# Bayesian Computing

Name :- Preksha A. Patel
sapid :- 60004210126
Branch :- Computer Engineering
Div :- C2

## Experiment no. 5

**Aim :-** To implement Logistic Modeling : A math department is interested in exploring the relationship between students scores on the ACT test, a standard college entrance exam, and their success (getting an A or B) in a Business calculus class. Data were obtained for a sample of students.

**Theory :-**

**Bayesian Logistic Modeling :**

Bayesian logistic Modeling is a statiscal approach used to model the relationship between a binary outcome variable and one or more predictor variables. The binary outcome variable $y_i$ follows a binomial distribution with parameters $n_i$ and $p_i$ where $p_i$ is determined by a logistic regression model.

**Model Description :**

1. **Logistic Regression Model :-**

The logistic regression model specifies the relationship between the log-odds of the probability of success ($p_i$) and the predictor variable ($x_i$) through the logistic function. The model is expressed as :-

$$\log\left(\frac{p_i}{1-p_i}\right) = \beta_0 + \beta_1 x_i, \text{ where}$$

$\beta_0$ is the intercept and $\beta_1$ is the coeffecient associated with the predictor variable $x_i$. This equation is then used to calculate $p_i$ as follows : $p_i = \dfrac{e^{(\beta_0 + \beta_1 x_i)}}{1 + e^{(\beta_0 + \beta_1 x_i)}}$

This logistic function ensures that $p_i$ lies between 0 & 1

2. Likelihood function :-

Given the binary nature of the outcome variable, the likelihood function for the parameters $\beta_0$ & $\beta_1$ is specified as :

$$L(\beta_0, \beta_1) = \prod_{i=1}^{n} {}^{n_i}C_{y_i} \, p_i^{y_i} (1-p_i)^{n_i - y_i}$$, where

${}^{n_i}C_{y_i}$ is the binomial coefficient, $n$ is the number of observations. In the context of Bayesian Analysis, the likelihood function quantifies the probability of observing the given data $(y_i)$ under the specified logistic regression model.

3. Prior Distribution :-

Mathematically, represented as :- $\pi(\beta_0, \beta_1) \propto 1$

The flat prior reflects a lack of prior knowledge or bias about the values of the regression parameters.

4. Posterior Density :-

When a flat non-informative prior is used, the posterior density is proportional to the product of the likelihood function and the prior distribution. Mathematically,

$$\pi(\beta_0, \beta_1 \mid data) \propto L(\beta_0, \beta_1)$$

This posterior distribution captures the updated beliefs about the parameters after observing the data.

Conclusion :-

In conclusion, we learned how to use Bayesian methods to model binary data and how to assess the model performance using posterior predictive checks.

SAP ID.: 60004210126 Name: Preksha Ashok Patel

Batch: C2-1

Subject: Bayesian Computing Laboratory Semester: VII

# Experiment No. 5

## Aim:

Implement Logistic Modeling: A math department is interested in exploring the relationship between students' scores on the ACT test, a standard college entrance exam, and their success (getting an A or a B) in a business calculus class. Data were obtained for a sample of students.

Code:

Importing Libraries:

```
library(brms)
library(ggplot2)
```

Generate Data for given Scenario:

```
set.seed(123) n <- 100  #

Number of students

ACT_scores <- rnorm(n, mean = 25, sd = 5)

Success <- ifelse(ACT_scores + rnorm(n) > 25, "A", "B")
calculus_data <- data.frame(ACT_scores, Success)
```

Data Exploration and Visualization:

```
ggplot(calculus_data, aes(x = ACT_scores, fill = Success)) +

+     geom_histogram(binwidth = 2, position = "identity", alpha = 0.7) +

+     labs(title = "Distribution of ACT Scores by Success",

+          x = "ACT Scores", y = "Frequency",

+          fill = "Success")
```

Logistic Regression Modeling using Bayesian Approach:

