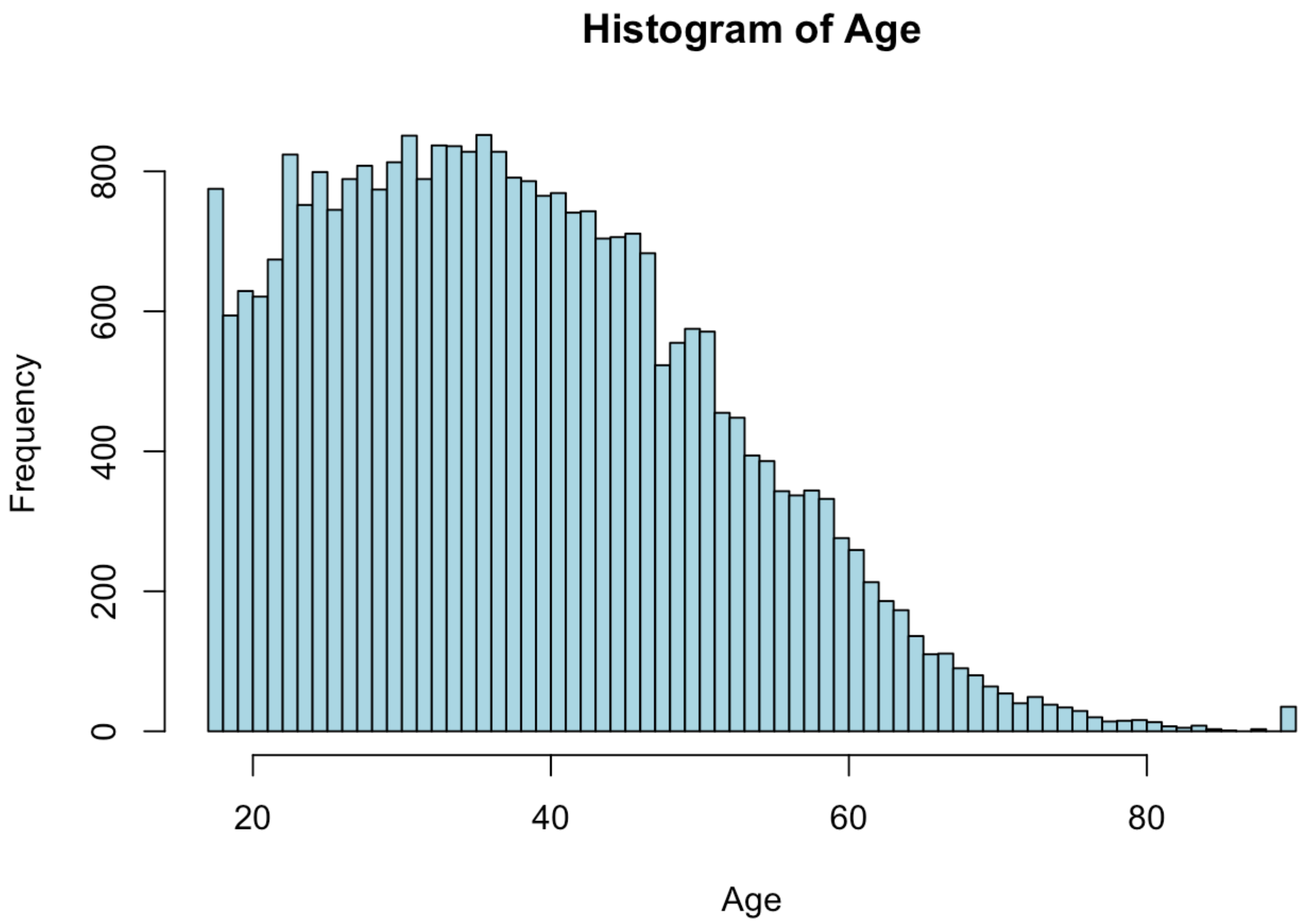
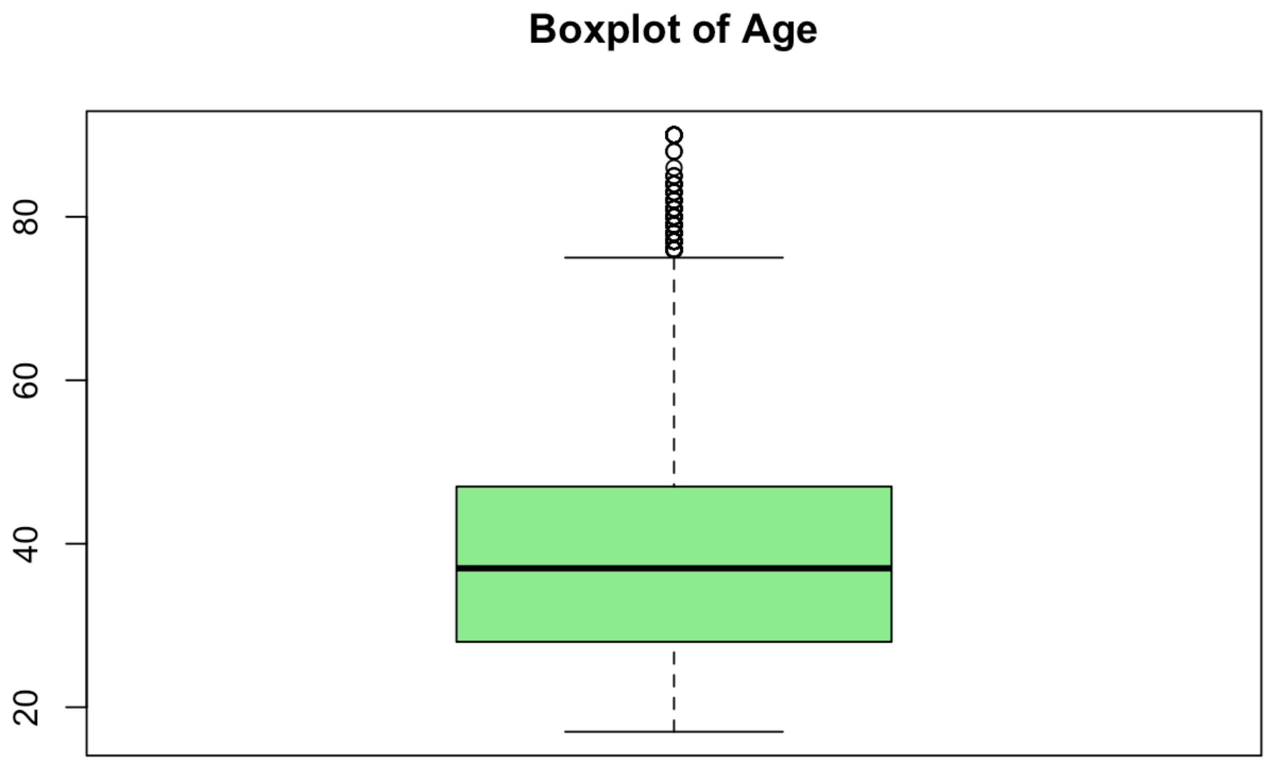
```
# QUESTION 2


library(ggplot2)

# Load the dataset from the CSV file
data <- read.csv("/Users/prabuddhadurge/Downloads/adult_data.csv")

X <- data$occupation.num                                # Discrete variable
Y <- data$age                                           # Continuous variable

# (2a) Frequency distribution for discrete variable X (e.g., occupation.num)
frequency_distribution <- table(X)
print(frequency_distribution)
```

```
## X
##    1    2    3    4    5    6    7    8    9   10   11   12   13   14
## 1350 3721 3212  989 3584 1572 1966 4030  912  644  143    9 3992 4038
```

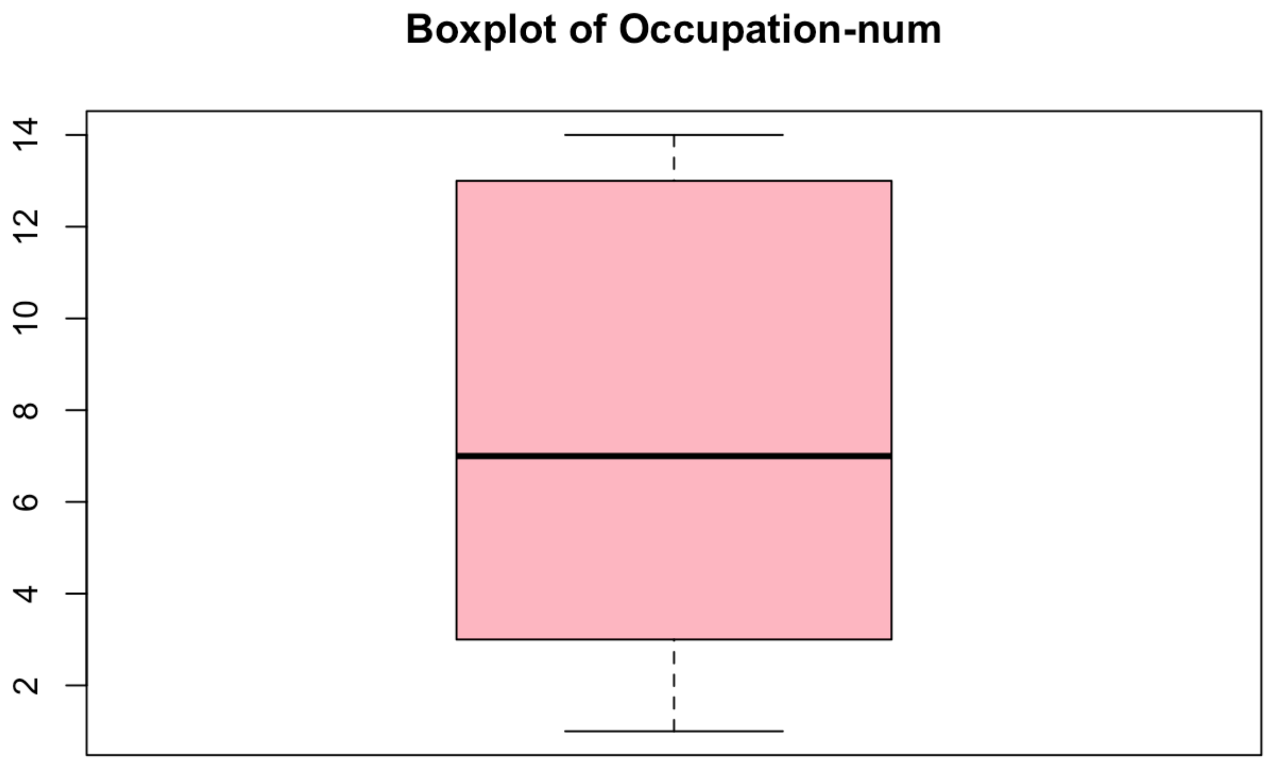
```
# (2b) Plot the histogram for continuous variable Y

# Calculate bin width using Freedman-Diaconis rule
iqr <- IQR(Y)                                           # Interquartile range
bin_width <- 2 * iqr / (length(Y)^(1/3))                # Freedman-Diaconis rule
num_bins <- ceiling((max(Y) - min(Y)) / bin_width)  # Number of bins based on bin
width

# Plot the histogram
hist(Y, breaks = num_bins, col = "lightblue",
     main = "Histogram of Age",
     xlab = "Age",
     ylab = "Frequency")
```



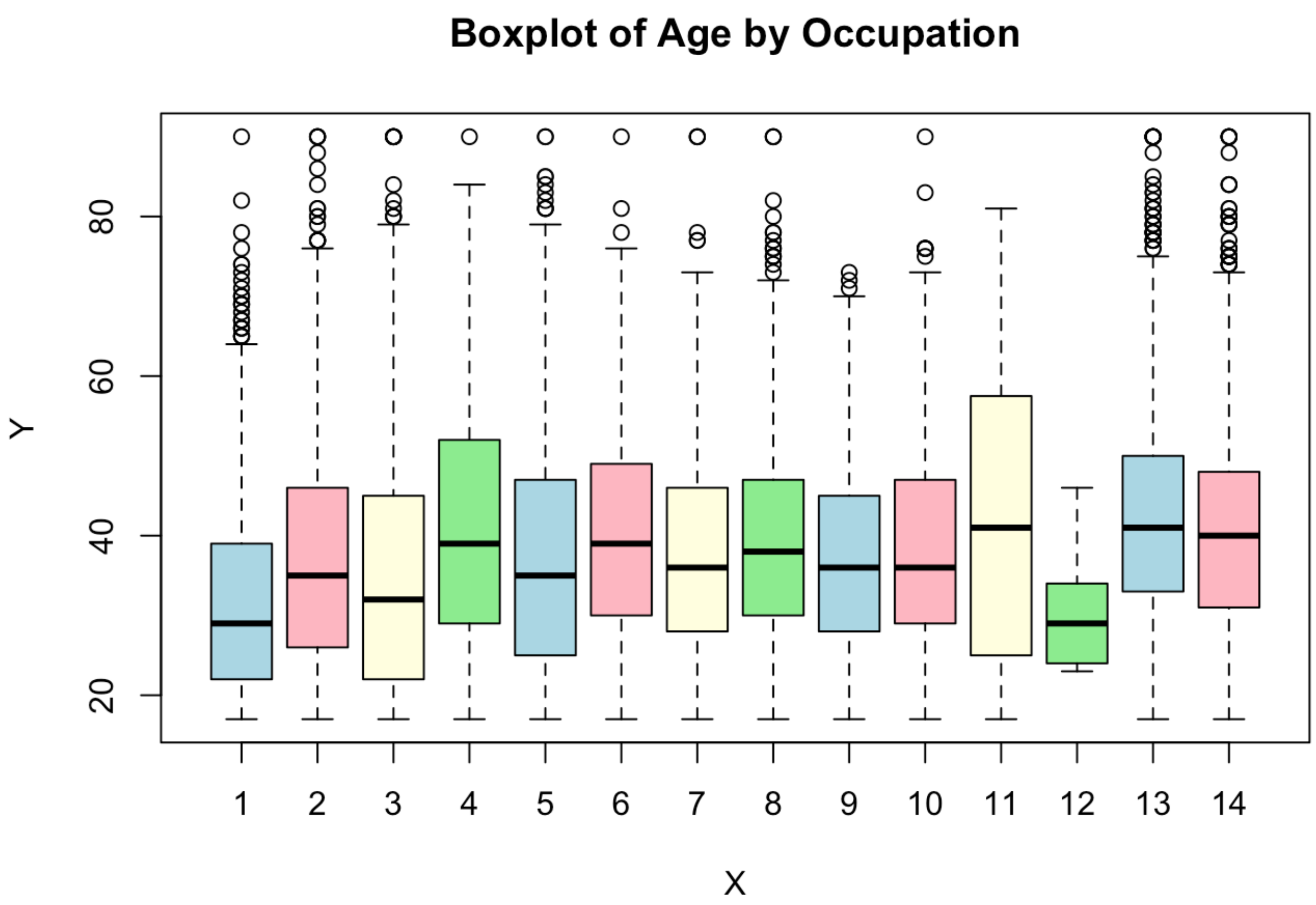

```
# (2c) Box-and-Whisker Plot for X and Y
par(mfrow = c(1, 1))

# Boxplot for continuous variable Y
boxplot(Y, main = "Boxplot of Age", col = "lightgreen")
```




```
#Boxplot for continuous variable X
boxplot(X, main = "Boxplot of Occupation-num", col = "lightpink")
```

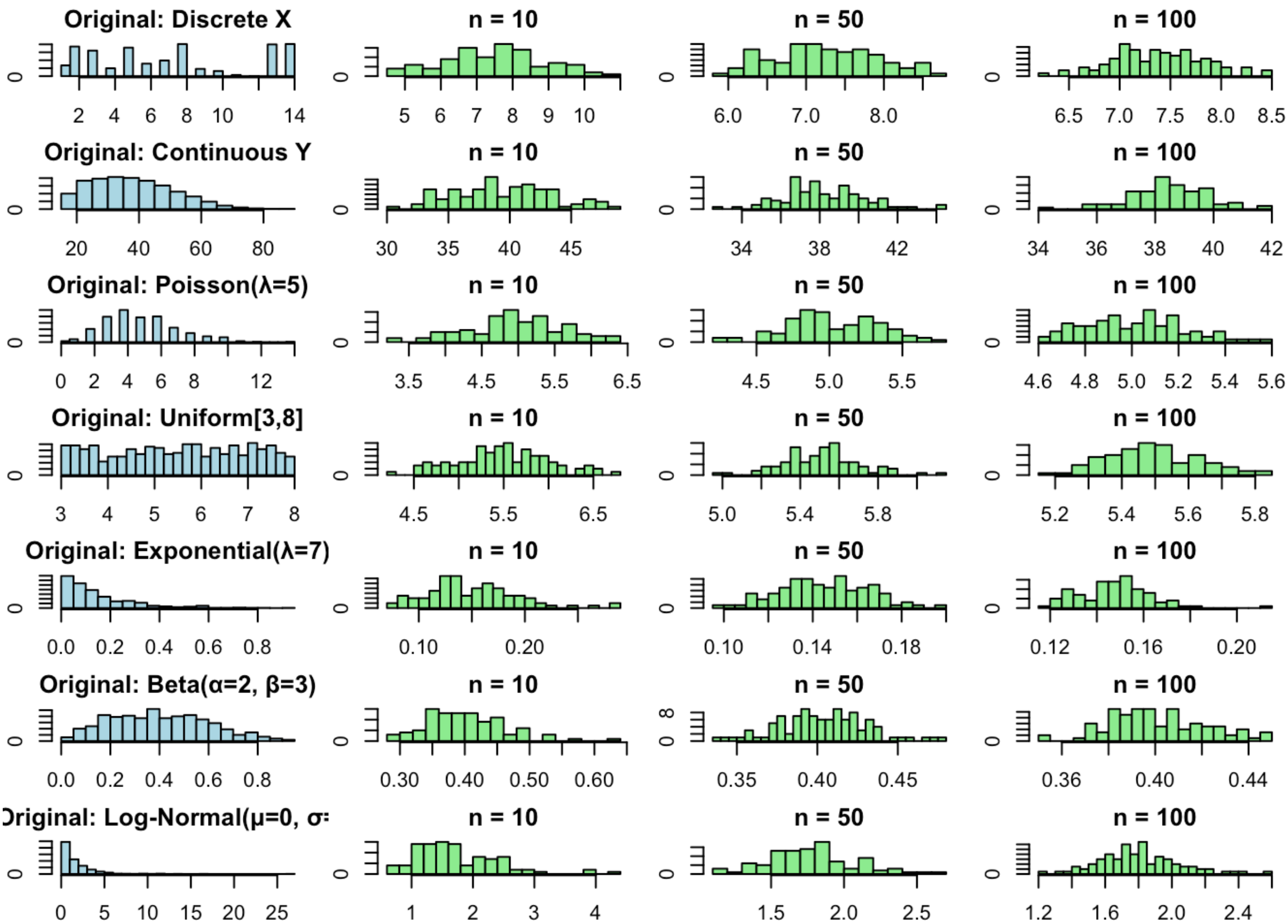



```
# Boxplot for continuous variable Y by discrete variable X
boxplot(Y~X, data = data, main = "Boxplot of Age by Occupation",
        col = c("lightblue", "lightpink", "lightyellow", "lightgreen"))
```




```
# QUESTION 3

# (3abc) Generate the 7 datasets
set.seed(123)  # For reproducibility

# Simulated datasets
poisson_data <- rpois(500, lambda = 5)                     # Poisson distribution
uniform_data <- runif(500, min = 3, max = 8)               # Continuous uniform distr
ibution
exponential_data <- rexp(500, rate = 7)                    # Exponential distribution
beta_data <- rbeta(500, shape1 = 2, shape2 = 3)            # Beta distribution
log_normal_data <- rlnorm(500, meanlog = 0, sdlog = 1)  # Log-normal distribution

datasets <- list(
  "Discrete X" = X,
  "Continuous Y" = Y,
  "Poisson(λ=5)" = poisson_data,
  "Uniform[3,8]" = uniform_data,
  "Exponential(λ=7)" = exponential_data,
  "Beta(α=2, β=3)" = beta_data,
  "Log-Normal(μ=0, σ=1)" = log_normal_data
)
```

```
# Draw 100 samples and compute means for n = 10, 50, 100
compute_sample_means <- function(data, n) {
  replicate(100, mean(sample(data, size = n, replace = TRUE)))
}

# Compute histograms
plot_histograms <- function(datasets, sample_sizes) {
  par(mfrow = c(7, 4), mar = c(2, 2, 2, 1))            # 7 rows, 4 columns

  for (i in 1:length(datasets)) {
    data <- datasets[[i]]

    # Plot histogram of the original dataset
    hist(data, main = paste("Original:", names(datasets)[i]), col = "lightblue",
         xlab = "Values", ylab = "Frequency", breaks = 20)

    # Compute and plot histograms for sample means
    for (n in sample_sizes) {
      sample_means <- compute_sample_means(data, n)
      hist(sample_means, main = paste("n =", n), col = "lightgreen",
           xlab = "Sample Mean", ylab = "Frequency", breaks = 20)
    }
  }
}

# Run the plotting function for sample sizes n = 10, 50, 100
plot_histograms(datasets, sample_sizes = c(10, 50, 100))
```




```
# QUESTION 4

# Load required libraries
library(missMethods)  # For introducing missing data
library(mice)         # For imputation
```

```
##
## Attaching package: 'mice'
```

```
## The following object is masked from 'package:stats':
##
##     filter
```

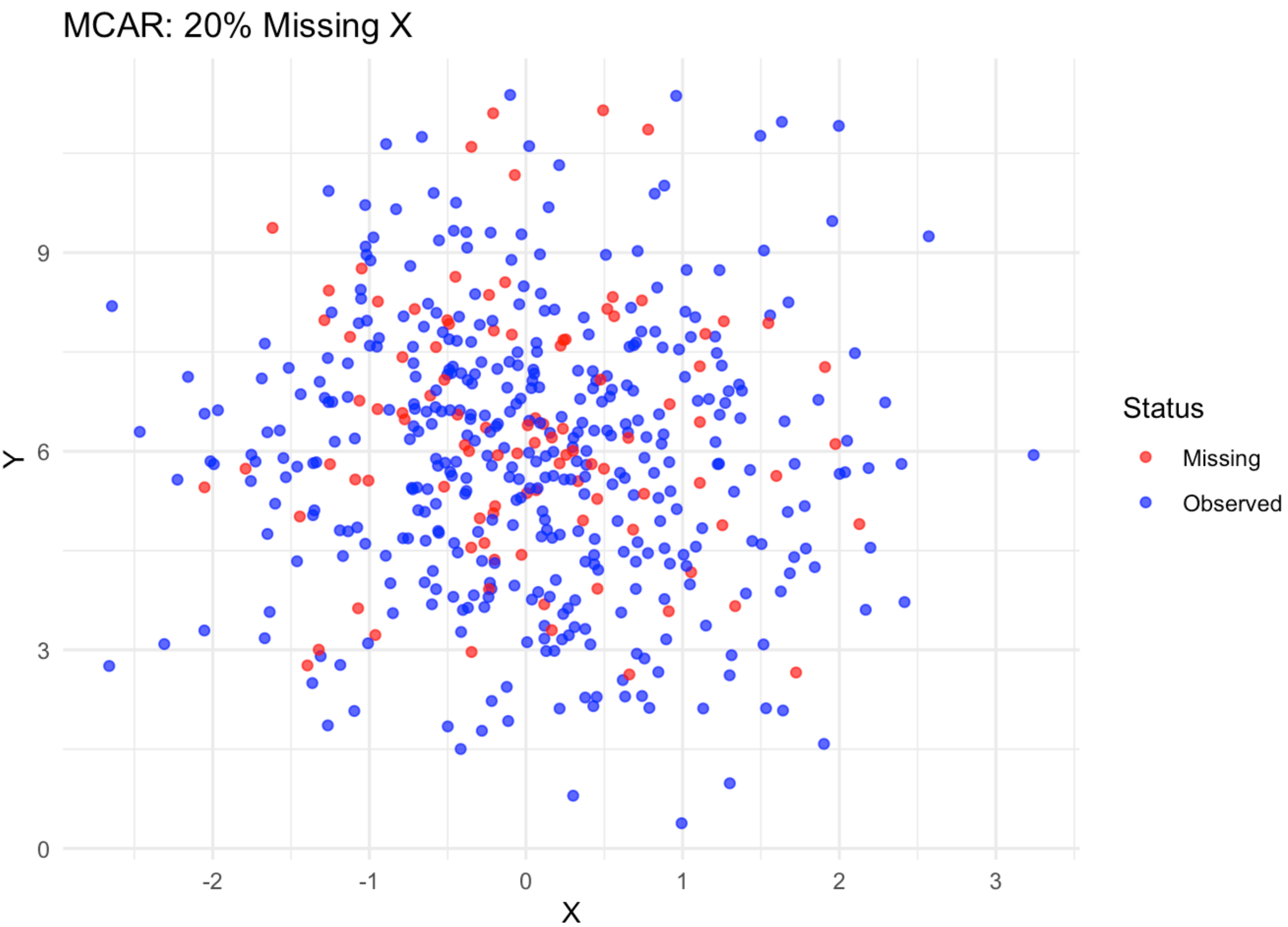
```
## The following objects are masked from 'package:base':
##
##     cbind, rbind
```

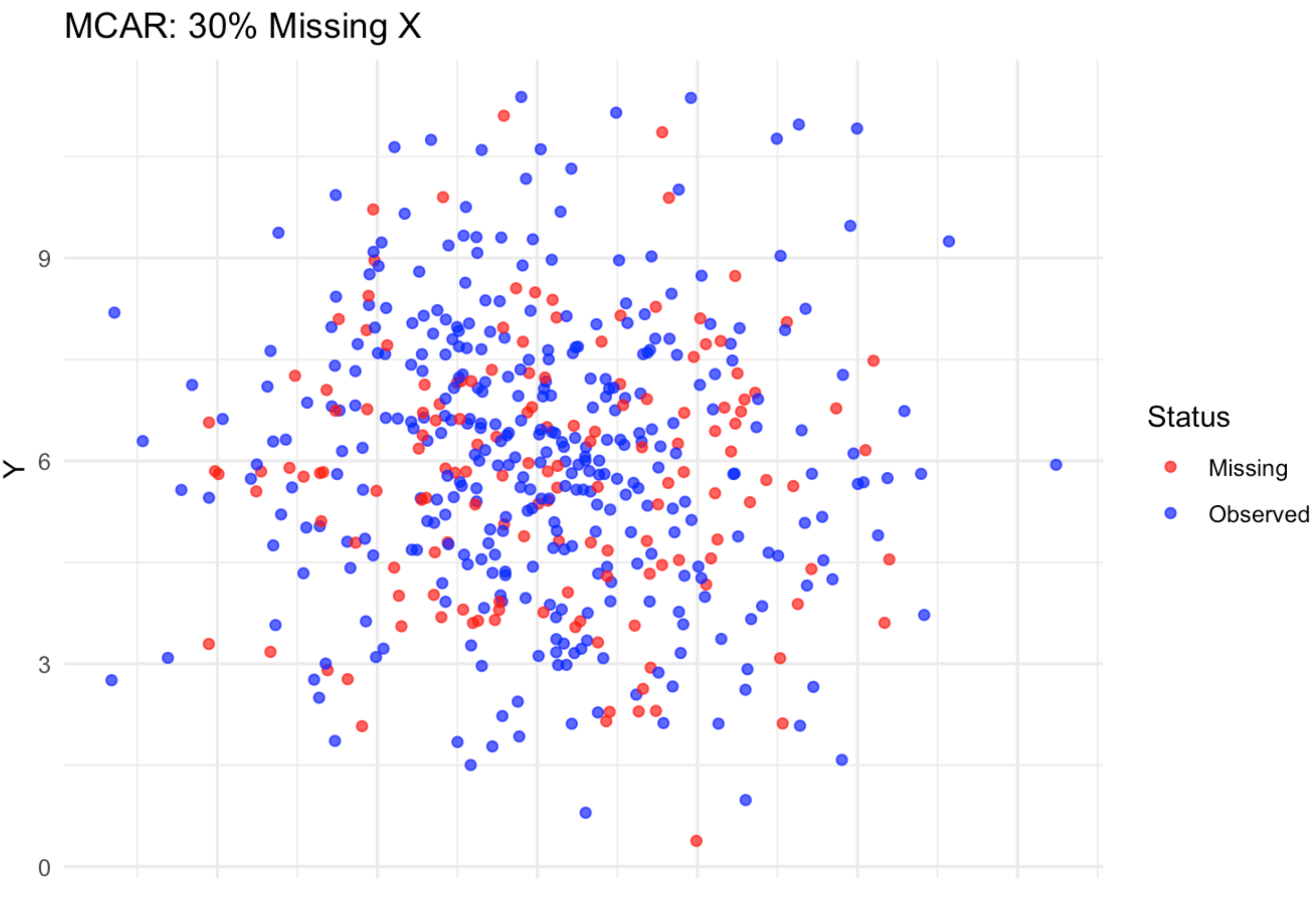
```
library(ggplot2)      # For plotting

# Set seed for reproducibility
```

```
set.seed(123)

# Generate bivariate dataset
n <- 500
X <- rnorm(n, mean = 0, sd = 1)  # X ~ N(0, 1)
Y <- rnorm(n, mean = 6, sd = 2)  # Y ~ N(6, 4)
data <- data.frame(X, Y)

# Function to introduce missing data mechanisms
delete_data <- function(data, p, mechanism, ctrl_col = NULL) {
  if (mechanism == "MCAR") {
    return(delete_MCAR(data, p = p, cols_mis = "X"))
  } else if (mechanism == "MAR") {
    return(delete_MAR_rank(data, p = p, cols_mis = "X", cols_ctrl = ctrl_col))
  } else if (mechanism == "NMAR") {
    return(delete_MNAR_censoring(data, p = p, cols_mis = "X"))
  }
}

# Function to plot missing data
plot_missing_data <- function(original, missing, title) {
  # Identify missing indices
  missing_indices <- is.na(missing$X)

  # Create a combined dataset
  data_combined <- data.frame(
    X = original$X,
    Y = original$Y,
    Status = ifelse(missing_indices, "Missing", "Observed")
  )

  # Assign colors
  colors <- c("blue", "red")
  names(colors) <- c("Observed", "Missing")

  # Plot using ggplot2
  ggplot(data_combined, aes(x = X, y = Y, color = Status)) +
    geom_point(alpha = 0.7) +
    scale_color_manual(values = colors) +
    labs(title = title, x = "X", y = "Y") +
    theme_minimal()
}

# Imputation methods
# Mean Imputation
impute_mean <- function(data) {
  data$X[is.na(data$X)] <- mean(data$X, na.rm = TRUE)
  return(data)
}
```

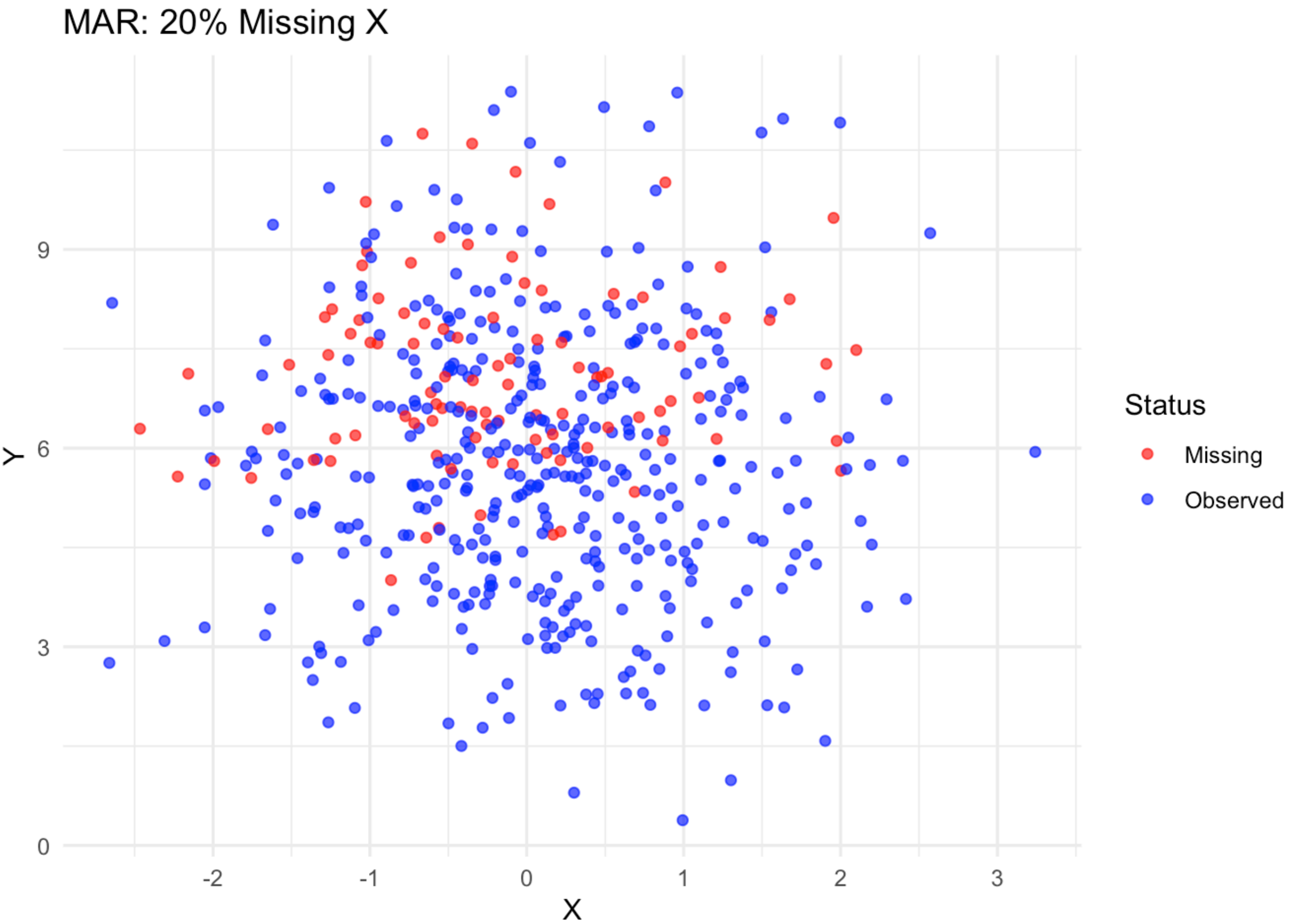
```
# PMM using mice
impute_pmm <- function(data) {
  imputed <- mice(data, method = "pmm", m = 1, maxit = 5, print = FALSE)
  return(complete(imputed))
}

# RMSE calculation
rmse <- function(original, imputed, missing_indices) {
  sqrt(mean((original[missing_indices] - imputed[missing_indices])^2, na.rm = TRU
E))
}

# Evaluate imputation methods
evaluate_imputation <- function(data_original, data_missing, method) {
  missing_indices <- is.na(data_missing$X)
  imputed_data <- method(data_missing)
  return(rmse(data_original$X, imputed_data$X, missing_indices))
}

# Initialize results dataframe
results <- data.frame(
  Mechanism = character(),
  Missing_Percentage = numeric(),
  RMSE_Mean = numeric(),
  RMSE_PMM = numeric()
)

# Missingness levels and mechanisms
missingness_levels <- c(0.2, 0.3)
mechanisms <- c("MCAR", "MAR", "NMAR")

# Loop over mechanisms and missingness levels
for (mechanism in mechanisms) {
  for (p in missingness_levels) {
    # Create missing data
    if (mechanism == "MAR") {
      data_missing <- delete_data(data, p, mechanism, ctrl_col = "Y")
    } else {
      data_missing <- delete_data(data, p, mechanism)
    }

    # Imputation and RMSE
    rmse_mean <- evaluate_imputation(data, data_missing, impute_mean)
    rmse_pmm <- evaluate_imputation(data, data_missing, impute_pmm)

    # Append to results
    results <- rbind(
      results,
      data.frame(
        Mechanism = mechanism,
```

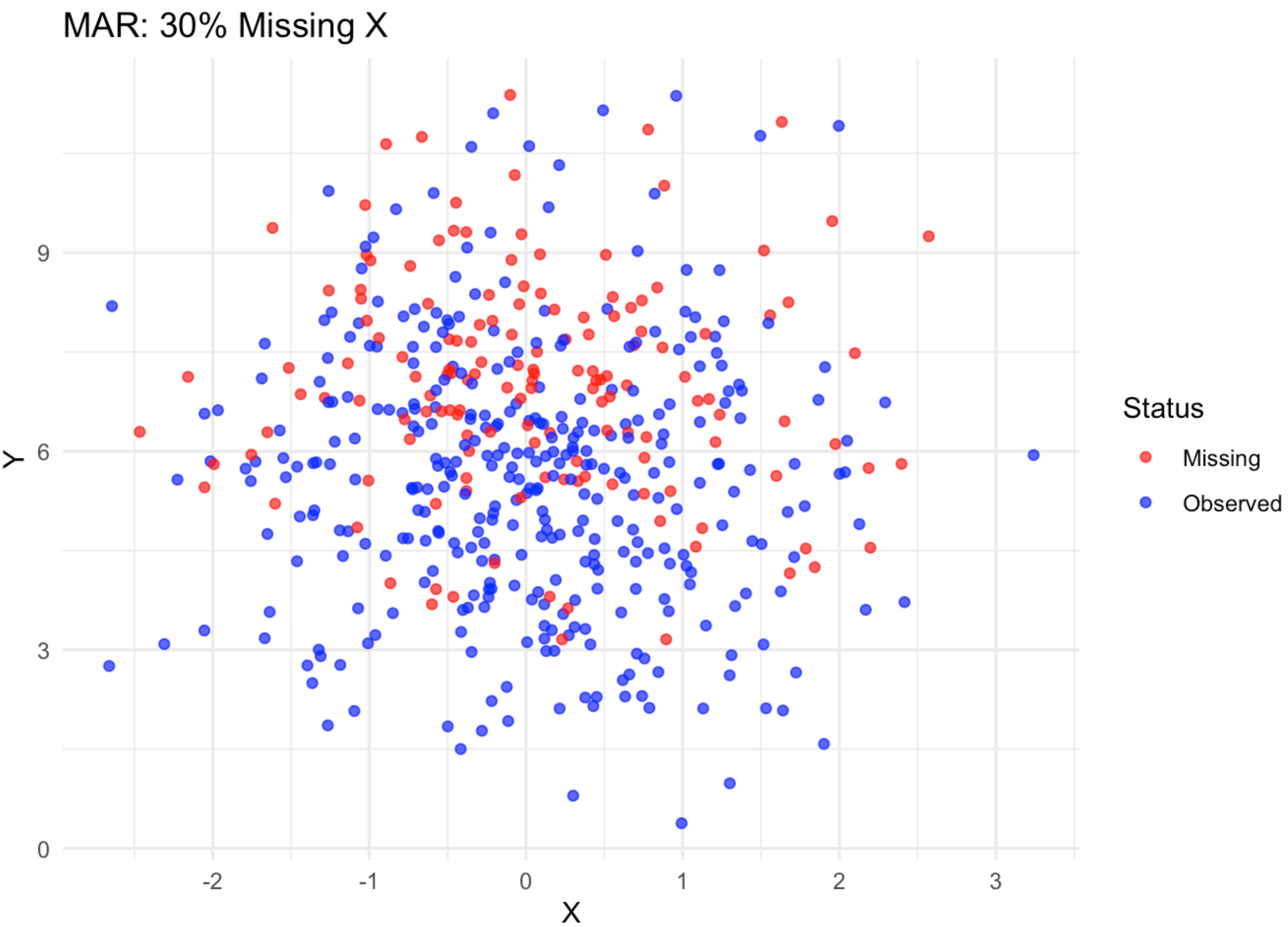
```
        Missing_Percentage = p * 100,
        RMSE_Mean = rmse_mean,
        RMSE_PMM = rmse_pmm
      )
    )

    # Plot missing data
    title <- paste(mechanism, ": ", p * 100, "% Missing X", sep = "")
    print(plot_missing_data(data, data_missing, title))
  }
}
```

