Final Exam

Ariz Kazani

2024-08-22

Final Exam

Name: Ariz Kazani

Student ID: 101311311

Notes

The CSV file required for question 1, should be in the same directory as this folder.

```r
# Libraries

# NOTE: if you do not have any of the below libraries installed,
# un-comment the line and run it

# install.packages("rmarkdown")
library(rmarkdown)

# install.packages("ggplot2")
library(ggplot2)
```

```
## Warning: package 'ggplot2' was built under R version 4.4.1
```

```r
# install.packages("dplyr")
library(dplyr)
```

```
## Warning: package 'dplyr' was built under R version 4.4.1
```

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```r
#install.packages("ompr")
library(ompr)
```

```
## Warning: package 'ompr' was built under R version 4.4.1
```

```r
#install.packages("ompr.roi")
library(ompr.roi)
```

```
## Warning: package 'ompr.roi' was built under R version 4.4.1
```

```r
#install.packages("ROI.plugin.glpk")
library(ROI.plugin.glpk)
```

```
## Warning: package 'ROI.plugin.glpk' was built under R version 4.4.1
```

Instructions

1. For full marks, you must show and justify all steps and interpret all aspects/conclusions of your work within the context of the question.
2. The exam will run Thursday August 22, 2024. It will open in Brightspace at 12:01AM and you will need to submit it by 11:59PM. There will be no late submissions for any reason. The submission format will be the same as all assignments. You will need to submit a single pdf of your work as well as a single file with all your code such that I can verify it runs and produces the same results as what is in your pdf file.
3. You are not allowed to work in groups, use any AI software to help you answer questions. This work is to be yours and yours alone. Any student found violating the Academic Integrity Policy will automatically receive an F.
4. Approved resources: (a) RStudio provided on lab computers. (b) Instructors Brightspace lecture notes and R files.
5. It is the student's responsibility to have read and understood all instructions.

Solutions

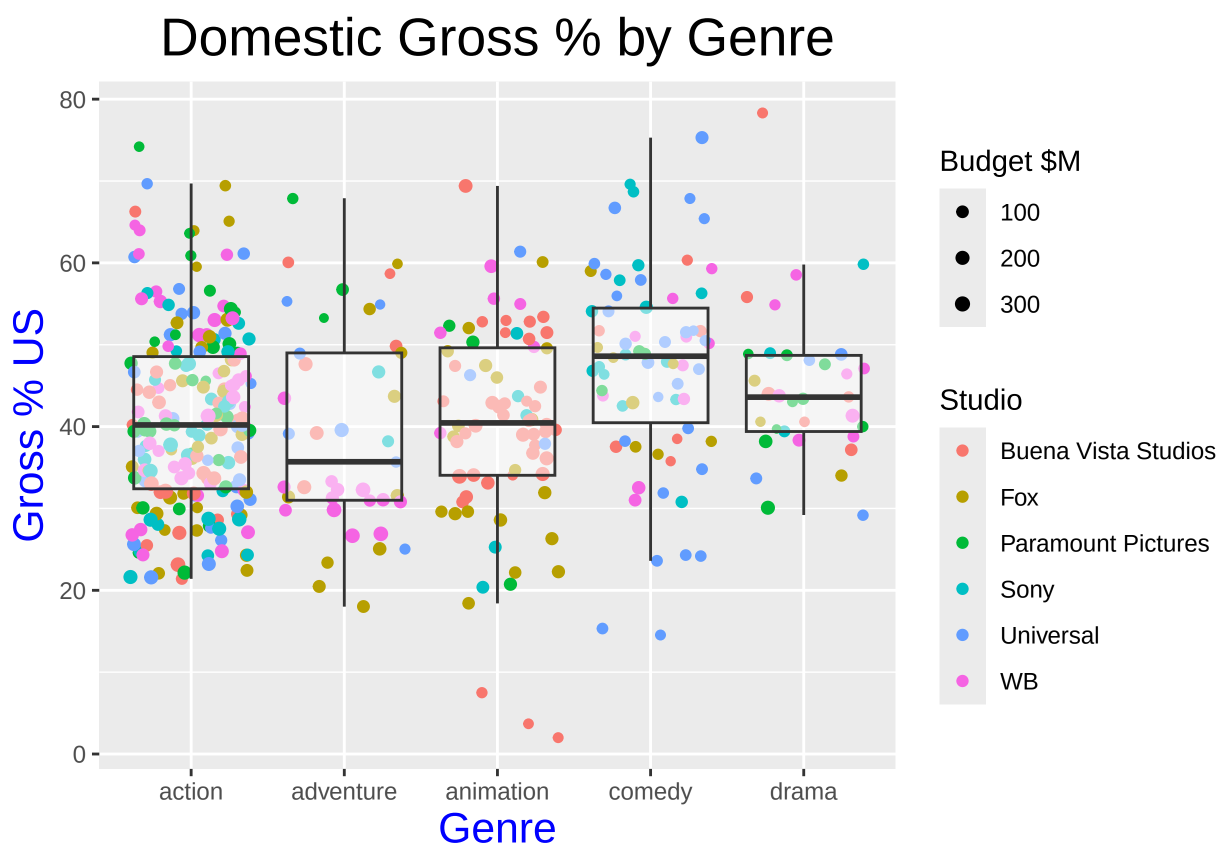
1.

A previous consultant had created a chart for a movie review website which is illustrated below. However, the R code used to create the diagram has since been lost and cannot be recovered. Your task is to come up with the code that will re-create the same chart making it look as close as possible to the original. A new dataset has been supplied – MovieReviewData.csv.