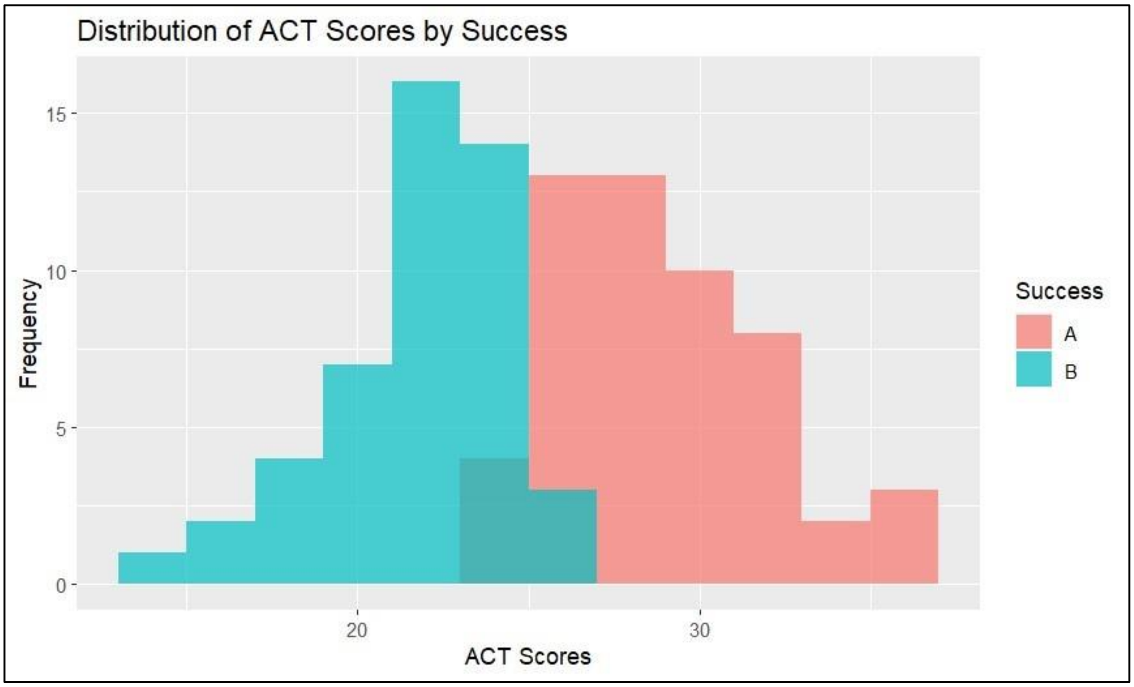
```
calculus_data$Success_binary <- as.numeric(calculus_data$Success == "A") model <-

brm(Success_binary ~ ACT_scores, data = calculus_data, family = bernoulli())



summary(model)
```

Posterior Predictive Checks:

```
posterior_preds <- posterior_predict(model)

ggplot(calculus_data, aes(x = ACT_scores, y = colMeans(posterior_preds), color =
Success)) +

+     geom_point() +

+     geom_line(aes(y = colMeans(posterior_preds) - 1.96 * sd(posterior_preds)), linetype =
"dashed", alpha = 0.5) +

+     geom_line(aes(y = colMeans(posterior_preds) + 1.96 * sd(posterior_preds)), linetype =
"dashed", alpha = 0.5) +

+     labs(title = "Posterior Predictive Checks",

+          x = "ACT Scores", y = "Probability of Success",

+          color = "Success")
```

Output:

Data Exploration and Visualization:



Logistic Regression Modeling using Bayesian Approach:

```
Compiling Stan program...
Start sampling

SAMPLING FOR MODEL 'anon_model' NOW (CHAIN 1).
Chain 1:
Chain 1: Gradient evaluation took 3.4e-05 seconds
Chain 1: 1000 transitions using 10 leapfrog steps per transition would take 0.34 seconds.
Chain 1: Adjust your expectations accordingly!
Chain 1:
Chain 1:
Chain 1: Iteration:    1 / 2000 [  0%]  (Warmup)
Chain 1: Iteration:  200 / 2000 [ 10%]  (Warmup)
Chain 1: Iteration:  400 / 2000 [ 20%]  (Warmup)
Chain 1: Iteration:  600 / 2000 [ 30%]  (Warmup)
Chain 1: Iteration:  800 / 2000 [ 40%]  (Warmup)
Chain 1: Iteration: 1000 / 2000 [ 50%]  (Warmup)
Chain 1: Iteration: 1001 / 2000 [ 50%]  (Sampling)
Chain 1: Iteration: 1200 / 2000 [ 60%]  (Sampling)
Chain 1: Iteration: 1400 / 2000 [ 70%]  (Sampling)
Chain 1: Iteration: 1600 / 2000 [ 80%]  (Sampling)
Chain 1: Iteration: 1800 / 2000 [ 90%]  (Sampling)
Chain 1: Iteration: 2000 / 2000 [100%]  (Sampling)
Chain 1:
Chain 1:  Elapsed Time: 0.046 seconds (Warm-up)
Chain 1:                0.045 seconds (Sampling)
Chain 1:                0.091 seconds (Total)
Chain 1:
```

```
SAMPLING FOR MODEL 'anon_model' NOW (CHAIN 2).
Chain 2:
Chain 2: Gradient evaluation took 9e-06 seconds
Chain 2: 1000 transitions using 10 leapfrog steps per transition would take 0.09 seconds.
Chain 2: Adjust your expectations accordingly!
Chain 2:
Chain 2:
Chain 2: Iteration:    1 / 2000 [  0%]  (Warmup)
Chain 2: Iteration:  200 / 2000 [ 10%]  (Warmup)
Chain 2: Iteration:  400 / 2000 [ 20%]  (Warmup)
Chain 2: Iteration:  600 / 2000 [ 30%]  (Warmup)
Chain 2: Iteration:  800 / 2000 [ 40%]  (Warmup)
Chain 2: Iteration: 1000 / 2000 [ 50%]  (Warmup)
Chain 2: Iteration: 1001 / 2000 [ 50%]  (Sampling)
Chain 2: Iteration: 1200 / 2000 [ 60%]  (Sampling)
Chain 2: Iteration: 1400 / 2000 [ 70%]  (Sampling)
Chain 2: Iteration: 1600 / 2000 [ 80%]  (Sampling)
Chain 2: Iteration: 1800 / 2000 [ 90%]  (Sampling)
Chain 2: Iteration: 2000 / 2000 [100%]  (Sampling)
Chain 2:
Chain 2:  Elapsed Time: 0.044 seconds (Warm-up)
Chain 2:                0.047 seconds (Sampling)
Chain 2:                0.091 seconds (Total)
Chain 2:
```

```
SAMPLING FOR MODEL 'anon_model' NOW (CHAIN 3).
Chain 3:
Chain 3: Gradient evaluation took 1.1e-05 seconds
Chain 3: 1000 transitions using 10 leapfrog steps per transition would take 0.11 seconds.
Chain 3: Adjust your expectations accordingly!
Chain 3:
Chain 3:
Chain 3: Iteration:    1 / 2000 [  0%]  (Warmup)
Chain 3: Iteration:  200 / 2000 [ 10%]  (Warmup)
Chain 3: Iteration:  400 / 2000 [ 20%]  (Warmup)
Chain 3: Iteration:  600 / 2000 [ 30%]  (Warmup)
Chain 3: Iteration:  800 / 2000 [ 40%]  (Warmup)
Chain 3: Iteration: 1000 / 2000 [ 50%]  (Warmup)
Chain 3: Iteration: 1001 / 2000 [ 50%]  (Sampling)
Chain 3: Iteration: 1200 / 2000 [ 60%]  (Sampling)
Chain 3: Iteration: 1400 / 2000 [ 70%]  (Sampling)
Chain 3: Iteration: 1600 / 2000 [ 80%]  (Sampling)
Chain 3: Iteration: 1800 / 2000 [ 90%]  (Sampling)
Chain 3: Iteration: 2000 / 2000 [100%]  (Sampling)
Chain 3:
Chain 3:  Elapsed Time: 0.05 seconds (Warm-up)
Chain 3:                0.038 seconds (Sampling)
Chain 3:                0.088 seconds (Total)
Chain 3:
```

```
SAMPLING FOR MODEL 'anon_model' NOW (CHAIN 4).
Chain 4:
Chain 4: Gradient evaluation took 7e-06 seconds
Chain 4: 1000 transitions using 10 leapfrog steps per transition would take 0.07 seconds.
Chain 4: Adjust your expectations accordingly!
Chain 4:
Chain 4:
Chain 4: Iteration:    1 / 2000 [  0%]  (Warmup)
Chain 4: Iteration:  200 / 2000 [ 10%]  (Warmup)
Chain 4: Iteration:  400 / 2000 [ 20%]  (Warmup)
Chain 4: Iteration:  600 / 2000 [ 30%]  (Warmup)
Chain 4: Iteration:  800 / 2000 [ 40%]  (Warmup)
Chain 4: Iteration: 1000 / 2000 [ 50%]  (Warmup)
Chain 4: Iteration: 1001 / 2000 [ 50%]  (Sampling)
Chain 4: Iteration: 1200 / 2000 [ 60%]  (Sampling)
Chain 4: Iteration: 1400 / 2000 [ 70%]  (Sampling)
Chain 4: Iteration: 1600 / 2000 [ 80%]  (Sampling)
Chain 4: Iteration: 1800 / 2000 [ 90%]  (Sampling)
Chain 4: Iteration: 2000 / 2000 [100%]  (Sampling)
Chain 4:
Chain 4:  Elapsed Time: 0.041 seconds (Warm-up)
Chain 4:                0.049 seconds (Sampling)
Chain 4:                0.09 seconds (Total)
Chain 4:
```

Model Summary:

```
 Family: bernoulli
  Links: mu = logit
Formula: Success_binary ~ ACT_scores
   Data: calculus_data (Number of observations: 100)
  Draws: 4 chains, each with iter = 2000; warmup = 1000; thin = 1;
         total post-warmup draws = 4000

Population-Level Effects:
           Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
Intercept    -59.83     15.57   -95.02   -34.91 1.00     1497     1834
ACT_scores     2.40      0.62     1.41     3.82 1.00     1480     1741

Draws were sampled using sampling(NUTS). For each parameter, Bulk_ESS
and Tail_ESS are effective sample size measures, and Rhat is the potential
scale reduction factor on split chains (at convergence, Rhat = 1).
```

Posterior Predictive Checks: