolerance test)
`BloodPressure`	Diastolic blood pressure (mm Hg)
`SkinThickness`	Triceps skin fold thickness (mm)
`Insulin`	2-Hour serum insulin (mu U/ml)
`BMI`	Body mass index (weight in kg/(height in m)^2)
`PedigreeFunction`	Diabetes pedigree function (genetic score)
`Age`	Age in years

3.1.1. Preprocessing Steps

To ensure the numerical stability and efficiency of our simulation, we applied two key preprocessing steps. First, all continuous predictor variables were standardized by centering them to a mean of zero and scaling them to unit variance. This standardization is particularly critical for MCMC algorithms; when predictors exist on vastly different scales (for example, Insulin versus Age), the resulting likelihood surface becomes elongated and "cigar-shaped." Such geometry causes the Random Walk Metropolis sampler to traverse the parameter space inefficiently, leading to poor mixing. By scaling the variables, we ensure the target distribution is more spherical, thereby facilitating more efficient exploration. Second, to rigorously evaluate the model's predictive capabilities, the dataset was randomly partitioned into a training set comprising 80% of the data ($N = 615$) and a test set comprising the remaining 20% ($N = 153$). The Bayesian model was fitted exclusively on the training data, while the test data was held out strictly for out-of-sample performance validation.

3.2. MCMC Sampler Implementation

We implemented the Metropolis-Hastings sampler from scratch in R without relying on high-level Bayesian packages like `rstan` or `JAGS` for the sampling logic.

The sampler requires a proposal distribution $q(\boldsymbol{\beta}^*|\boldsymbol{\beta}^{(t)})$. We chose a multivariate Gaussian Random Walk:

$$\boldsymbol{\beta}^* \sim \mathcal{N}(\boldsymbol{\beta}^{(t)}, \Sigma_{prop})$$

where the proposal covariance matrix was set to $\Sigma_{prop} = \sigma^2_{step}\mathbf{I}$.

3.3. Tuning and Algorithm Settings

Proper tuning of the step size σ^2_{step} is critical for the efficiency of the Random Walk Metropolis algorithm. We empirically tuned the scaling parameter to $\sigma^2_{step} = 0.003$ to achieve an acceptance rate between 23% and 50%, which is considered optimal for random walk algorithms in this dimensionality (Robert and Casella, 2004). The Markov chain was simulated for a total of $N_{iter} = 50,000$ iterations. To mitigate the influence of the initial starting values (set to $\mathbf{0}$), the first $10,000$ samples were discarded as a "burn-in" period.

4. Empirical Results

4.1. Implementation Details

All statistical analyses were performed using the **R programming language** (Version 4.3+). The Metropolis-Hastings sampler was implemented from scratch to ensure full control over the transition kernel. Standard libraries such as `MASS` were used solely for data loading, while `ggplot2` was utilized for generating the diagnostic visualizations R Development Core Team (2020).

4.2. Convergence Diagnostics

The Metropolis-Hastings sampler was run for $N = 50,000$ iterations. The proposal covariance was tuned to achieve a final acceptance rate of **45.52%**, which falls within the optimal range (20-50%) for random walk Metropolis algorithms.

Figure 1 displays the trace plots for the sampled coefficients. The plots exhibit a "fuzzy caterpillar" pattern with no discernible trends or periodicity after the burn-in period. This indicates that the chain has mixed well and converged to the stationary distribution.