```r
movieReviewData <- read.csv("MovieReviewData.csv")

studiosTK <- c("Buena Vista Studios",
               "Fox",
               "Paramount Pictures",
               "Sony",
               "Universal",
               "WB")
genresTK <- c("action", "adventure", "animation", "comedy", "drama")
movieReviewData <- movieReviewData %>% filter(Studio %in% studiosTK)
movieReviewData <- movieReviewData %>% filter(Genre %in% genresTK)

set.seed(7)
ggplot(movieReviewData) +
  geom_jitter(aes(
    x = Genre,
    y = Gross...US,
    colour = Studio,
    size = Budget...mill.,
  )) +
  scale_size_continuous(range = c(1, 2)) +
  geom_boxplot(
    aes(x = Genre, y = Gross...US),
    alpha = 0.5,
    size = 0.5,
    outlier.shape = NA
  ) +
  labs(title = "Domestic Gross % by Genre",
       x = "Genre",
       y = "Gross % US",
       size = "Budget $M") +
  theme(
    plot.title = element_text(hjust = 0.5, size = 20),
    axis.title.x = element_text(color = "blue", size = 16),
    axis.title.y = element_text(color = "blue", size = 16),
  )
```



2.

One very early method for pseudo random number generation is von Neumann's middle square method (von Neumann, 1951). The method works as follows: starting with X_0 E {0, 1, . . . , 99}, define X_n for n E N to be the middle two digits of the four-digit number X_{n-1}ˆ2. If X_{n-1}ˆ2 does not have four digits, it is padded with leading zeros.

A. Write a function which computes x_n from X_{n-1}.

```
getXn <- function(xPrev) {
  x2 <- xPrev * xPrev
  if (x2 < 10) {
    return(0)
  } else {
    x2 <- x2 %/% 10
    x2 <- x2 %% 100
    return(x2)
  }
}
```

```r
# testinng function
getXn(11) #should be 12
```

```
## [1] 12
```

```r
getXn(99) #should be 80
```

```
## [1] 80
```

```r
getXn(50) #should be 50
```

```
## [1] 50
```

B. The output of the middle square method has loops. For example, once we have $\mathbf{X_N = 0}$, we will have $\mathbf{X_n =}$

0 for all n >= N. Write a program to find all cycles of the middle square method.

