ST551: HOMEWORK 5

Brian Cervantes Alvarez
November 28, 2023

Question 1

Part A

When we're dealing with the original data, the hypothesis test is built around how the data looks and what we assume about it. We set up null and alternative hypotheses, based on the original size and pattern of the data. But, if we transform the data, like using a log, it changes things. The scale and even the shape of the data might shift. So, we have to rethink or redo the hypothesis test to match the transformed data. And when we interpret the results, we need to talk about the new size and pattern introduced by the transformation.

```
set.seed(123)
#------------------------------------------------------------------------------#

# First data set
m1 <- 20
n1 <- 50
alpha <- 0.05

# Untransformed data
rejectionRatesUntransformed1 <- replicate(10000, {
  sample1 <- rchisq(m1, df = 4)
  sample2 <- rgamma(n1, shape = 0.5, scale = 8)
  tTestResult <- t.test(sample1, sample2)
  tTestResult$p.value < alpha
})

# Log-transformed data
rejectionRatesTransformed1 <- replicate(10000, {
  sample1 <- log(rchisq(m1, df = 4))
  sample2 <- log(rgamma(n1, shape = 0.5, scale = 8))
  tTestResult <- t.test(sample1, sample2)
  tTestResult$p.value < alpha
})

#------------------------------------------------------------------------------#

# Second data set
m2 <- 200
n2 <- 500

# Untransformed data
rejectionRatesUntransformed2 <- replicate(10000, {
  sample1 <- rchisq(m2, df = 4)
  sample2 <- rgamma(n2, shape = 0.5, scale = 8)
```

```r
  tTestResult <- t.test(sample1, sample2)
  tTestResult$p.value < alpha
})

# Log-transformed data
rejectionRatesTransformed2 <- replicate(10000, {
  sample1 <- log(rchisq(m2, df = 4))
  sample2 <- log(rgamma(n2, shape = 0.5, scale = 8))
  tTestResult <- t.test(sample1, sample2)
  tTestResult$p.value < alpha
})

#-------------------------------------------------------------------------------#

# Third data set
m3 <- 20
n3 <- 50

# Untransformed data
rejectionRatesUntransformed3 <- replicate(10000, {
  sample1 <- rchisq(m3, df = 4)
  sample2 <- rgamma(n3, shape = 0.5, scale = 21.75)
  tTestResult <- t.test(sample1, sample2)
  tTestResult$p.value < alpha
})

# Log-transformed data
rejectionRatesTransformed3 <- replicate(10000, {
  sample1 <- log(rchisq(m3, df = 4))
  sample2 <- log(rgamma(n3, shape = 0.5, scale = 21.75))
  tTestResult <- t.test(sample1, sample2)
  tTestResult$p.value < alpha
})


#-------------------------------------------------------------------------------#

# Fourth Data set
m4 <- 200
n4 <- 500

# Untransformed data
rejectionRatesUntransformed4 <- replicate(10000, {
  sample1 <- rchisq(m4, df = 4)
  sample2 <- rgamma(n4, shape = 0.5, scale = 21.75)
  tTestResult <- t.test(sample1, sample2)
  tTestResult$p.value < alpha
})

# Log-transformed data
rejectionRatesTransformed4 <- replicate(10000, {
  sample1 <- log(rchisq(m4, df = 4))
```

```
  sample2 <- log(rgamma(n4, shape = 0.5, scale = 21.75))
  tTestResult <- t.test(sample1, sample2)
  tTestResult$p.value < alpha
})

# Print Rejection Rates for Data set 1
print(paste0("Data set 1 - Untransformed Data: Rejection Rate =", mean(rejectionRatesUntrans
```

```
[1] "Data set 1 - Untransformed Data: Rejection Rate =0.0473"
```

```
print(paste0("Data set 1 - Log-transformed Data: Rejection Rate =", mean(rejectionRatesTrans
```

```
[1] "Data set 1 - Log-transformed Data: Rejection Rate =0.8018"
```

```
# Print Rejection Rates for Data set 2
print(paste0("Data set 2 - Untransformed Data: Rejection Rate =", mean(rejectionRatesUntrans
```

```
[1] "Data set 2 - Untransformed Data: Rejection Rate =0.052"
```

```
print(paste0("Data set 2 - Log-transformed Data: Rejection Rate =", mean(rejectionRatesTrans
```

```
[1] "Data set 2 - Log-transformed Data: Rejection Rate =1"
```

```
# Print Rejection Rates for Data set 3
print(paste0("Data set 3 - Untransformed Data: Rejection Rate =", mean(rejectionRatesUntrans
```

```
[1] "Data set 3 - Untransformed Data: Rejection Rate =0.9408"
```

```
print(paste0("Data set 3 - Log-transformed Data: Rejection Rate =", mean(rejectionRatesTrans
```

```
[1] "Data set 3 - Log-transformed Data: Rejection Rate =0.0506"
```

```
# Print Rejection Rates for Data set 4
print(paste0("Data set 4 - Untransformed Data: Rejection Rate =", mean(rejectionRatesUntrans
```

```
[1] "Data set 4 - Untransformed Data: Rejection Rate =1"
```