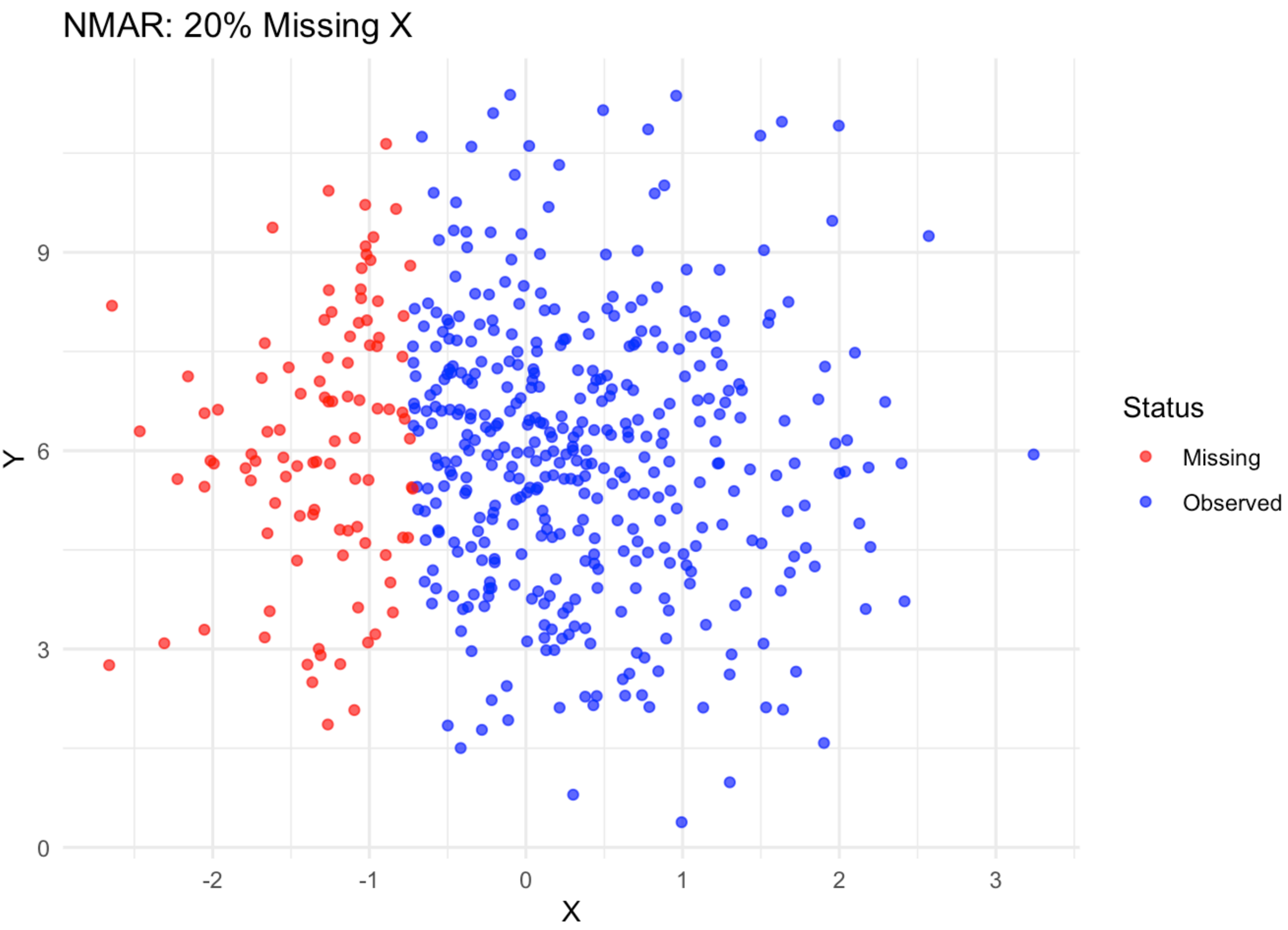

MCAR: 20% Missing X
Y
X
0
3
6
9
-2
-1
0
1
2
3
Status
Missing
Observed


MCAR: 30% Missing X
Y
0
3
6
9
Status
Missing
Observed


-2
-1
0
1
2
3
X


MAR: 20% Missing X
9
6
3
0
Y
-2
-1
0
1
2
3
X
Status
Missing
Observed

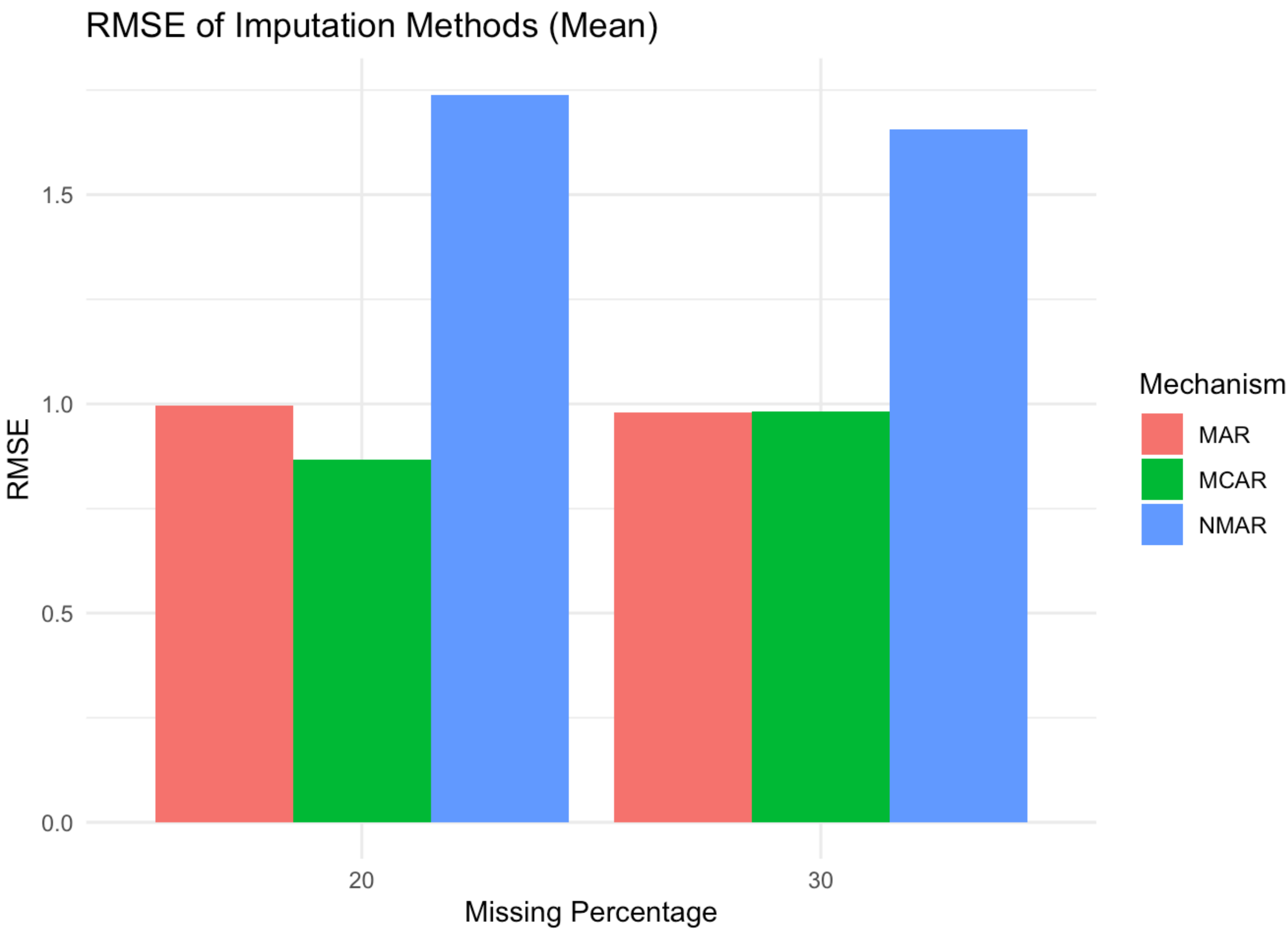

MAR: 30% Missing X
Y
X
9
6
3
0
-2
-1
0
1
2
3
Status
Missing
Observed


NMAR: 20% Missing X
Y
9
6
3
0
-2
-1
0
1
2
3
X
Status
Missing
Observed

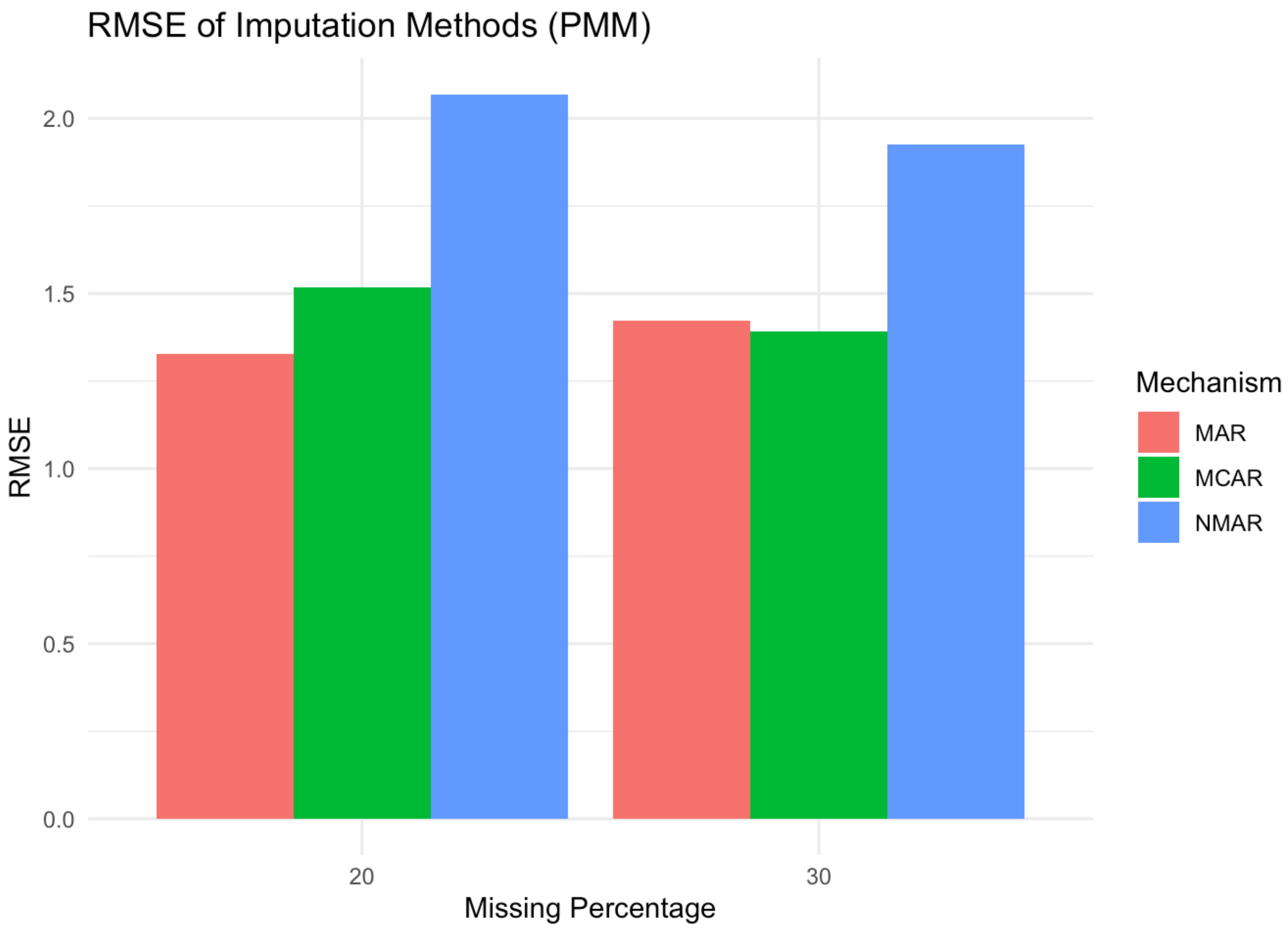

```
# Print results
print(results)
```

```
##   Mechanism Missing_Percentage RMSE_Mean RMSE_PMM
## 1      MCAR                 20 0.8656281 1.516610
## 2      MCAR                 30 0.9815377 1.392496
## 3       MAR                 20 0.9963683 1.328518
## 4       MAR                 30 0.9791792 1.422421
## 5      NMAR                 20 1.7385654 2.068372
## 6      NMAR                 30 1.6548717 1.924810
```

```
# Visualize RMSE results for Mean Imputation
ggplot(results, aes(x = factor(Missing_Percentage), y = RMSE_Mean, fill = Mechanis
m)) +
  geom_bar(stat = "identity", position = "dodge") +
  labs(title = "RMSE of Imputation Methods (Mean)", x = "Missing Percentage", y =
"RMSE") +
  theme_minimal()
```




```
# Visualize RMSE results for PMM
ggplot(results, aes(x = factor(Missing_Percentage), y = RMSE_PMM, fill = Mechanis
m)) +
  geom_bar(stat = "identity", position = "dodge") +
  labs(title = "RMSE of Imputation Methods (PMM)", x = "Missing Percentage", y = "
RMSE") +
  theme_minimal()
```




```
# QUESTION 5


library(fitdistrplus)
```

```
## Loading required package: MASS
```

```
## Loading required package: survival
```

```
data <- read.csv("/Users/prabuddhadurge/Downloads/adult_data.csv")
head(data)
```

```
##   age         workclass fnlwgt  education education.num      marital.status
## 1  39         State-gov  77516  Bachelors            13       Never-married
## 2  50  Self-emp-not-inc  83311  Bachelors            13  Married-civ-spouse
## 3  38           Private 215646    HS-grad             9            Divorced
## 4  53           Private 234721       11th             7  Married-civ-spouse
## 5  28           Private 338409  Bachelors            13  Married-civ-spouse
## 6  37           Private 284582    Masters            14  Married-civ-spouse
##           occupation   relationship   race     sex capital.gain capital.loss
## 1       Adm-clerical  Not-in-family  White    Male         2174            0
## 2    Exec-managerial        Husband  White    Male            0            0
## 3  Handlers-cleaners  Not-in-family  White    Male            0            0
## 4  Handlers-cleaners        Husband  Black    Male            0            0
## 5     Prof-specialty           Wife  Black  Female            0            0
## 6    Exec-managerial           Wife  White  Female            0            0
##   hours.per.week native.country income occupation.num
## 1             40  United-States  <=50K              2
## 2             13  United-States  <=50K             13
## 3             40  United-States  <=50K              1
## 4             40  United-States  <=50K              1
## 5             40           Cuba  <=50K             14
## 6             40  United-States  <=50K             13
```

```
A <- data$occupation.num   # Discrete variable
B <- data$education.num   # Discrete variable
C <- data$age              # Continuous variable
D <- data$hours.per.week   # Continuous variable

# A <- occupation.num (discrete), B <- education.num (discrete), C <- age (continu
ous), D <- hours.per.week (continuous)

# Step 1: Dataset A - Discrete Variable: occupation.num
A <- data$occupation.num

# (5a) Compute the median of the data.
a <- median(A, na.rm = TRUE)  # Compute the median of occupation.num
cat("Median of occupation.num (A):", a, "\n")
```

```
## Median of occupation.num (A): 7
```

```
# (5b) Select a simple random sample of size 100 from the dataset.
set.seed(123)  # For reproducibility
sample_A <- sample(A, size = 100, replace = TRUE)

# (5c) Count the number of observations in the sample that are less than 'a'.
Z <- sum(sample_A <= a)  # Count how many values are less than or equal to the med
ian
cat("Number of observations in the sample less than or equal to median:", Z, "\n")
```

```
## Number of observations in the sample less than or equal to median: 52
```

```
# (5d) Test the hypothesis that p = P[X ≤ a] is larger than 0.5 at the 5% signific
ance level.
p0 <- 0.5  # Null hypothesis: p = 0.5
p_hat <- Z / 100  # Proportion of observations less than or equal to median

# Perform the binomial test
test_result <- binom.test(Z, 100, p = p0, alternative = "greater", conf.level = 0.
95)
cat("Hypothesis test result for occupation.num:\n")
```

```
## Hypothesis test result for occupation.num:
```

```
print(test_result)
```

```
##
##  Exact binomial test
##
## data:  Z and 100
## number of successes = 52, number of trials = 100, p-value = 0.3822
## alternative hypothesis: true probability of success is greater than 0.5
## 95 percent confidence interval:
##  0.4332319 1.0000000
## sample estimates:
## probability of success
##                   0.52
```

```
# (5e) Provide an approximate 95% confidence interval for p.
prop_test <- prop.test(Z, 100, conf.level = 0.95)
cat("95% Confidence Interval for p (occupation.num):\n")
```

```
## 95% Confidence Interval for p (occupation.num):
```

```
print(prop_test$conf.int)
```

```
## [1] 0.4183183 0.6201278
## attr(,"conf.level")
## [1] 0.95
```

```
# QUESTION 6

B <- data$education.num
B <- sample(B, size = 3000)

# (a) Fit a normal distribution to dataset B
fit_b <- fitdist(B, "norm")
cat("Fitted Normal Distribution for education.num(B):\n")
```

```
## Fitted Normal Distribution for education.num(B):
```

```
print(fit_b)
```

```
## Fitting of the distribution ' norm ' by maximum likelihood
## Parameters:
##       estimate Std. Error
## mean 10.121333 0.04590066
## sd    2.514083 0.03245665
```

```
# (b) Conduct a Shapiro-Wilk test for normality
shapiro_test_b <- shapiro.test(B)
cat("Shapiro-Wilk Test for Normality education.num(B):\n")
```

```
## Shapiro-Wilk Test for Normality education.num(B):
```

```
print(shapiro_test_b)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  B
## W = 0.92561, p-value < 2.2e-16
```