```r
# technically all numbers will eventually create a cycle, I'm going to
# assume this question is asking for what numbers is xn-1 == xn
getCycles <- function() {
  rv <- c()
  for (i in 0:99) {
    if (i == getXn(i)) {
      rv <- c(rv, i)
    }
  }
  return(rv)
}

getCycles()
```

```
## [1]  0 10 50 60
```

C. Comment on the quality of the middle square method as a PRNG.

The middle square method is very bad. first of all 4% of the number cause the output to be the input number. Also since the input number can be a maximum of 99, there is a very small number of numbers that could be generated. Since there are so few numbers that can be generated, numbers will start to repeat after only a few cycles.

3.

Its 9am on a Friday when your boss, Todd, comes into the office demanding that you have two reports on his desk by end of day. His demands today are particularly frustrating to you as you scheduled months ago a meet up with your friends that's taking place at 6pm. Now, for better or worse, you want to keep your job, but you also don't want to miss out on the event that is in just 9 hours. You ask yourself what is the chance that you could even finish two reports in time for the party and how could you actually even go about computing that chance?

Using a Monte Carlo simulation, what is the probability that you will not make it to the party? Perform at least 10000 simulations. You may assume that report 1 takes between 2 to 6 hours to complete and report 2 takes between 3 to 7 hours to complete and your performance on one report has no bearing on the other.

```r
getProbParty <- function(n) {
  r1 <- runif(n, 2, 6)
  r2 <- runif(n, 1, 7)
  return(r1 + r2)
}

set.seed(7)
numFail <- sum(getProbParty(10000) > 9)
cat("The probablility I will not make it to the party is.",
    numFail / 10000,
    "%\n")
```

```
## The probablility I will not make it to the party is. 0.3373 %
```

4.

Using the bisection method, show that f(x) = x^3 - sin^2(x)

A. Has a root between [0.5,1].

```r
fx <- function(x) {
  return(x ^ 3 - sin(x) ^ 2)
}

bisection <- function(f, a, b, tol = 1e-4, maxI = 1000) {
  if (f(a) * f(b) >= 0) {
    return("fail")
  }

  i <- 0

  repeat {
    c <- (a + b) / 2
    if (f(c) == 0) {
      return(c)
    } else if (f(a) * f(c) < 0) {
      b <- c
    } else {
      a <- c
    }
    if (abs(f(c)) < tol || i >= maxI) {
      i <- i + 1
      break

    }
    i <- i + 1
  }
  return(list(ans = c, itterations = i))
}

bisection(fx, 0.5, 1) != "fail"
```

```
##         ans itterations
##        TRUE        TRUE
```

Since we got an answer and not an error we can conclude that there is a root between [0.5, 1]

B. Determine an approximation to the root that is accurate to at least within 10^-4.

```
bisection(fx, 0.5, 1)
```

```
## $ans
## [1] 0.8027344
##
## $itterations
## [1] 8
```

C. Determine the maximum number of iterations necessary to solve f(x) = 0 with 10^-4. How does this compare with (b).

```
maxNumI <- function(a, b, tol) {
  return(ceiling(log2(abs(b - a) / tol)))
}

maxNumI(0.5, 1, 1e-4)
```

```
## [1] 13
```

The maximum number of iterations depends depends in the initial guess of the a and b values as well as the tolerance. I have written a function above that does so. We can see that the maximum amount given was 13 and we got 8.

5.

Alice has two favorite foods: pasta and salad. If she has pasta one night, there is a 40% chance she will have pasta again the next night. If she has salad, there is a 70% chance she will have salad the next night.

A. If Alice has pasta for dinner on Monday night, what is the probability that she will have salad on Thursday?

Note: I'm going to assume if Alice doesn't eat one she is going to eat the other.

```r
pTrans <- matrix(c(0.4, 0.7, 0.6, 0.3), nrow = 2, byrow = TRUE)

MC <- function(tMatrix, start, numSteps) {
  p <- start
  for (i in 1:numSteps) {
    p <- tMatrix %*% p
  }

  return(p)
}

MC(pTrans, c(1, 0), 4)
```

```
##        [,1]
## [1,] 0.5422
## [2,] 0.4578
```

There is a 45.78% chance that Alice will have salad on Thursday.

B. In the long run, what percentage of her dinners will be salad?

