1.1

CGS698c-Assignment -2

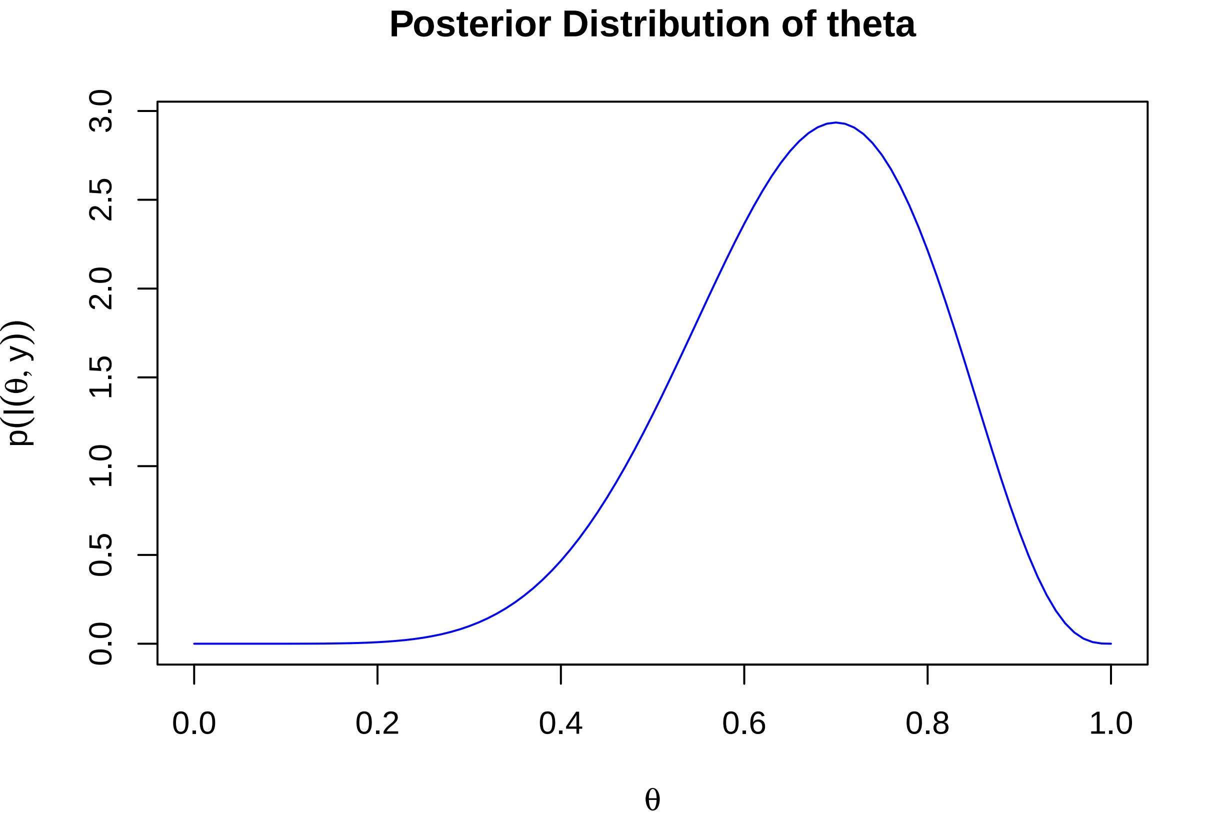
Anil Yadav(210138)

```
# Define the likelihood function

likelihood <- function(theta, y, n = 10) {
  return(choose(n, y) * theta^y * (1 - theta)^(n - y))
}

# Given data
y <- 7

# Marginal likelihood
marginal_likelihood <- 1 / 11

# Calculate the posterior density
posterior <- function(theta, y, marginal_likelihood) {
  # Prior is 1 for 0 <= theta <= 1
  prior <- ifelse(0 <= theta & theta <= 1, 1, 0)
  # Likelihood
  L <- likelihood(theta, y)
  # Posterior using Bayes' rule
  return((L * prior) / marginal_likelihood)
}

# Values of theta to evaluate
theta_values <- c(0.75, 0.25, 1)

# Calculate and print posterior densities for the given theta values
posterior_densities <- sapply(theta_values, function(theta) posterior(theta, y,
marginal_likelihood))
names(posterior_densities) <- theta_values
posterior_densities
```

for theta=0.75, posterior_density=2.75310518
for theta=0.25, posterior_density=0.3398895
for theta=1, posterior_density=0.00

1.2

```
# Step 1: Create a vector of equidistant theta values
theta_values <- seq(0, 1, by = 0.01)

# Step 2: Calculate posterior density for each theta value
posterior_densities <- sapply(theta_values, function(theta) posterior(theta, y,
marginal_likelihood))

# Step 3: Plot the graph
plot(theta_values, posterior_densities, type = "l", col = "blue",
     xlab = expression(theta), ylab = expression(p(theta | y)),
     main = "Posterior Distribution of theta")
```



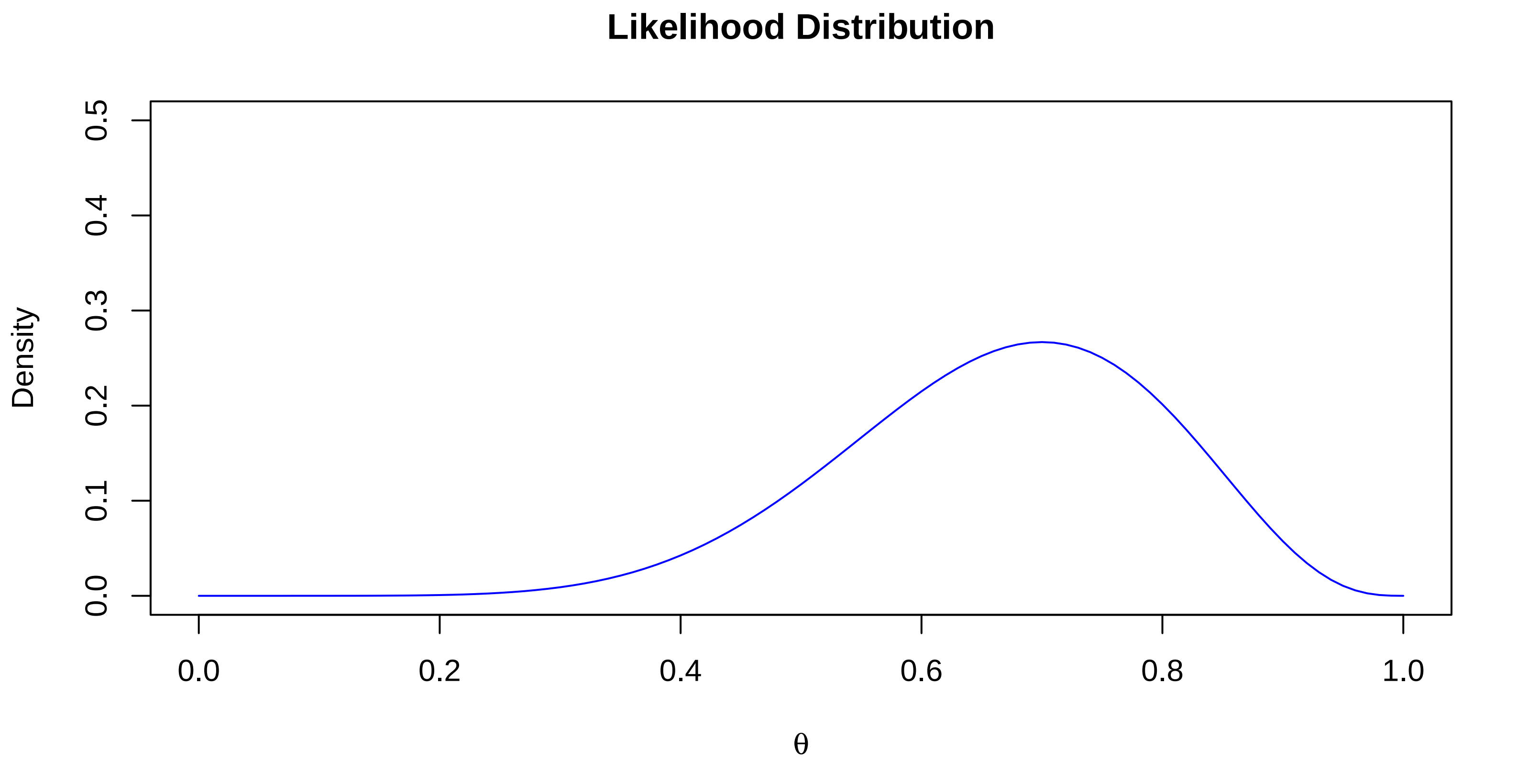
Posterior Distribution of theta
0.0
0.5
1.0
1.5
2.0
2.5
3.0
p(l(θ, y))
0.0
0.2
0.4
0.6
0.8
1.0
θ

1.3

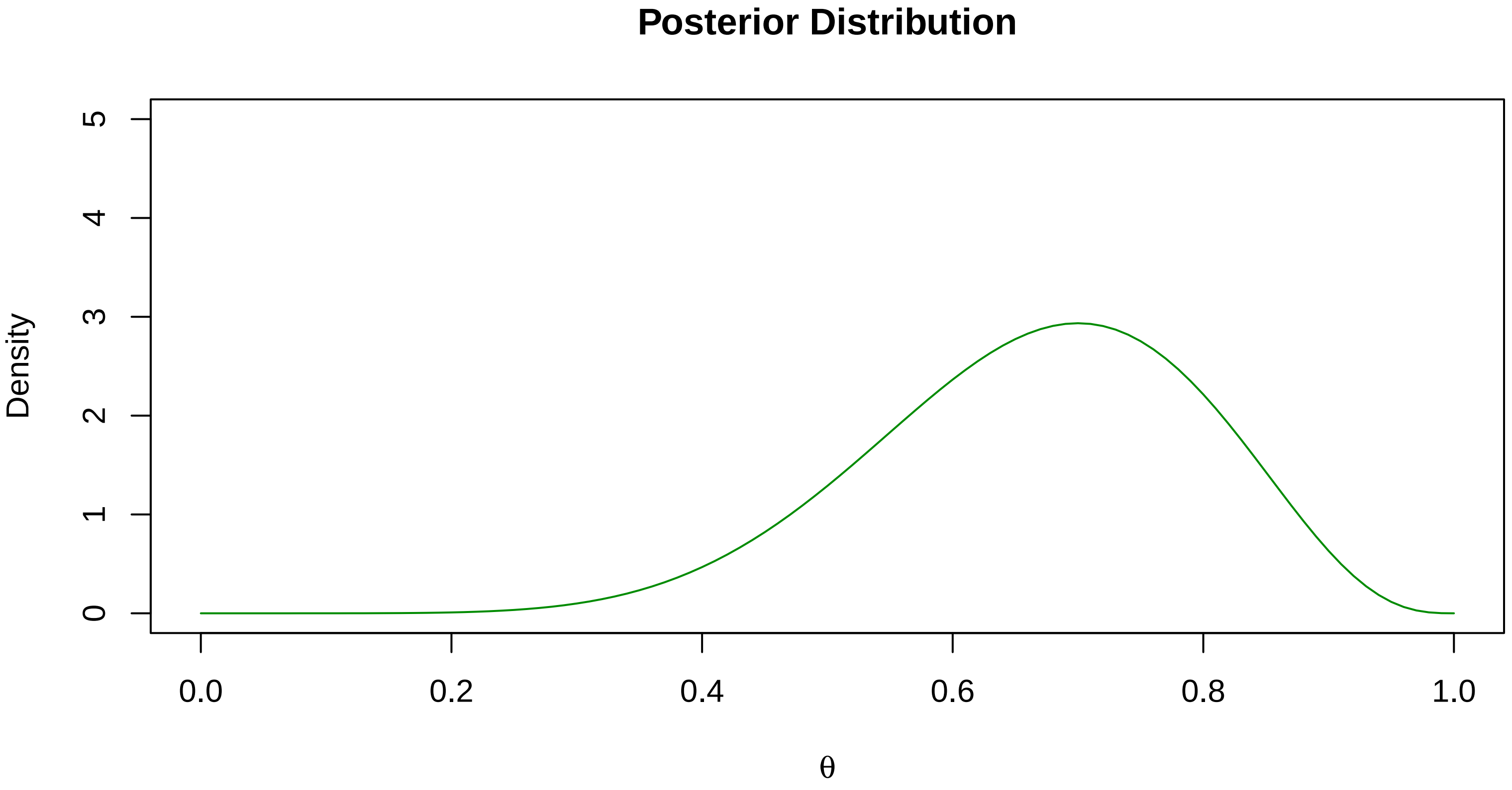
```
# Find the value of theta with the maximum posterior density
theta_max_posterior <- theta_values[which.max(posterior_densities)]

theta_max_posterior

# obtained value is 0.699699
```

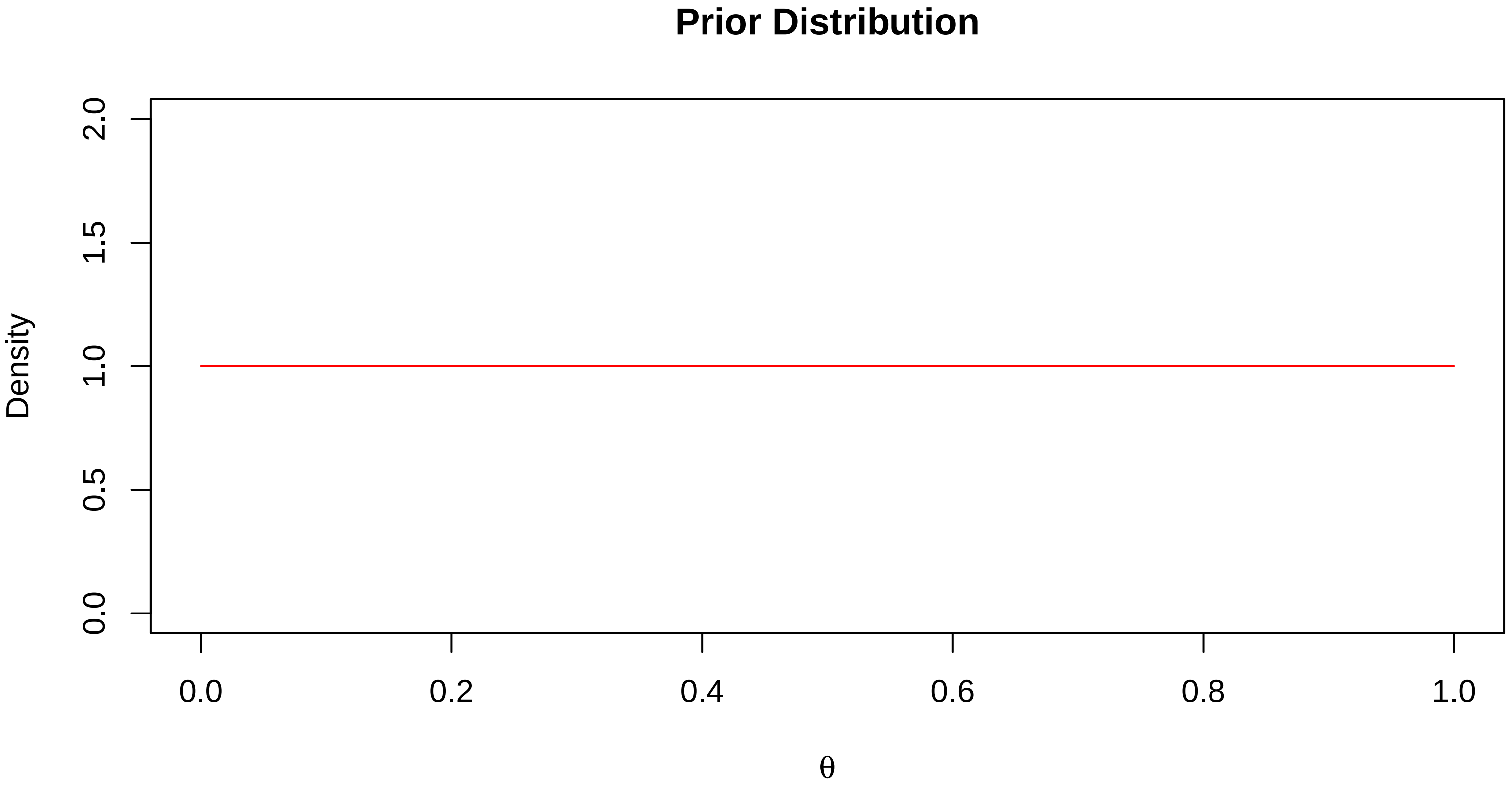
1.4

```
# Define the likelihood function
likelihood <- function(theta, y, n = 10) {
  return(choose(n, y) * theta^y * (1 - theta)^(n - y))
}

# Define the prior distribution
prior <- function(theta) {
  return(ifelse(0 <= theta & theta <= 1, 1, 0))
}

# Given data
y <- 7

# Marginal likelihood
marginal_likelihood <- 1 / 11

# Calculate the posterior density
posterior <- function(theta, y, marginal_likelihood) {
  # Prior is 1 for 0 <= theta <= 1
  prior <- ifelse(0 <= theta & theta <= 1, 1, 0)
  # Likelihood
  L <- likelihood(theta, y)
  # Posterior using Bayes' rule
  return((L * prior) / marginal_likelihood)
}

# Values of theta to evaluate
theta_values <- seq(0, 1, by = 0.01)

# Calculate likelihood, prior, and posterior densities
likelihood_densities <- likelihood(theta_values, y)
prior_densities <- prior(theta_values)
posterior_densities <- sapply(theta_values, function(theta) posterior(theta, y,
marginal_likelihood))

# Plot likelihood
plot(theta_values, likelihood_densities, type = "l", col = "blue",
     xlab = expression(theta), ylab = "Density",
     main = "Likelihood Distribution", ylim = c(0, 0.5))

# Plot prior
plot(theta_values, prior_densities, type = "l", col = "red",
     xlab = expression(theta), ylab = "Density",
     main = "Prior Distribution", ylim = c(0, 2))

# Plot posterior
plot(theta_values, posterior_densities, type = "l", col = "green4",
     xlab = expression(theta), ylab = "Density",
     main = "Posterior Distribution", ylim = c(0, 5))
```



Likelihood Distribution
Density
0.0
0.1
0.2
0.3
0.4
0.5
0.0
0.2
0.4
0.6
0.8
1.0
θ


Posterior Distribution
Density
θ
0
1
2
3
4
5
0.0
0.2
0.4
0.6
0.8
1.0

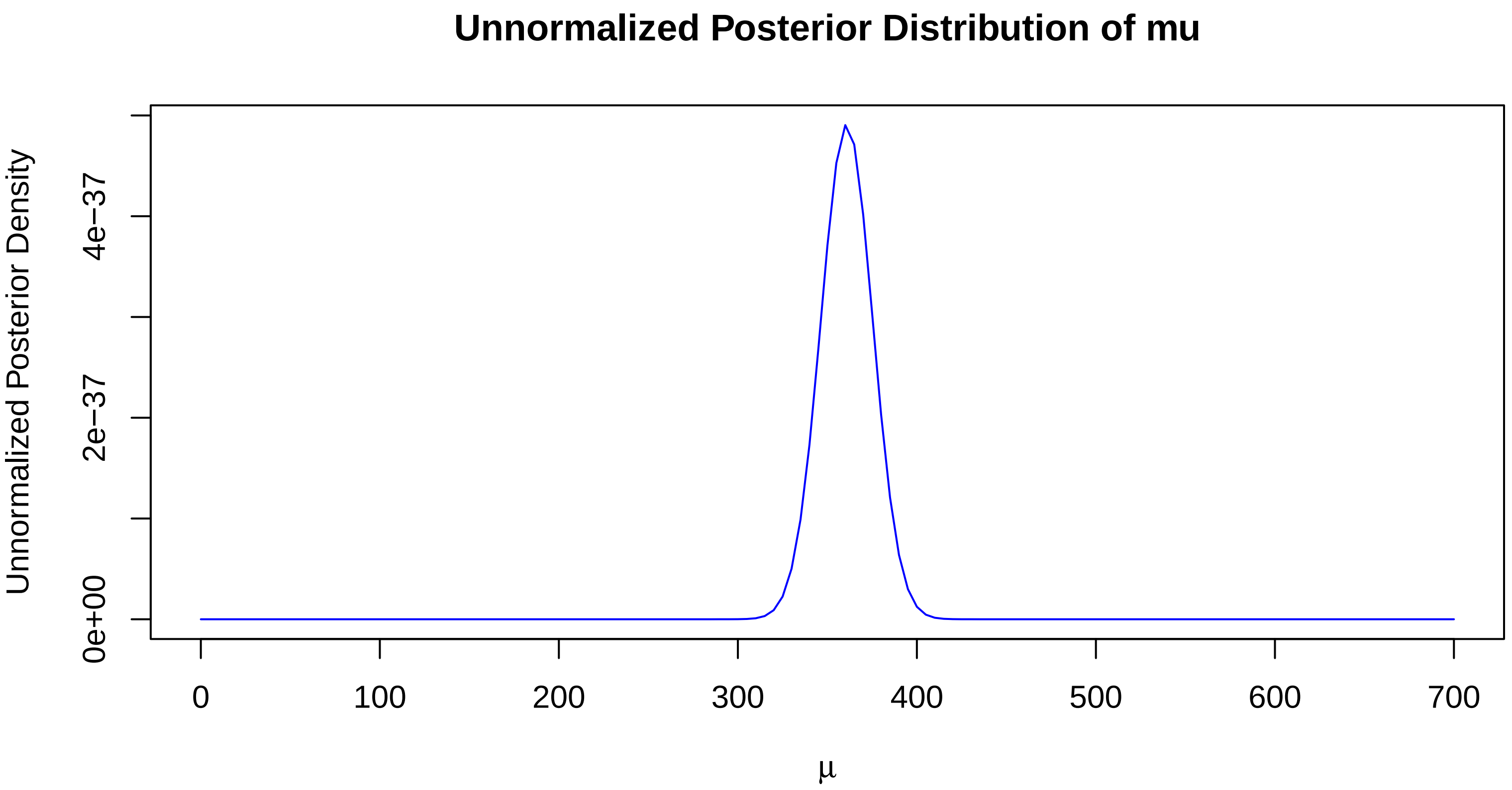

Prior Distribution
Density
2.0
1.5
1.0
0.5
0.0
0.0
0.2
0.4
0.6
0.8
1.0
θ

2.1

```
# Likelihood function
likelihood <- function(mu, sigma, y) {
  n <- length(y)
  return((1 / (sigma * sqrt(2 * pi)))^n * exp(-(1 / (2 * sigma^2)) * sum((y - mu)^2)))
}

# Prior distribution for µ
prior <- function(mu) {
  return(dnorm(mu, mean = 250, sd = 25))
}

# Given data
y <- c(300, 270, 390, 450, 500, 290, 680, 450)

# Values of µ to evaluate
mu_values <- c(300, 900, 50)

# Standard deviation σ
sigma <- 50

# Calculate unnormalized posterior density for each value of µ
posterior_density <- sapply(mu_values, function(mu) {
  likelihood(mu, sigma, y) * prior(mu)
})

posterior_density

## [1] for mu=300 we got posterior_density=6.824248e-41
## [2] for mu=900 we got posterior denisty=0.000000e+00
## [3] for mu=900 we got posterior denisty=9.691374e-138
```

2.2

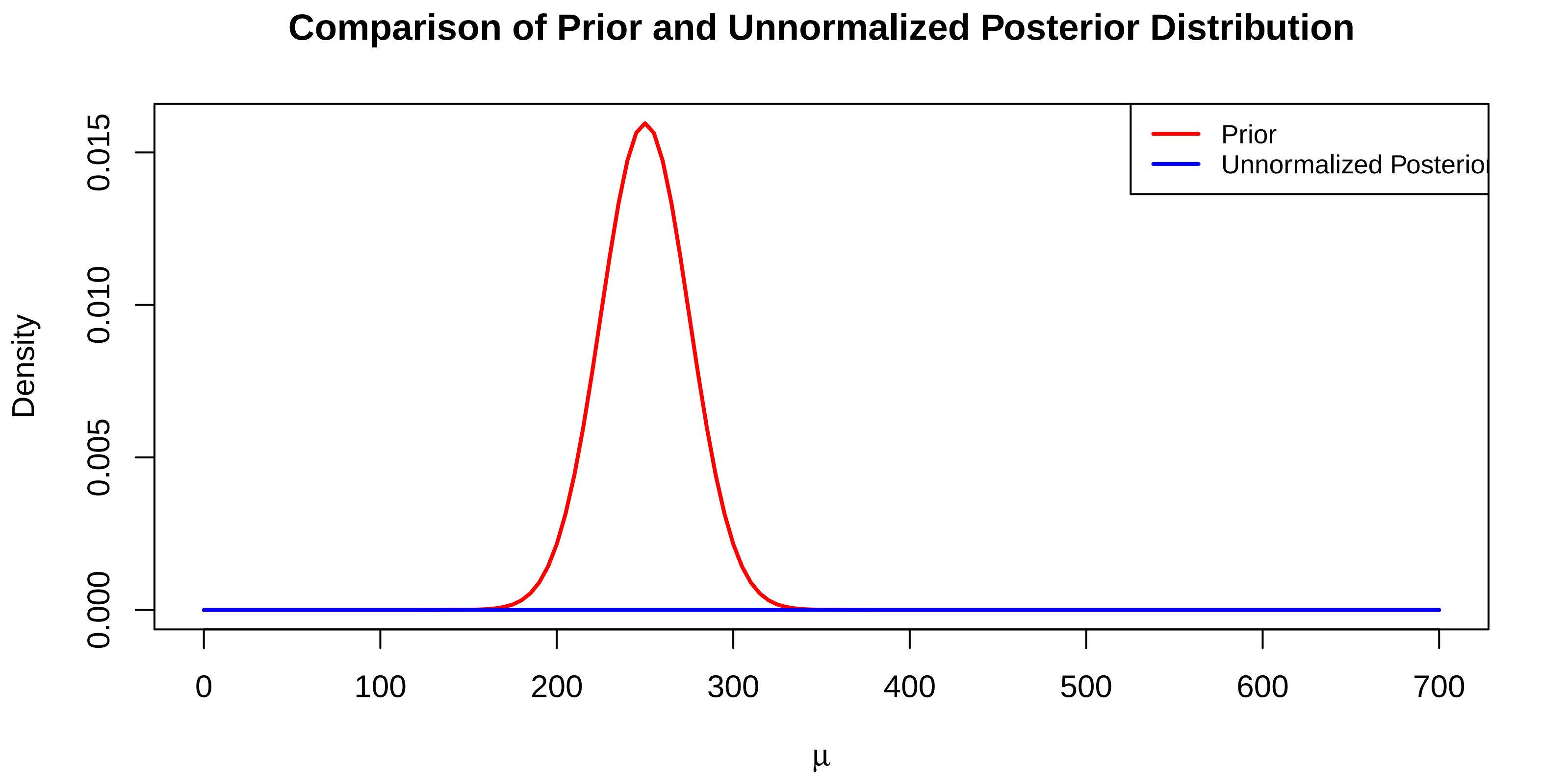
```
# Likelihood function
likelihood <- function(mu, sigma, y) {
  n <- length(y)
  return((1 / (sigma * sqrt(2 * pi)))^n * exp(-(1 / (2 * sigma^2)) * sum((y - mu)^2)))
}

# Prior distribution for μ
prior <- function(mu) {
  return(dnorm(mu, mean = 250, sd = 25))
}

# Given data
y <- c(300, 270, 390, 450, 500, 290, 680, 450)

# Values of μ to evaluate
mu_values <- seq(0, 700, by = 5)  # Adjust the sequence for better resolution if needed

# Standard deviation σ
sigma <- 50

# Calculate unnormalized posterior density for each value of μ
posterior_density <- sapply(mu_values, function(mu) {
  likelihood(mu, sigma, y) * prior(mu)
})

# Plot the unnormalized posterior distribution
plot(mu_values, posterior_density, type = "l", col = "blue",
     xlab = expression(mu), ylab = "Unnormalized Posterior Density",
     main = "Unnormalized Posterior Distribution of mu")
```



Unnormalized Posterior Distribution of mu
Unnormalized Posterior Density
0e+00
2e−37
4e−37
0
100
200
300
400
500
600
700
μ

2.3

```r
# Prior distribution for μ
prior <- function(mu) {
  return(dnorm(mu, mean = 250, sd = 25))
}

# Likelihood function
likelihood <- function(mu, sigma, y) {
  n <- length(y)
  return((1 / (sigma * sqrt(2 * pi)))^n * exp(-(1 / (2 * sigma^2)) * sum((y - mu)^2)))
}

# Given data
y <- c(300, 270, 390, 450, 500, 290, 680, 450)

# Values of μ to evaluate
mu_values <- seq(0, 700, by = 5)  # Adjust the sequence for better resolution if needed

# Standard deviation σ
sigma <- 50

# Calculate prior density for each value of μ
prior_density <- prior(mu_values)

# Calculate unnormalized posterior density for each value of μ
posterior_density <- sapply(mu_values, function(mu) {
  likelihood(mu, sigma, y) * prior(mu)
})

# Plot the prior and unnormalized posterior distributions
plot(mu_values, prior_density, type = "l", col = "red", lwd = 2,
     xlab = expression(mu), ylab = "Density",
     main = "Comparison of Prior and Unnormalized Posterior Distribution")
lines(mu_values, posterior_density, type = "l", col = "blue", lwd = 2)
legend("topright", legend = c("Prior", "Unnormalized Posterior"),
       col = c("red", "blue"), lwd = 2, cex = 0.8)
```



Comparison of Prior and Unnormalized Posterior Distribution
Prior
Unnormalized Posterior
Density
0.000
0.005
0.010
0.015
0
100
200
300
400
500
600
700
μ

Part-3 : The Baysian learning.

3.1 Given likelihood $L(d|k) = \frac{d^k e^{-d}}{d!}$

Priors on $d \sim Gamma(40, 2)$

$\Rightarrow$ Posterior after day 1

$d/k_1 \sim Gamma(40 + k_1, 2+1)$

As given posterior distribution of d after day 1 will become prior for predicating day 2.

$\Rightarrow$ Prior on d to generate predication on day 5 will be same as posterior distribution of d after day 4 data.

$\Rightarrow$ Prior for day 5 -

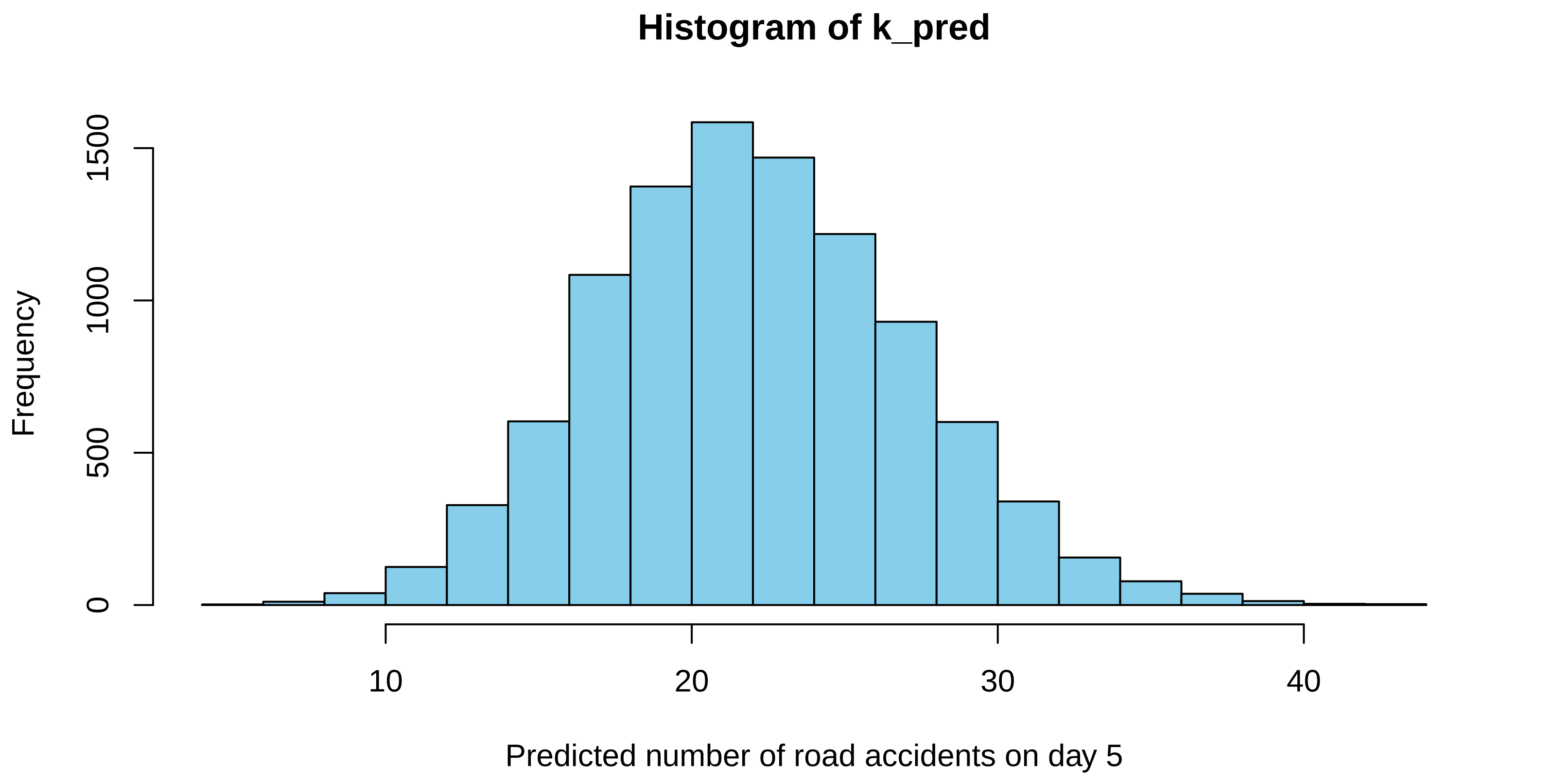
$d \sim Gamma(40 + k_1 + k_2 + k_3 + k_4, 6)$

$k_1 = 25, \; k_2 = 20, \; k_3 = 23, \; k_4 = 27$

$\boxed{d \sim Gamma(135, 6)}$ Ans.

3.2

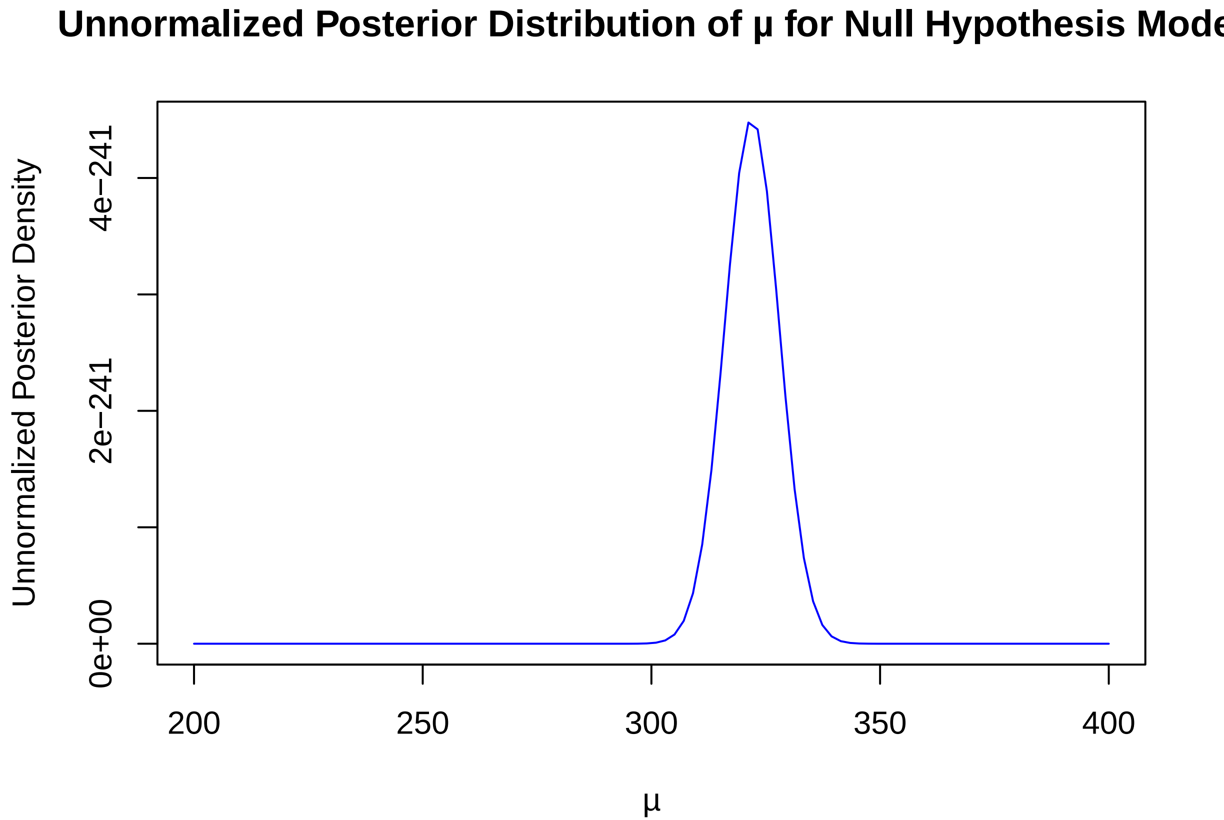
```
lambda <- rgamma(10000,135,6)
k_pred <- rep(NA,10000)
for(i in 1:10000){
  k_pred[i] <- rpois(1,lambda[i])
}

hist(k_pred,xlab = "Predicted number of road accidents on day 5",col = "skyblue")
```



Histogram of k_pred
Frequency
0
500
1000
1500
10
20
30
40
Predicted number of road accidents on day 5

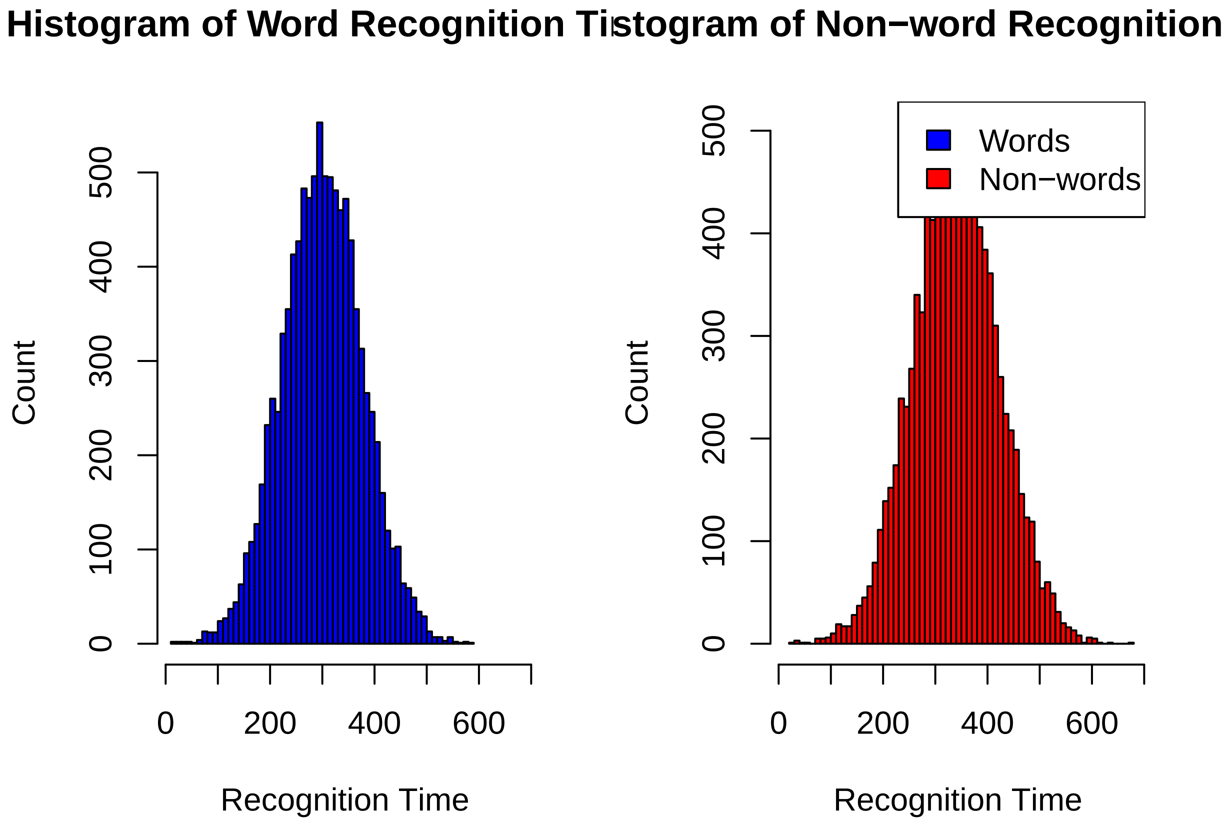
4.1

```
library(truncnorm)

# Load the data
dat <-
read.table("https://raw.githubusercontent.com/yadavhimanshu059/CGS698C/main/notes/Module-
2/recognition.csv", sep=",", header=T)[,-1]
head(dat)

# Extract Tw and Tnw vectors
Tw <- dat$Tw
Tnw <- dat$Tnw

# Define constants
sigma <- 60
mu_prior_mean <- 300
mu_prior_sd <- 50

# Define a sequence of μ values to evaluate
mu_values <- seq(200, 400, length.out = 100)


prior_mu <- function(mu) {
  dnorm(mu, mean = mu_prior_mean, sd = mu_prior_sd)
}

# Define the likelihood function for Tw and Tnw
likelihood <- function(mu, delta) {
  prod(dnorm(Tw, mean = mu, sd = sigma)) * prod(dnorm(Tnw, mean = mu + delta, sd = sigma))
}

# Define the unnormalized posterior function for the Null hypothesis model
unnormalized_posterior <- function(mu) {
  delta <- 0
  likelihood(mu, delta) * prior_mu(mu)
}


posterior_values <- sapply(mu_values, unnormalized_posterior)

# Plot the unnormalized posterior distribution of μ
plot(mu_values, posterior_values, type = "l", col = "blue",
     main = "Unnormalized Posterior Distribution of μ for Null Hypothesis Model",
     xlab = "μ", ylab = "Unnormalized Posterior Density")
```


Unnormalized Posterior Distribution of μ for Null Hypothesis Mode
Unnormalized Posterior Density
0e+00
2e−241
4e−241
200
250
300
350
400
μ

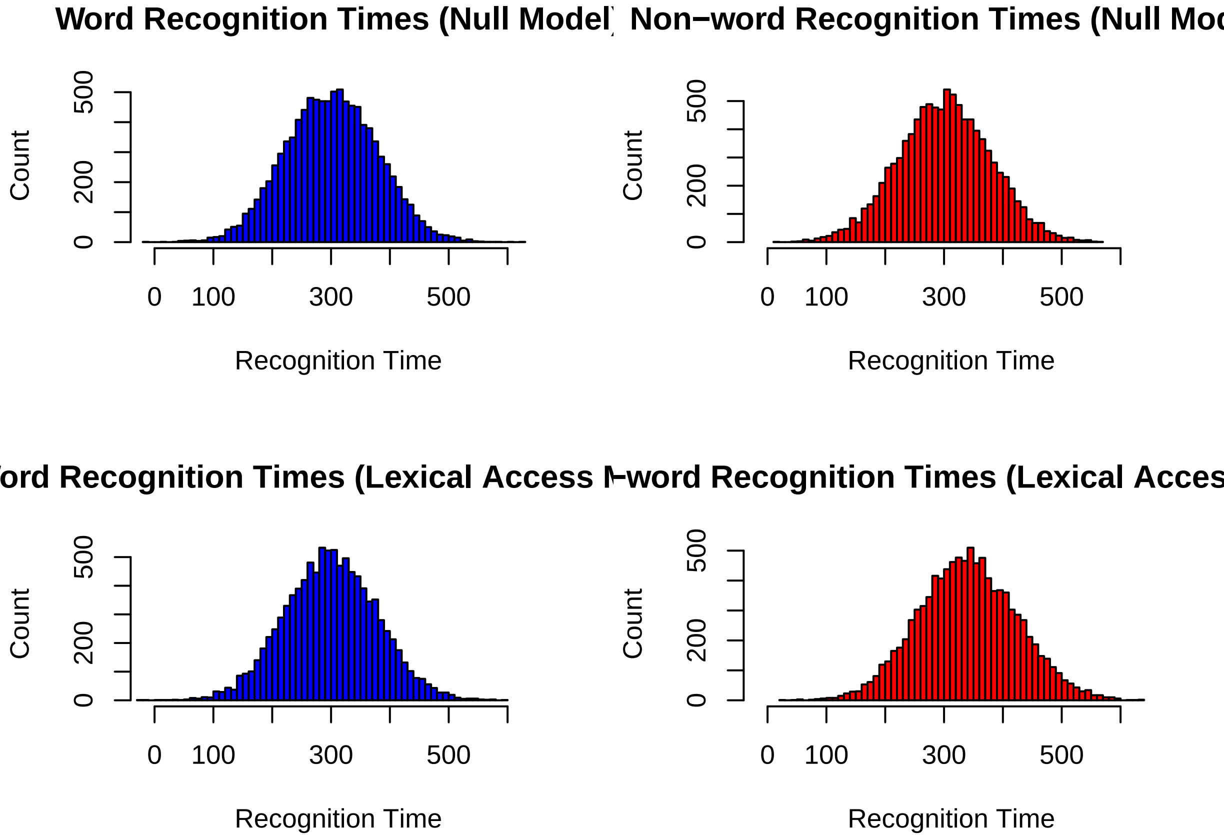
4.2

```
library(truncnorm)

# Define constants
mu_prior_mean <- 300
mu_prior_sd <- 50
sigma <- 60
delta_prior_mean <- 0
delta_prior_sd <- 50
n_samples <- 10000

# Step 1: Draw samples for μ from its prior distribution N(300, 50)
mu_samples <- rnorm(n_samples, mean = mu_prior_mean, sd = mu_prior_sd)

# Step 2: Draw samples for δ from its truncated normal prior N+(0, 50)
delta_samples <- rtruncnorm(n_samples, a = 0, b = Inf, mean = delta_prior_mean, sd =
delta_prior_sd)

# Step 3: Generate word and non-word recognition times
Tw_samples <- rnorm(n_samples, mean = mu_samples, sd = sigma)
Tnw_samples <- rnorm(n_samples, mean = mu_samples + delta_samples, sd = sigma)

# Step 4: Plot the histograms of the recognition times
par(mfrow = c(1, 2))  # Set up the plotting area to have 1 row and 2 columns

# Histogram of word recognition times
hist(Tw_samples, breaks = 50, col = "blue", main = "Histogram of Word Recognition Times",
     xlab = "Recognition Time", ylab = "Count", xlim = c(min(Tw_samples, Tnw_samples),
max(Tw_samples, Tnw_samples)))

# Histogram of non-word recognition times
hist(Tnw_samples, breaks = 50, col = "red", main = "Histogram of Non-word Recognition
Times",
     xlab = "Recognition Time", ylab = "Count", xlim = c(min(Tw_samples, Tnw_samples),
max(Tw_samples, Tnw_samples)))

# Add legends
legend("topright", legend = c("Words", "Non-words"), fill = c("blue", "red"))
```


Histogram of Word Recognition Ti
Histogram of Non−word Recognition
Count
Recognition Time
0
100
200
300
400
500
0
200
400
600
Words
Non−words

4.3

```
# Load necessary libraries
library(truncnorm)

# Define constants
mu_prior_mean <- 300
mu_prior_sd <- 50
sigma <- 60
delta_prior_mean <- 0
delta_prior_sd <- 50
n_samples <- 10000

# Null Hypothesis Model
# Step 1: Draw samples for μ from its prior distribution N(300, 50)
mu_samples_null <- rnorm(n_samples, mean = mu_prior_mean, sd = mu_prior_sd)

# Step 2: Generate word and non-word recognition times with δ = 0
Tw_samples_null <- rnorm(n_samples, mean = mu_samples_null, sd = sigma)
Tnw_samples_null <- rnorm(n_samples, mean = mu_samples_null, sd = sigma)

# Lexical Access Model
# Step 1: Draw samples for μ from its prior distribution N(300, 50)
mu_samples_lexical <- rnorm(n_samples, mean = mu_prior_mean, sd = mu_prior_sd)

# Step 2: Draw samples for δ from its truncated normal prior N+(0, 50)
delta_samples_lexical <- rtruncnorm(n_samples, a = 0, b = Inf, mean = delta_prior_mean, sd
= delta_prior_sd)

# Step 3: Generate word and non-word recognition times
Tw_samples_lexical <- rnorm(n_samples, mean = mu_samples_lexical, sd = sigma)
Tnw_samples_lexical <- rnorm(n_samples, mean = mu_samples_lexical + delta_samples_lexical,
sd = sigma)

# Step 4: Plot the histograms for comparison
par(mfrow = c(2, 2))  # Set up the plotting area to have 2 rows and 2 columns

# Histogram of word recognition times (Null Hypothesis Model)
hist(Tw_samples_null, breaks = 50, col = "blue", main = "Word Recognition Times (Null
Model)",
     xlab = "Recognition Time", ylab = "Count", xlim = c(min(Tw_samples_null,
Tnw_samples_lexical), max(Tw_samples_null, Tnw_samples_lexical)))

# Histogram of non-word recognition times (Null Hypothesis Model)
hist(Tnw_samples_null, breaks = 50, col = "red", main = "Non-word Recognition Times (Null
Model)",
     xlab = "Recognition Time", ylab = "Count", xlim = c(min(Tw_samples_null,
Tnw_samples_lexical), max(Tw_samples_null, Tnw_samples_lexical)))

# Histogram of word recognition times (Lexical Access Model)
hist(Tw_samples_lexical, breaks = 50, col = "blue", main = "Word Recognition Times
(Lexical Access Model)",
     xlab = "Recognition Time", ylab = "Count", xlim = c(min(Tw_samples_null,
Tnw_samples_lexical), max(Tw_samples_null, Tnw_samples_lexical)))

# Histogram of non-word recognition times (Lexical Access Model)
hist(Tnw_samples_lexical, breaks = 50, col = "red", main = "Non-word Recognition Times
(Lexical Access Model)",
     xlab = "Recognition Time", ylab = "Count", xlim = c(min(Tw_samples_null,
Tnw_samples_lexical), max(Tw_samples_null, Tnw_samples_lexical)))
```