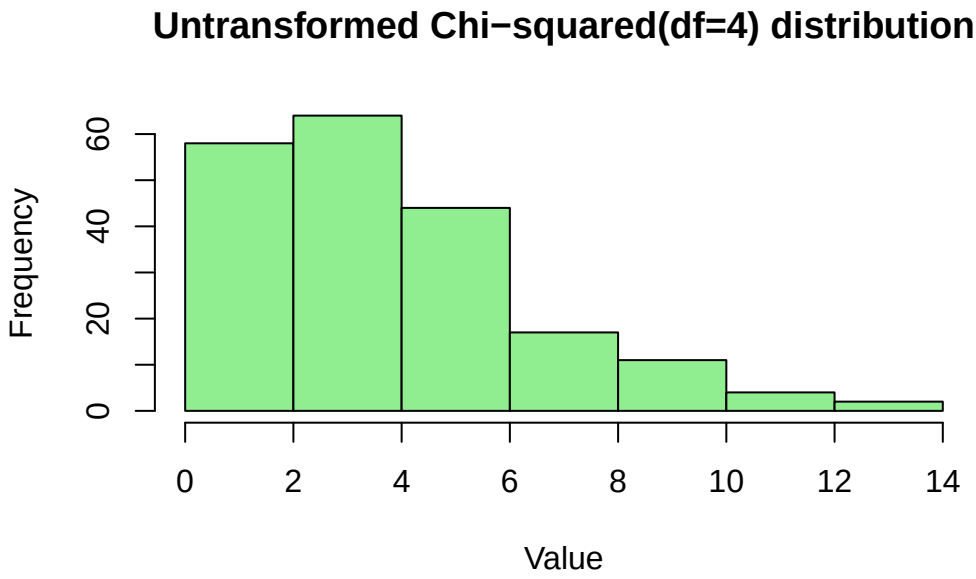
```
print(paste0("Data set 4 - Log-transformed Data: Rejection Rate =", mean(rejectionRatesTrans
```

```
[1] "Data set 4 - Log-transformed Data: Rejection Rate =0.0522"
```

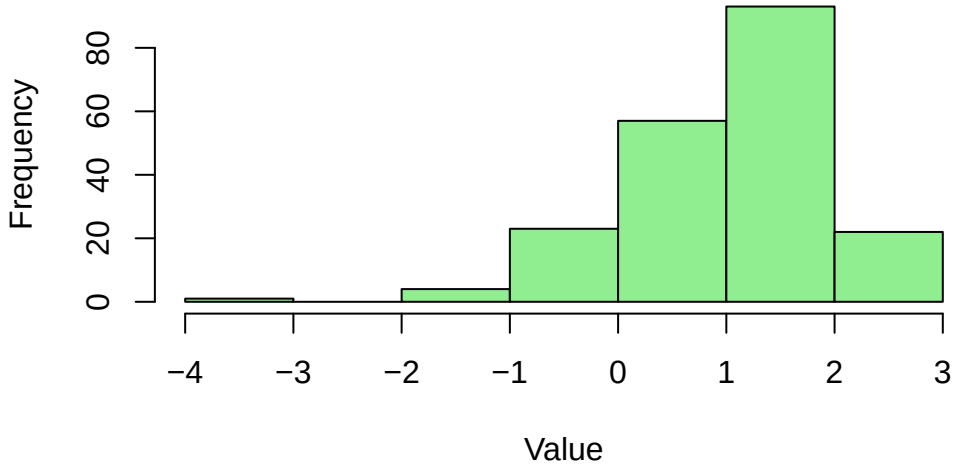
Part B

My takeaway is that the histograms show a more normal distribution after applying the log-transformation, signaling a positive outcome in addressing skewness in the data. I find that this transformation often proves effective in promoting a more symmetric shape. Nevertheless, it's crucial to exercise caution and consider the demands we have for our analysis.