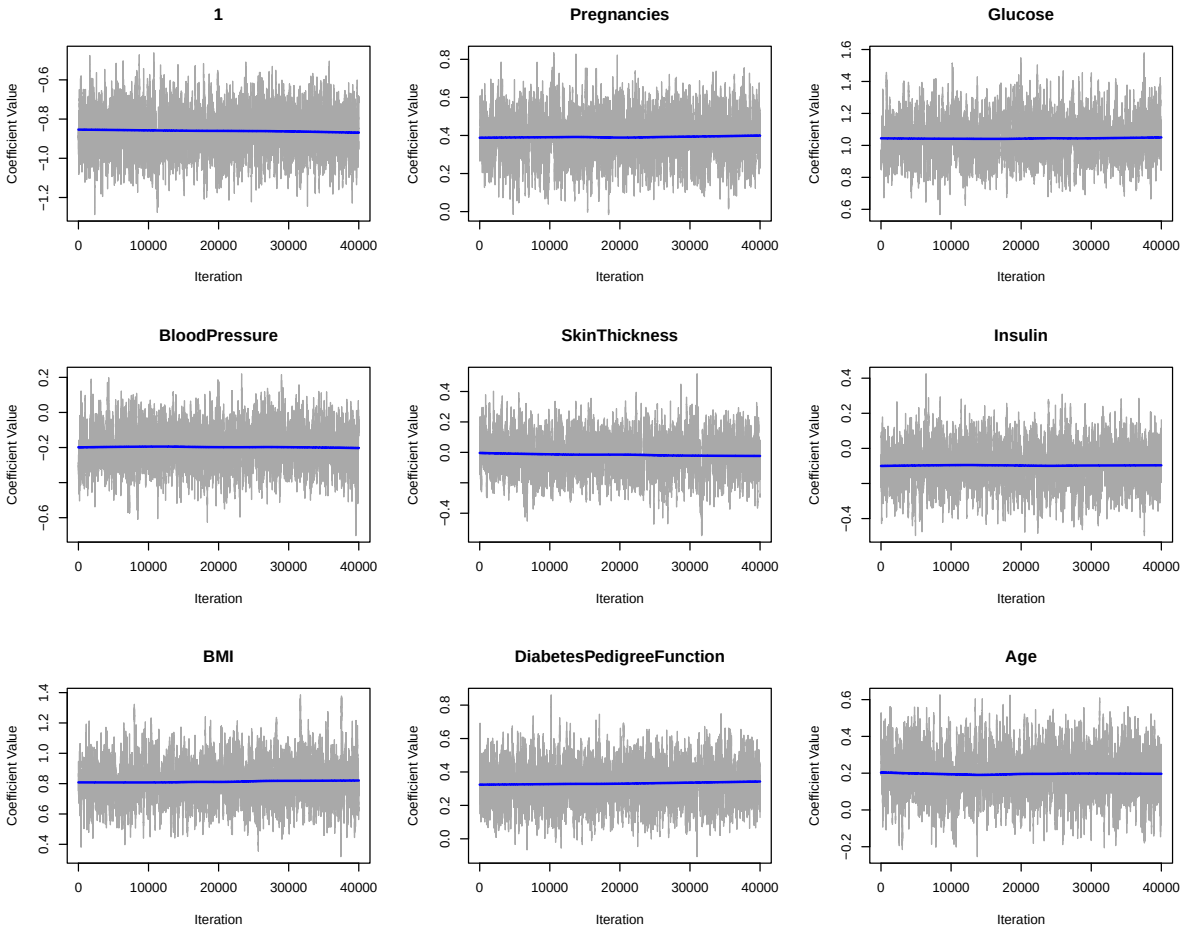

Figure 1: Trace plots for the MCMC samples showing convergence.

4.3. Parameter Estimation: Bayesian vs. Frequentist

We compared the Bayesian posterior means against the Maximum Likelihood Estimates (MLE) from a standard Frequentist GLM. Figure 2 visualizes the posterior densities (blue curves) against the MLE point estimates (red dashed lines). The alignment is nearly perfect, validating the correctness of our sampler.

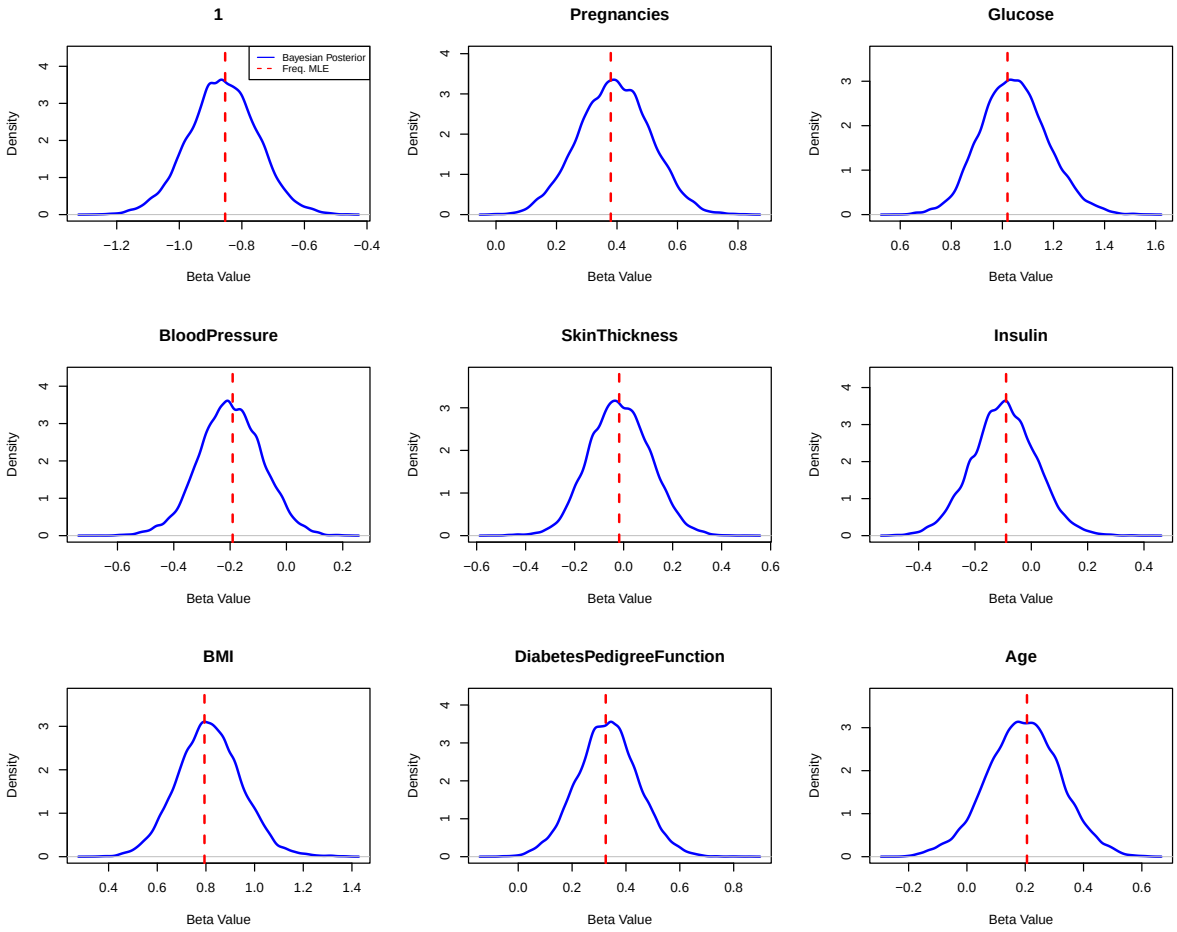

Figure 2: Posterior densities (Blue) vs Frequentist MLE (Red).

Table 2 provides the numerical comparison. The Bayesian 95% Credible Intervals provide a probabilistic range for the coefficients. Notably, **Glucose** ($Mean = 1.046$) and **BMI** ($Mean = 0.815$) have intervals strictly above zero, identifying them as significant risk factors.