```r
MC(pTrans, c(1, 0), 999)
```

```
##           [,1]
## [1,] 0.5384615
## [2,] 0.4615385
```

In the long term Alice will have a salad about 46.15% of the time

C. Use a Monte Carlo simulation to numerically determine the distribution of dinner choices on day 5.

```r
simMC <- function(tMatrix, start, numRow, numSteps) {
  states <- numeric(numSteps + 1)
  states[1] <- sample(1:numRow, 1, prob = start)
  for (i in 1:numSteps) {
    states[i + 1] <- sample(1:numRow, 1, prob = tMatrix[, states[i]])
  }

  return(states)
}

set.seed(7)
simMC(pTrans, c(1, 0), 2, 5)
```

```
## [1] 1 2 1 2 1 1
```

If Alice starts with pasta, she will eat pasta 3 times over the next 5 days

D. Analytically compute the distribution of dinner choices on day 5.

```
p <- MC(pTrans, c(1, 0), 5)
p
```

```
##          [,1]
## [1,] 0.53734
## [2,] 0.46266
```

```
p[1, 1] * 5
```

```
## [1] 2.6867
```

```
p[2, 1] * 5
```

```
## [1] 2.3133
```

Based on the probabilities we can see that Alice would have had 3 pastas and 2 salads by rounding the probabilities.

6.

The Spooky Boogie Costume Salon makes and sells four different Halloween costumes: the witch, the ghost, the goblin, and the werewolf. Each witch costume uses 3 yards of material and takes 3 hours to sew. Each ghost costume uses 2 yards of material and takes 1 hour to sew. Each goblin costume uses 2 yards of material and takes 3 hours to sew. Each werewolf costume uses 2 yards of material and takes 4 hours to sew. The profits for each costume are as follows: $10 for the witch, $8 for the ghost, $12 for the goblin, and $16 for the werewolf.

If they have 600 yards of material and 510 sewing hours available before the holiday, how many of each costume should they make to maximize profit, assuming they sell everything they make?

```
# Let x1 = witch, x2 = ghost, x3 = goblin, x4 = goblin
# Objective Function
# max <- 10 * x1 + 8 * x2 + 12 * x3 + 16 * x4

# Constraints
# 3 * x1 + 2 * x2 + 2 * x3 + 2 * x4 <= 600
# 3 * x1 + x2 + 3 * x3 + 4 * x4 <= 510
# x1, x2, x3, x4 >=0

model <- MIPModel() %>%
```

```r
  add_variable(x1, type = "integer", lb = 0) %>%
  add_variable(x2, type = "integer", lb = 0) %>%
  add_variable(x3, type = "integer", lb = 0) %>%
  add_variable(x4, type = "integer", lb = 0) %>%
  set_objective(10 * x1 + 8 * x2 + 12 * x3 + 16 * x4, "max") %>%
  add_constraint(3 * x1 + 2 * x2 + 2 * x3 + 2 * x4 <= 600) %>%
  add_constraint(3 * x1 + x2 + 3 * x3 + 4 * x4 <= 510)

result <- solve_model(model, with_ROI(solver = "glpk"))

solution <- bind_cols(
  x1 = get_solution(result, x1),
  x2 = get_solution(result, x2),
  x3 = get_solution(result, x3),
  x4 = get_solution(result, x4),
  max = result %>%
    objective_value()
)

solution
```

```
## # A tibble: 1 x 5
##      x1    x2    x3    x4   max
##   <dbl> <dbl> <dbl> <dbl> <dbl>
## 1     0   230     0    70  2960
```

There for

7.

A. Use the bootstrap method to estimate the median and 95% confidence interval of the following dataset:

data <- c(22.5, 24.3, 21.7, 23.8, 22.0, 25.1, 24.6, 23.1, 22.9, 21.9)

Write an R function to perform bootstrap resampling and calculate the median for each resample. Perform 10,000 bootstrap resamples.

```r
data <- c(22.5, 24.3, 21.7, 23.8, 22.0, 25.1, 24.6, 23.1, 22.9, 21.9)
bootstrapMedian <- function(data, resamples) {
  n <- length(data)
  medians <- numeric(resamples)
  for (i in 1:resamples) {
    resample <- sample(data, n, replace = TRUE)
    medians[i] <- median(resample)
  }
  return(medians)
}
```

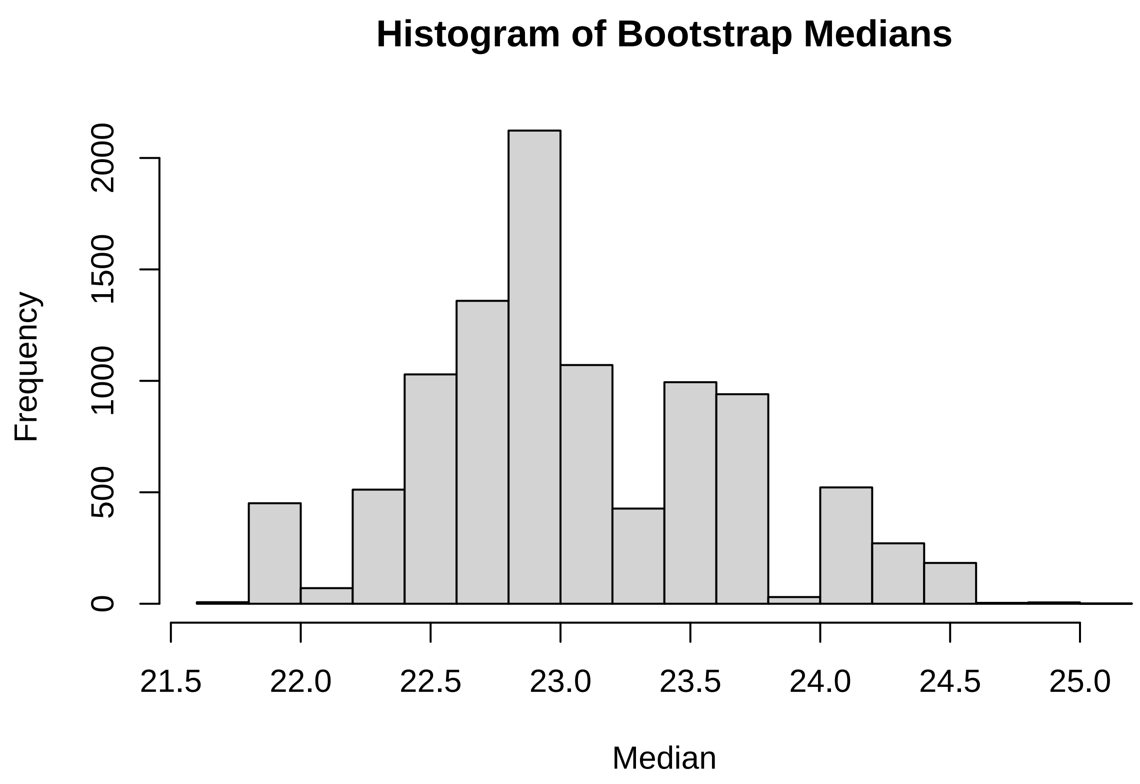
```r
set.seed(7)
BSMED <- bootstrapMedian(data, 10000)
BSCI <- quantile(BSMED, c(0.025, 0.975))
BSCI
```

```
##  2.5% 97.5%
##  22.0  24.3
```

B. Construct a histogram of the bootstrap medians and calculate the 95% confidence interval for the median based on

the bootstrap samples.

```r
hist(BSMED, main = "Histogram of Bootstrap Medians", xlab = "Median")
```



```r
BSCI <- quantile(BSMED, c(0.025, 0.975))
BSCI
```

```
##  2.5% 97.5%
##  22.0  24.3
```

C. Compare the bootstrap confidence interval with the theoretical confidence interval for the median based on the

binomial distribution. Discuss any differences.

```r
mTCI <- mean(data)
sTCI <- sd(data)
nTCI <- length(data)
seTCI <- sTCI / sqrt(nTCI)
TCI <- mTCI + c(qnorm(0.025), qnorm(0.975)) * seTCI
print("Theoretical Confidence Interval")
```

```
## [1] "Theoretical Confidence Interval"
```

```r
TCI
```

```
## [1] 22.44198 23.93802
```

```r
print("Bootstrap Confidence Interval")
```

```
## [1] "Bootstrap Confidence Interval"
```

```r
BSCI
```

```
##  2.5% 97.5%
##  22.0  24.3
```

The bootstrap confidence interval is slightly wider. This could be because of variance when drawing a random sample.

D. Investigate the effect of varying the number of bootstrap resamples on the confidence interval by repeating the

bootstrap process with 1,000, 5,000, and 20,000 resamples. Compare the results and discuss any observed trends.

```r
set.seed(7)
BSMED <- bootstrapMedian(data, 1000)
BSCI <- quantile(BSMED, c(0.025, 0.975))
BSCI
```

```
##  2.5% 97.5%
## 22.00 24.45
```

```r
BSMED <- bootstrapMedian(data, 5000)
BSCI <- quantile(BSMED, c(0.025, 0.975))
BSCI
```

```
##  2.5% 97.5%
##  22.0  24.3
```

```r
BSMED <- bootstrapMedian(data, 20000)
BSCI <- quantile(BSMED, c(0.025, 0.975))
BSCI
```

```
##  2.5% 97.5%
##  22.0  24.3
```

Between 1000 and 5000 the confidence interval got slightly narrower. It did not get any narrower after that. This could indicate a non-binomial distribution.

8.

A. Consider the following dataset where each value represents a success (1) or failure (0):

data <- c(1, 0, 1, 0, 1, 1, 0, 0, 1, 0, 1, 0)

Use the bootstrap method to estimate the proportion of successes and the 95% confidence interval for the proportion.

```r
data <- c(1, 0, 1, 0, 1, 1, 0, 0, 1, 0, 1, 0)
bootstrapM <- function(data, resamples) {
  n <- length(data)
  s <- numeric(resamples)
  for (i in 1:resamples) {
    resample <- sample(data, n, replace = TRUE)
    # since only two, mean can be used to calculate probability of success
    s[i] <- mean(resample)
  }
  return(s)
}

set.seed(7)
BSMED <- bootstrapM(data, 10000)
BSCI <- quantile(BSMED, c(0.025, 0.975))
BSCI
```

```
##  2.5% 97.5%
##  0.25  0.75
```

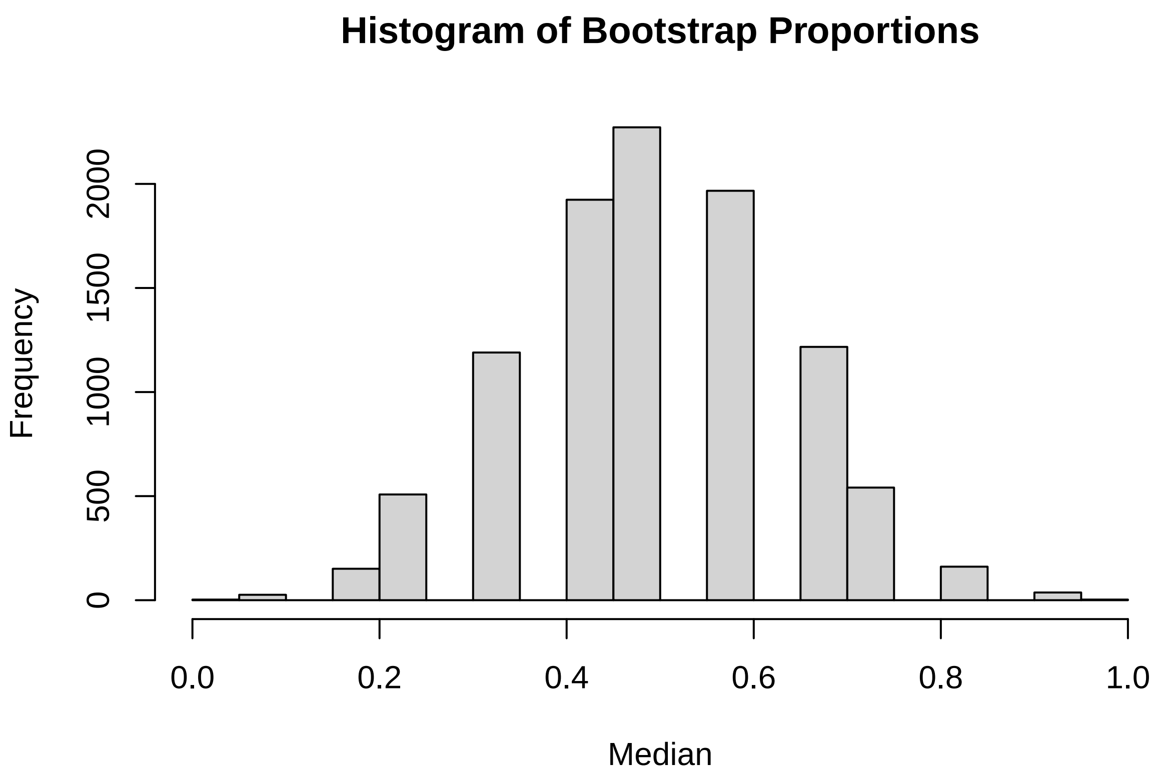
B. Perform 10,000 bootstrap resamples and calculate the proportion for each resample.