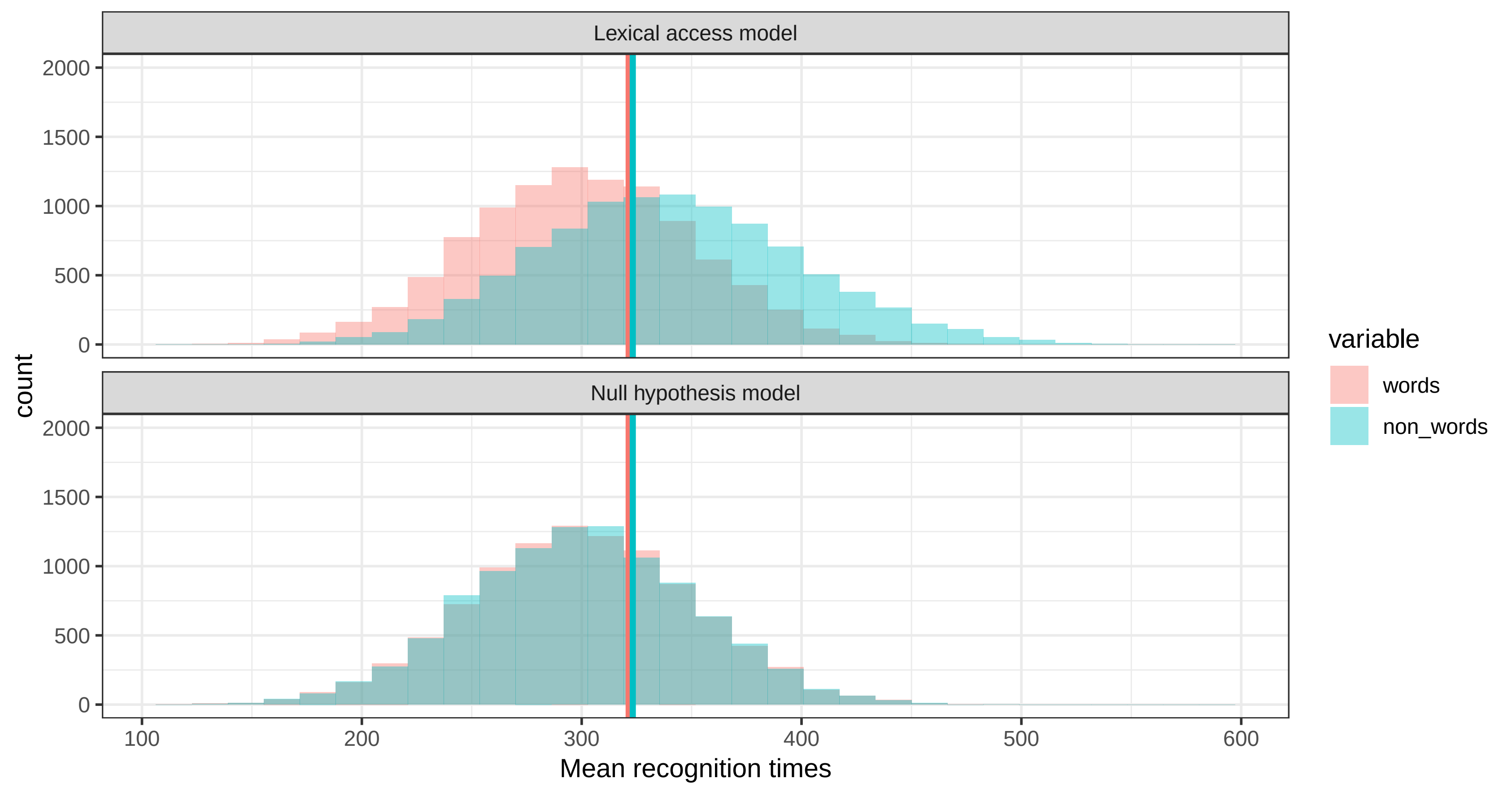

Word Recognition Times (Null Model)
Non−word Recognition Times (Null Mod
Count
0
200
500
0
100
300
500
Recognition Time
Count
0
200
500
0
100
300
500
Recognition Time
ord Recognition Times (Lexical Access M
−word Recognition Times (Lexical Acces
Count
0
200
500
0
100
300
500
Recognition Time
Count
0
200
500
0
100
300
500
Recognition Time

4.4

```
# Load necessary libraries
library(reshape2)
library(dplyr)
library(ggplot2)

# Read the data
dat <- read.table(
  "https://raw.githubusercontent.com/yadavhimanshu059/CGS698C/main/notes/Module-
2/recognition.csv",
  sep=",", header=T)[,-1]
head(dat)

# Generate prior predictions for the null hypothesis model
mu <- rnorm(10000, 300, 50)
sigma <- rep(60, 10000)
delta <- rep(0, 10000)

n <- length(dat$Tw)
df.pred <- data.frame(matrix(nrow=0, ncol=6))
colnames(df.pred) <- c("sample_id", "mu", "delta", "obs_id", "Tw_pred", "Tnw_pred")

for(i in 1:10000){
  Tw_pred <- rnorm(n, mean=mu[i], sd=sigma[i])
  Tnw_pred <- rnorm(n, mu[i] + delta[i], sd=sigma[i])
  df.pred <- rbind(df.pred, data.frame(sample_id=rep(i, n),
                                       mu=rep(mu[i], n),
                                       delta=rep(delta[i], n),
                                       obs_id=1:n,
                                       Tw_pred=Tw_pred,
                                       Tnw_pred=Tnw_pred))
}

df.pred.mean <- df.pred %>%
  group_by(sample_id) %>%
  summarize(words=mean(Tw_pred), non_words=mean(Tnw_pred))
df.pred.mean$model <- "Null hypothesis model"
df.pred.null.hypothesis <- melt(df.pred.mean, id=c("model", "sample_id"))

# Generate prior predictions for the lexical-access model
delta_lexical <- rtruncnorm(10000, a=0, b=Inf, mean=0, sd=50)

df.pred.lexical.access <- data.frame(matrix(nrow=0, ncol=6))
colnames(df.pred.lexical.access) <- c("sample_id", "mu", "delta", "obs_id", "Tw_pred",
"Tnw_pred")

for(i in 1:10000){
  Tw_pred <- rnorm(n, mean=mu[i], sd=sigma[i])
  Tnw_pred <- rnorm(n, mean=mu[i] + delta_lexical[i], sd=sigma[i])
  df.pred.lexical.access <- rbind(df.pred.lexical.access, data.frame(sample_id=rep(i, n),
                                                                     mu=rep(mu[i], n),

delta=rep(delta_lexical[i], n),
                                                                     obs_id=1:n,
                                                                     Tw_pred=Tw_pred,
                                                                     Tnw_pred=Tnw_pred))
}

df.pred.mean.lexical <- df.pred.lexical.access %>%
  group_by(sample_id) %>%
  summarize(words=mean(Tw_pred), non_words=mean(Tnw_pred))
df.pred.mean.lexical$model <- "Lexical access model"
df.pred.lexical.access <- melt(df.pred.mean.lexical, id=c("model", "sample_id"))

# Combine the data frames
```

```
df.pred <- rbind(df.pred.lexical.access, df.pred.null.hypothesis)

# Plot the data
ggplot(df.pred, aes(x=value, group=variable, fill=variable)) +
  geom_histogram(alpha=0.4, bins=30, position="identity") +
  xlab("Mean recognition times") + theme_bw() +
  facet_wrap(~model, nrow=2) +
  geom_vline(xintercept=mean(dat$Tw), color="#F8766D", size=1.2) +
  geom_vline(xintercept=mean(dat$Tnw), color="#00BFC4", size=1.2) +
  ylim(0, 2500)
```

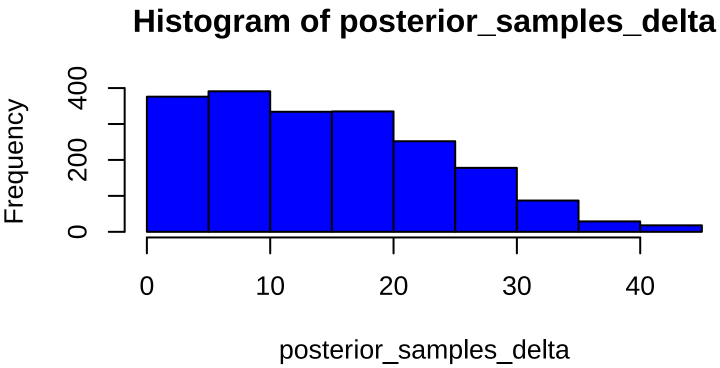
The predictions of the null hypothesis model seem more consistent with the data.



Lexical access model
Null hypothesis model
count
2000
1500
1000
500
0
100
200
300
400
500
600
Mean recognition times
variable
words
non_words

4.5

```
# Take equidistant values of mu
mu <- runif(50000,100,500)
sigma <- 60
delta <- runif(50000,0,50)
# Compute likelihood, prior, posterior for each of them
likelihood <- rep(NA,50000)
prior <- rep(NA,50000)
posterior_unnorm <- rep(NA,50000)

for(i in 1:50000){
  likelihood[i] <- prod(dnorm(dat$Tw,mu[i],sigma))*
    prod(dnorm(dat$Tnw,mu[i]+delta[i],sigma))
  prior[i] <- dnorm(mu[i],300,50)*
    dtruncnorm(delta[i],a=0,b=Inf,mean=0,sd=50)
  posterior_unnorm[i] <- likelihood[i]*prior[i]

}
posterior_samples_delta <- sample(delta,size=2000,prob = posterior_unnorm)
hist(posterior_samples_delta,col = "blue")
```


Histogram of posterior_samples_delta
Frequency
0
200
400
0
10
20
30
40
posterior_samples_delta