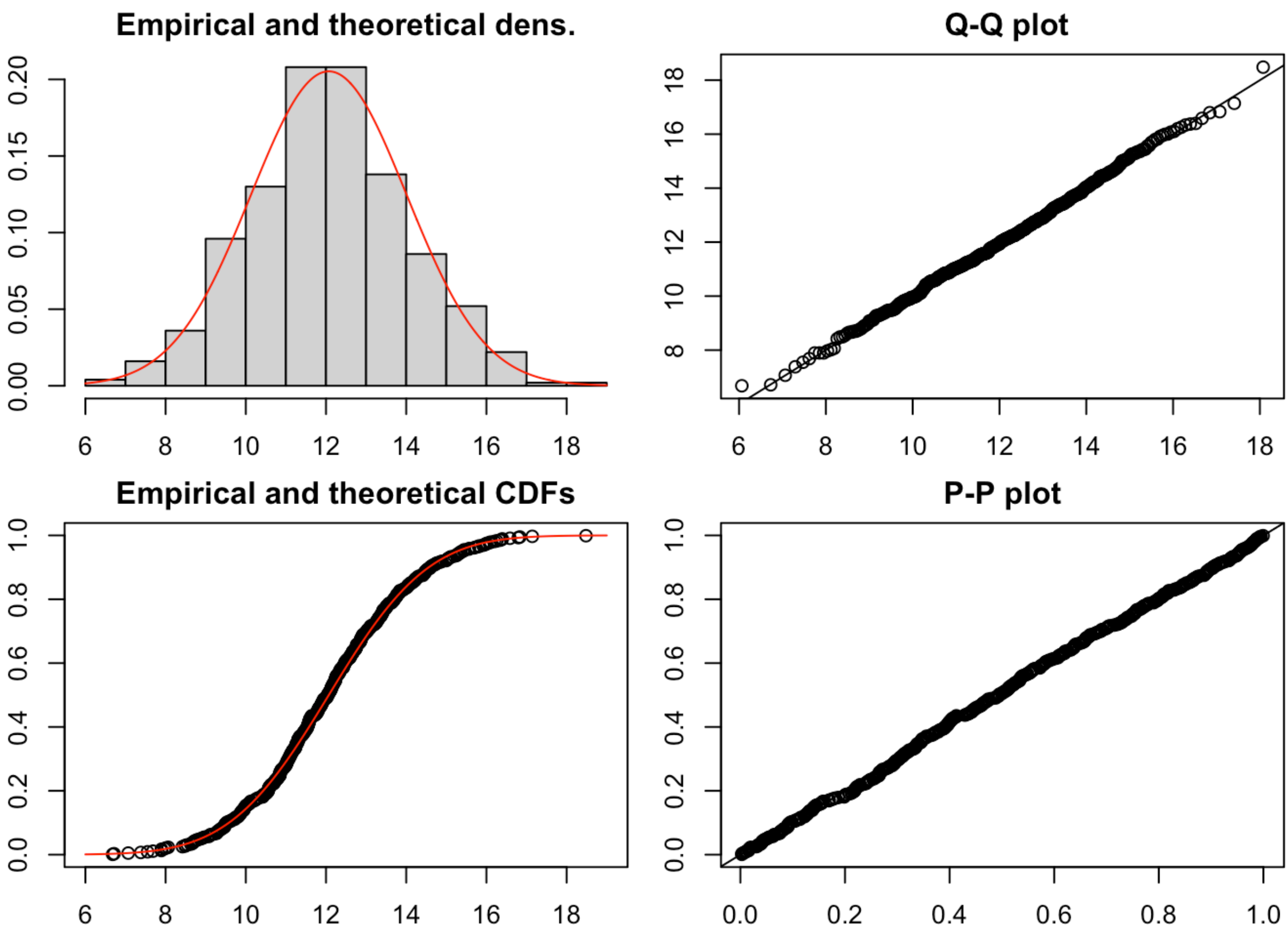
```
set.seed(123)
data <- data.frame(education.num = rnorm(500, mean = 12, sd = 2))  # Simulated dat
a

# Fit a normal distribution to dataset B (education.num)
fit <- fitdist(data$education.num, "norm")

# Print the summary of the fitted distribution
cat("Summary of the fitted normal distribution:\n")
```

```
## Summary of the fitted normal distribution:
```

```
summary(fit)
```

```
## Fitting of the distribution ' norm ' by maximum likelihood
## Parameters :
##       estimate Std. Error
## mean 12.069181 0.08692009
## sd    1.943592 0.06146171
## Loglikelihood:  -1041.738   AIC:  2087.476   BIC:  2095.906
## Correlation matrix:
##               mean            sd
## mean  1.000000e+00 -2.277541e-09
## sd   -2.277541e-09  1.000000e+00
```

```
# Visualize the fitted distribution
plot(fit)
```



```
# Conduct a goodness-of-fit test
gof_results <- gofstat(fit)

# Print the goodness-of-fit test results
cat("Goodness-of-Fit Test Results:\n")
```

```
## Goodness-of-Fit Test Results:
```

```
print(gof_results)
```

```
## Goodness-of-fit statistics
##                              1-mle-norm
## Kolmogorov-Smirnov statistic 0.02300307
## Cramer-von Mises statistic   0.04776675
## Anderson-Darling statistic   0.28007918
##
## Goodness-of-fit criteria
##                                1-mle-norm
## Akaike's Information Criterion   2087.476
## Bayesian Information Criterion   2095.906
```

```
# QUESTION 7

library(car)
```

```
## Loading required package: carData
```

```
B <- data$education.num

# (a) Divide dataset B into two subsets B1 and B2 (3:2 ratio)
set.seed(123)
sample_indices_B <- sample(1:length(B), size = 0.6 * length(B))  # 60% for B1
B1 <- B[sample_indices_B]
B2 <- B[-sample_indices_B]  # Remaining 40% for B2

# (b) Levene's test for equality of variances between B1 and B2

levene_test_B <- leveneTest(c(B1, B2) ~ factor(c(rep(1, length(B1)), rep(2, lengt
h(B2)))))
cat("Levene's Test for Equality of Variances between B1 and B2 (hours.per.week):\
n")
```

```
## Levene's Test for Equality of Variances between B1 and B2 (hours.per.week):
```

```
print(levene_test_B)
```

```
## Levene's Test for Homogeneity of Variance (center = median)
##        Df F value  Pr(>F)
## group   1  3.3522 0.06771 .
##       498
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
# (c) t-test for equality of means between B1 and B2
t_test_result_B <- t.test(B1, B2)
cat("t-test for Equality of Means between B1 and B2 (hours.per.week):\n")
```

```
## t-test for Equality of Means between B1 and B2 (hours.per.week):
```

```
print(t_test_result_B)
```

```
##
##  Welch Two Sample t-test
##
## data:  B1 and B2
## t = -0.95988, df = 395.02, p-value = 0.3377
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -0.5307886  0.1825201
## sample estimates:
## mean of x mean of y
##  11.99953  12.17366
```

```
# (d) Confidence Interval for the difference of means (99% CI)
conf_interval_B <- t.test(B1, B2)$conf.int
cat("99% Confidence Interval for the Difference in Means (hours.per.week):\n")
```

```
## 99% Confidence Interval for the Difference in Means (hours.per.week):
```

```
print(conf_interval_B)
```

```
## [1] -0.5307886  0.1825201
## attr(,"conf.level")
## [1] 0.95
```