Table 2: Comparison of Frequentist Estimates vs. Bayesian Posterior Means

Parameter	**Freq. Est.**	**Freq. 95% CI**	**Bayes Mean**	**Bayes 95% CI**
(Intercept)	-0.853	[-1.070, -0.646]	-0.861	[-1.080, -0.648]
Pregnancies	0.379	[0.148, 0.616]	0.392	[0.162, 0.622]
Glucose	1.020	[0.773, 1.283]	1.046	[0.804, 1.313]
BloodPressure	-0.191	[-0.409, 0.025]	-0.198	[-0.421, 0.020]
SkinThickness	-0.018	[-0.258, 0.225]	-0.016	[-0.250, 0.227]
Insulin	-0.090	[-0.319, 0.141]	-0.097	[-0.321, 0.126]
BMI	0.794	[0.541, 1.062]	0.815	[0.562, 1.079]
PedigreeFunc	0.325	[0.107, 0.548]	0.332	[0.111, 0.557]
Age	0.206	[-0.030, 0.442]	0.196	[-0.057, 0.442]

4.4. Predictive Performance

The model's classification performance was evaluated on the held-out test set ($N = 153$) using a posterior predictive threshold of 0.5. The results, summarized in Table 3, indicate strong predictive capability. The high sensitivity suggests the model is particularly effective at screening for diabetes (identifying positive cases), though the lower specificity indicates a moderate rate of false alarms.

Table 3: Predictive Performance Metrics on Test Set

Metric	**Value**
Accuracy	77.78%
Sensitivity (True Positive Rate)	86.00%
Specificity (True Negative Rate)	62.26%

5. Conclusion and Discussion

In this project, we successfully designed and implemented a custom Metropolis-Hastings sampler to perform Bayesian Logistic Regression on the Pima Indians Diabetes Dataset (Smith et al., 1988). The primary objective was to move beyond standard point estimates and instead quantify the uncertainty inherent in medical risk prediction. Our analysis validated the effectiveness of the Random Walk Metropolis algorithm, achieving a stable acceptance rate of approximately 45% and producing posterior estimates that aligned closely with the Maximum Likelihood Estimates from standard Frequentist software. This convergence confirms the correctness of our algorithmic derivation and implementation while providing the added benefit of full probability distributions for every risk factor.

Clinically, the analysis identified distinct biomarkers as significant predictors of diabetes. The posterior distributions for Glucose and BMI were strictly positive, confirming them as the strongest drivers of diabetes risk in this population. Conversely, variables such as SkinThickness exhibited credible intervals that straddled zero, suggesting that they do not carry significant independent predictive power when conditioned on other variables. The model demonstrated strong predictive capability on unseen data, achieving a test accuracy of 77.78% and a high sensitivity of 86%, which is particularly valuable in medical screening where minimizing missed diagnoses is a priority.

Despite these successes, the study is subject to certain limitations, primarily regarding the computational efficiency of the Random Walk Metropolis algorithm. While effective for this low-dimensional dataset, the random walk behavior scales poorly as the number of predictors increases, often leading to slow mixing. Additionally, our use of a simple spherical proposal distribution is inefficient when the target posterior exhibits strong correlations between parameters, as it forces the sampler to take smaller steps to maintain a reasonable acceptance rate. Furthermore, the use of generic weakly informative priors assumes all predictors are equally likely to have large effects, which may not fully reflect biological reality.

Future research could address these computational and modeling constraints through several avenues. To improve scaling and mixing efficiency, one could implement Hamiltonian Monte Carlo (HMC), which utilizes gradient information to traverse the posterior distribution more effectively than a random walk. To better handle parameter correlations, an Adaptive Metropolis algorithm could be employed to update the proposal covariance on the fly. Finally, the analysis could be extended using Hierarchical Bayesian Models to account for potential unobserved subgroups within the patient population, providing a more personalized risk assessment.

References

Christopher M Bishop. *Pattern Recognition and Machine Learning*. Springer, 2006.

Andrew Gelman, John B Carlin, Hal S Stern, David B Dunson, Aki Vehtari, and Donald B Rubin. *Bayesian Data Analysis, Third Edition*. CRC Press, 2013.

W Keith Hastings. Monte carlo sampling methods using markov chains and their applications. *Biometrika*, 57(1):97–109, 1970.