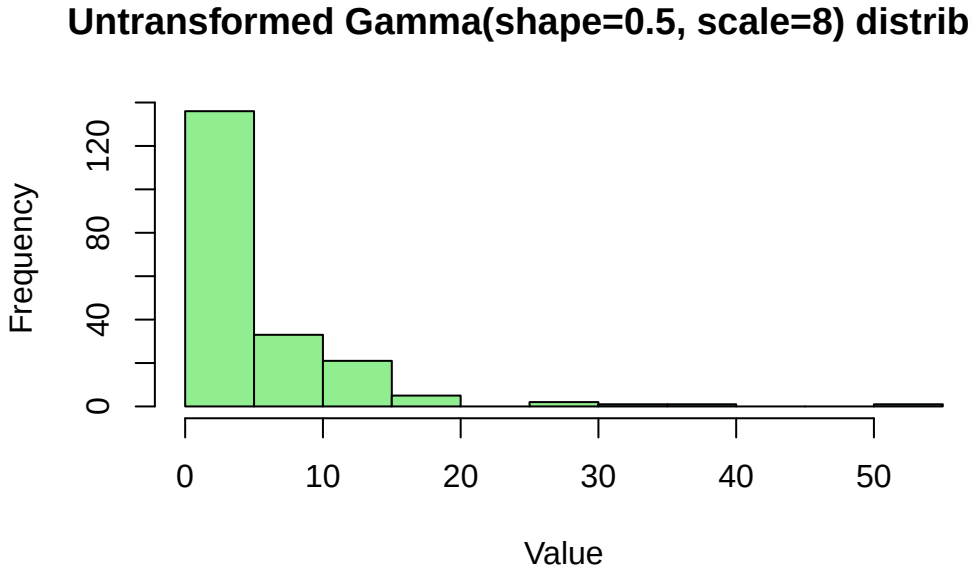
```r
set.seed(42)

# Untransformed Chi-squared(df=4) distribution
hist1 <- rchisq(200, df = 4)
hist(hist1, main = "Untransformed Chi-squared(df=4) distribution",
     xlab = "Value",
     ylab = "Frequency",
     col = "lightgreen",
     border = "black")
```




```r
# Log-transformed
hist2 <- log(rchisq(200, df = 4))
hist(hist2, main = "Log-transformed Chi-squared(df=4) distribution",
     xlab = "Value",
     ylab = "Frequency",
     col = "lightgreen",
     border = "black")
```

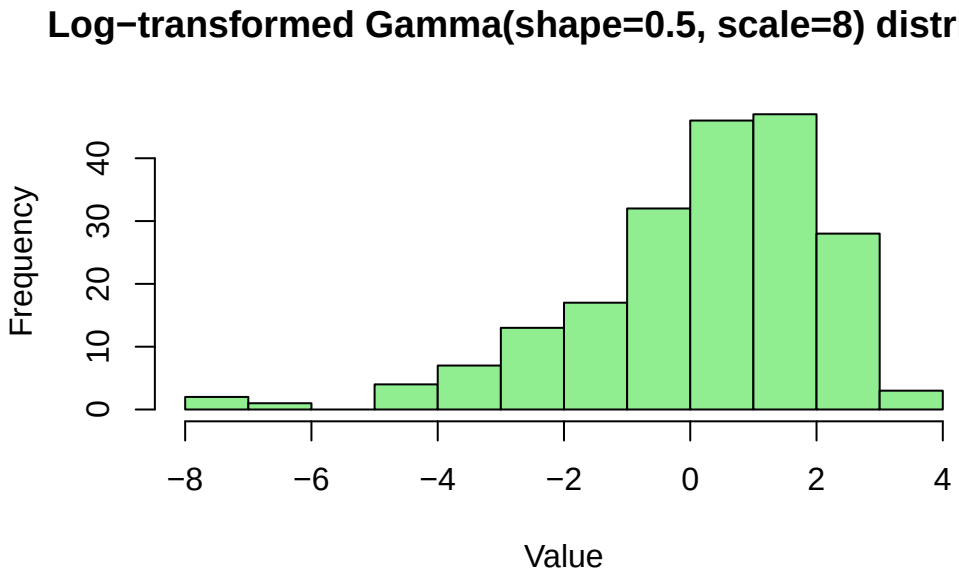



```r
# Untransformed Gamma(shape=0.5, scale=8) distribution
hist3 <- rgamma(200, shape = 0.5, scale = 8)
hist(hist3, main = "Untransformed Gamma(shape=0.5, scale=8) distribution",
     xlab = "Value",
     ylab = "Frequency",
     col = "lightgreen",
     border = "black")
```