```r
set.seed(7)
BSMED <- bootstrapM(data, 10000)
BSCI <- quantile(BSMED, c(0.025, 0.975))
BSCI
```

```
##  2.5% 97.5%
##  0.25  0.75
```

C. Construct a histogram of the bootstrap proportions and calculate the 95% confidence interval.

```r
hist(BSMED, main = "Histogram of Bootstrap Proportions", xlab = "Median")
```



```r
BSCI <- quantile(BSMED, c(0.025, 0.975))
BSCI
```

```
##  2.5% 97.5%
##  0.25  0.75
```

The bootstrap confidence interval is slightly wider. This could be because of variance when drawing a random sample.

D. Compare the bootstrap confidence interval with the theoretical confidence interval based on the normal

approximation to the binomial distribution. Discuss any differences.

```r
mTCI <- mean(data)
sTCI <- sd(data)
nTCI <- length(data)
seTCI <- sTCI / sqrt(nTCI)
```

```r
TCI <- mTCI + c(qnorm(0.025), qnorm(0.975)) * seTCI
print("Theoretical Confidence Interval")
```

```
## [1] "Theoretical Confidence Interval"
```

```r
TCI
```

```
## [1] 0.2045243 0.7954757
```

```r
print("Bootstrap Confidence Interval")
```

```
## [1] "Bootstrap Confidence Interval"
```

```r
BSCI
```

```
##  2.5% 97.5%
##  0.25  0.75
```

The bootstrap confidence interval is narrower indicating a binomial distribution.

D. Assess the stability of the bootstrap confidence intervals by calculating the bootstrap confidence intervals for

different numbers of resamples (e.g., 1,000, 5,000, and 20,000). Compare the intervals and discuss how the number of resamples affects the precision and stability of the bootstrap estimates.

```r
set.seed(7)
BSMED <- bootstrapM(data, 1000)
BSCI <- quantile(BSMED, c(0.025, 0.975))
BSCI
```

```
##  2.5% 97.5%
##  0.25  0.75
```

```r
BSMED <- bootstrapM(data, 5000)
BSCI <- quantile(BSMED, c(0.025, 0.975))
BSCI
```

```
##  2.5% 97.5%
##  0.25  0.75
```

```r
BSMED <- bootstrapM(data, 20000)
BSCI <- quantile(BSMED, c(0.025, 0.975))
BSCI
```

```
##  2.5% 97.5%
##  0.25  0.75
```

The bootsrap estimates stay the same indicating a high level of precision and stability.