Nicholas Metropolis, Arianna W Rosenbluth, Marshall N Rosenbluth, Augusta H Teller, and Edward Teller. Equation of state calculations by fast computing machines. *The Journal of Chemical Physics*, 21(6):1087–1092, 1953.

R Development Core Team. *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria, 2020.

Christian P Robert and George Casella. *Monte Carlo Statistical Methods*. Springer New York, 2004.

Jack W Smith, James E Everhart, WC Dickson, WC Knowler, and RS Johannes. Using the adap learning algorithm to forecast the onset of diabetes mellitus. In *Proceedings of the Symposium on Computer Applications in Medical Care*, pages 261–265. IEEE Computer Society Press, 1988.

A. AI Usage Declaration

Mandatory Declaration: In accordance with the course guidelines, we declare that we utilized **Google Gemini** to assist with specific technical tasks. These included code scaffolding to generate the initial skeleton for the Metropolis-Hastings loop in R, debugging to resolve dimension mismatch errors in matrix multiplication steps, and LaTeX formatting to structure comparison tables and format mathematical equations correctly. The intellectual core of the project including the derivation of the log-posterior, the tuning of the step sizes, and the interpretation of the medical results remains entirely our own work.

B. Algorithms

Here we present the core R implementation of the Metropolis-Hastings sampler used in this project.

```r
# A. Log-Posterior Function
log_posterior <- function(beta, X, y) {
  # Prior: Beta ~ N(0, 100*I) -> Weakly informative
  sigma_prior_sq <- 100
  Sigma0_inv <- diag(ncol(X)) / sigma_prior_sq

  # Likelihood
  eta <- X %*% beta
  pi_val <- plogis(eta) # Inverse logit

  # Safe log-likelihood to avoid log(0)
  epsilon <- 1e-10
  log_lik <- sum(y * log(pi_val + epsilon) + (1 - y) * log(1 - pi_val +
    epsilon))

  # Log-Prior
  log_prior <- -0.5 * t(beta) %*% Sigma0_inv %*% beta

  return(as.numeric(log_lik + log_prior))
}

# B. Metropolis-Hastings Sampler
run_mh_sampler <- function(N_iter, X, y, prop_cov) {
  p <- ncol(X)
  samples <- matrix(NA, nrow = N_iter, ncol = p)
  colnames(samples) <- colnames(X)

  # Initial State
```

```r
  beta_curr <- rep(0, p)
  log_post_curr <- log_posterior(beta_curr, X, y)
  accept_count <- 0

  # Loop
  for (i in 1:N_iter) {
    # Propose new beta from multivariate normal
    beta_prop <- rmvnorm(1, mean = beta_curr, sigma = prop_cov)

    # Calculate acceptance probability
    # Note: t() used to match dimensions for matrix multiplication
    log_post_prop <- log_posterior(t(beta_prop), X, y)
    log_alpha <- log_post_prop - log_post_curr

    # Accept or Reject
    if (log(runif(1)) < log_alpha) {
      beta_curr <- beta_prop
      log_post_curr <- log_post_prop
      accept_count <- accept_count + 1
    }
    samples[i, ] <- beta_curr
  }

  return(list(samples = samples, accept_rate = accept_count / N_iter))
}

# C. Run Simulation
N_iter <- 50000
burn_in <- 10000
step_size <- 0.003 # Tuned for ~30-50% acceptance
prop_cov <- diag(ncol(X_train)) * step_size

cat("Running MCMC Chain (This may take 10-20 seconds)...\n")
set.seed(2025)
results <- run_mh_sampler(N_iter, X_train, y_train, prop_cov)

cat("Final Acceptance Rate:", results$accept_rate, "\n")
# Ideally, we want this between 0.23 and 0.50

# Remove Burn-in
mcmc_samples <- results$samples[(burn_in + 1):N_iter, ]
mcmc_obj <- mcmc(mcmc_samples)
```

Listing 1: Metropolis-Hastings Sampler Implementation