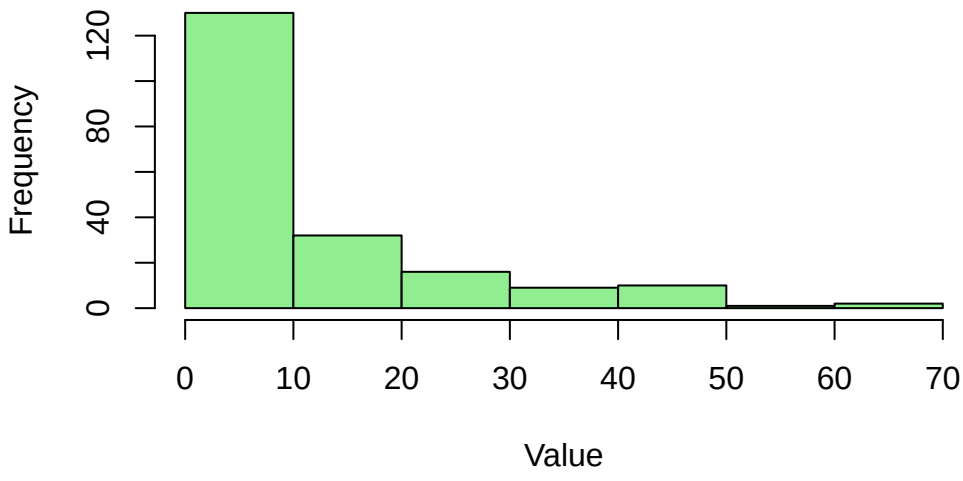



```r
# Log-transformed
hist4 <- log(rgamma(200, shape = 0.5, scale = 8))
hist(hist4, main = "Log-transformed Gamma(shape=0.5, scale=8) distribution",
     xlab = "Value",
```

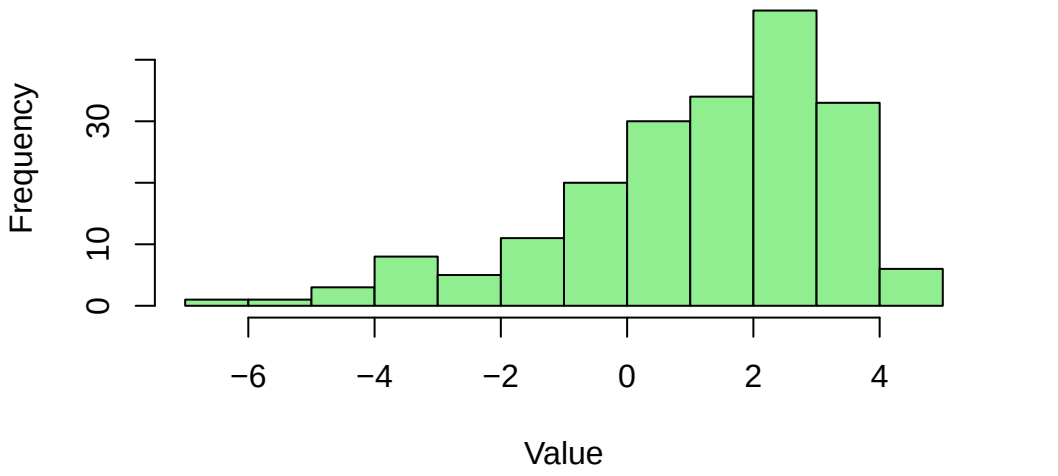
```
      ylab = "Frequency",
      col = "lightgreen",
      border = "black")
```




```
# Untransformed Gamma(shape=0.5, scale=21.75) distribution
hist5<- rgamma(200, shape = 0.5, scale = 21.75)
hist(hist5, main = "Untransformed Gamma(shape=0.5, scale=21.75) distribution",
     xlab = "Value",
     ylab = "Frequency",
     col = "lightgreen",
     border = "black")
```



```r
# Log-transformed
hist6 <- log(rgamma(200, shape = 0.5, scale = 21.75))
hist(hist6, main = "Log-transformed Gamma(shape=0.5, scale=21.75) distribution",
     xlab = "Value", ylab = "Frequency", col = "lightgreen", border = "black")
```



Part C

```r
# Set seed for reproducibility
set.seed(42)

# Parameters
numSimulations <- 10000
nSamp <- 1000

# Initialize variables
chiSquaredMeans <- 0
gamma1Means <- 0
gamma2Means <- 0

# Initialize log-transformed variables
logChiSquaredMeans <- 0
logGamma1Means <- 0
logGamma2Means <- 0

# Perform simulations
for (i in 1:numSimulations) {
  # Simulate random samples from Chi-squared distribution
  chiSquaredSample <- rchisq(nSamp, 4)

  # Simulate random samples from Gamma distributions
  gammaSample1 <- rgamma(nSamp, shape = 0.5, scale = 8)
  gammaSample2 <- rgamma(nSamp, shape = 0.5, scale = 21.75)

  # Calculate means and store results
  chiSquaredMeans[i] <- mean(chiSquaredSample)
  gamma1Means[i] <- mean(gammaSample1)
  gamma2Means[i] <- mean(gammaSample2)

  # Calculate log-transformed means and store results
  logChiSquaredMeans[i] <- mean(log(chiSquaredSample))
  logGamma1Means[i] <- mean(log(gammaSample1))
  logGamma2Means[i] <- mean(log(gammaSample2))
}

# Calculate average of the simulation results
averageChiSquaredMean <- mean(chiSquaredMeans)
averageGamma1Mean <- mean(gamma1Means)
averageGamma2Mean <- mean(gamma2Means)

# Calculate average of log-transformed simulation results
averageLogChiSquaredMean <- mean(logChiSquaredMeans)
averageLogGamma1Mean <- mean(logGamma1Means)
averageLogGamma2Mean <- mean(logGamma2Means)

# Print the average estimated means
print(paste0("Average Estimated True Population Mean (Chi-squared):", averageChiSquaredMean)
```

```
[1] "Average Estimated True Population Mean (Chi-squared):3.99947494515278"
```

```
print(paste0("Average Estimated True Population Mean (Log-transformed Chi-squared):", averag
```

```
[1] "Average Estimated True Population Mean (Log-transformed Chi-squared):1.11562342964515"
```

```
print(paste0("Average Estimated True Population Mean (Gamma, shape=0.5, scale=8):", average
```

```
[1] "Average Estimated True Population Mean (Gamma, shape=0.5, scale=8):3.99976217261419"
```

```
print(paste0("Average Estimated True Population Mean (Log-transformed Gamma, shape=0.5, scal
```

```
[1] "Average Estimated True Population Mean (Log-transformed Gamma, shape=0.5, scale=8):0.1151
```

```
print(paste0("Average Estimated True Population Mean (Gamma, shape=0.5, scale=21.75):", aver
```

```
[1] "Average Estimated True Population Mean (Gamma, shape=0.5, scale=21.75):10.879679816176"
```

```
print(paste0("Average Estimated True Population Mean (Log-transformed Gamma, shape=0.5, scal
```

```
[1] "Average Estimated True Population Mean (Log-transformed Gamma, shape=0.5, scale=21.75):1.
```

Part D

The high rejection rates for log-transformed data suggest a substantial impact of the log transformation on the distribution. If we do not reject the hypothesis for log-transformed data, it implies, assuming normality, that the means on the log scale are not significantly different. Hence, it is crucial to highlight the need to understand how transformations influence hypothesis testing for meaningful conclusions.

Question 2

Part A

```r
# Given data
data <- matrix(c(18, 2, 37, 53), nrow = 2, byrow = TRUE)
colnames(data) <- c("Yes", "No")
rownames(data) <- c("Yes", "No")

print(data)
```

```
    Yes No
Yes  18  2
No   37 53
```

```r
# Total number of observations
totalObs <- 100

# P(Library Member)
probLibraryMember <- sum(data[,"Yes"]) / totalObs
print(paste0("P(Library Member) = ", probLibraryMember))
```

```
[1] "P(Library Member) = 0.55"
```

```r
# P(Vegetarian | Non-Library Member)
probVegetarianNonLibrary <- data["Yes", "No"] / sum(data[,"No" ])
print(paste0("P(Vegetarian | Non-Library Member) = ", probVegetarianNonLibrary))
```

```
[1] "P(Vegetarian | Non-Library Member) = 0.0363636363636364"
```

```r
# P(Vegetarian)
probVegetarian <- sum(data["Yes", ]) / totalObs
print(paste0("P(Vegetarian) = ", probVegetarian))
```

```
[1] "P(Vegetarian) = 0.2"
```

Part B

```r
# Given data
data <- matrix(c(5, 15, 95, 85), nrow = 2, byrow = TRUE)
colnames(data) <- c("Yes", "No")
rownames(data) <- c("Yes", "No")

# Total number of observations
totalObs <- 200

# P(Marathon Runner)
probMarathonRunner <- sum(data["Yes", ]) / totalObs
print(paste0("P(Marathon Runner) = ", probMarathonRunner))
```

```
[1] "P(Marathon Runner) = 0.1"
```

```r
# P(Smoker)
probSmoker <- sum(data[, "Yes"]) / totalObs
print(paste0("P(Smoker) = ", probSmoker))
```

```
[1] "P(Smoker) = 0.5"
```

```r
# P(Smoker | Marathon Runner)
probSmokerGivenMarathon <- data["Yes", "Yes"] / sum(data[,"Yes"])
print(paste0("P(Smoker | Marathon Runner) = ", probSmokerGivenMarathon))
```

```
[1] "P(Smoker | Marathon Runner) = 0.05"
```

```r
# P(Marathon Runner | Non-smoker)
probMarathonGivenNonSmoker <- data["Yes", "No"] / sum(data["Yes",])
print(paste0("P(Marathon Runner | Non-smoker) = ", probMarathonGivenNonSmoker))
```

```
[1] "P(Marathon Runner | Non-smoker) = 0.75"
```

Part C

```r
# Given data
data <- matrix(c(2, 98, 8, 92), nrow = 2, byrow = TRUE)
colnames(data) <- c("Yes", "No")
rownames(data) <- c("Yes", "No")

# Total number of observations
totalObs <- 200

# P(Marathon Runner)
probMarathonRunner <- sum(data["Yes", ]) / totalObs
print(paste0("P(Marathon Runner) = ", probMarathonRunner))
```

```
[1] "P(Marathon Runner) = 0.5"
```

```r
# P(Smoker)
probSmoker <- sum(data[, "Yes"]) / totalObs
print(paste0("P(Smoker) = ", probSmoker))
```

```
[1] "P(Smoker) = 0.05"
```

```r
# P(Smoker | Marathon Runner)
probSmokerGivenMarathon <- data["Yes", "Yes"] / sum(data[,"Yes"])
print(paste0("P(Smoker | Marathon Runner) = ", probSmokerGivenMarathon))
```

```
[1] "P(Smoker | Marathon Runner) = 0.2"
```

```r
# P(Marathon Runner | Non-smoker)
probMarathonGivenNonSmoker <- data["Yes", "No"] / sum(data["No",])
print(paste0("P(Marathon Runner | Non-smoker) = ", probMarathonGivenNonSmoker))
```

```
[1] "P(Marathon Runner | Non-smoker) = 0.98"
```

Part D

```r
# Given data
data <- matrix(c(5, 25, 15, 55), nrow = 2, byrow = TRUE)
colnames(data) <- c("Undergrad", "Grad")
rownames(data) <- c("Ski", "Snowboard")

# Total number of observations
totalObs <- 100

# P(Undergrad)
probUndergrad <- sum(data[, "Undergrad"]) / totalObs
print(paste0("P(Undergrad) = ", probUndergrad))
```

```
[1] "P(Undergrad) = 0.2"
```

```r
# P(Skier)
probSkier <- sum(data["Ski", ]) / totalObs
print(paste0("P(Skier) = ", probSkier))
```

```
[1] "P(Skier) = 0.3"
```

```r
# P(Skier | Grad)
probSkierGivenGrad <- data["Ski", "Grad"] / sum(data["Ski",])
print(paste0("P(Skier | Grad) = ", probSkierGivenGrad))
```

```
[1] "P(Skier | Grad) = 0.833333333333333"
```

```r
# P(Grad | Snowboarder)
probGradGivenSnowboarder <- data["Snowboard", "Grad"] / sum(data[,"Grad"])
print(paste0("P(Grad | Snowboarder) = ", probGradGivenSnowboarder))
```

```
[1] "P(Grad | Snowboarder) = 0.6875"
```

```r
# Estimate population proportion of OSU students who prefer skiing
estimatedPropSkiing <- sum(data["Ski", ]) / (totalObs * 0.8)
print(paste0("Estimated Population Proportion of Skiing Preference = ", estimatedPropSkiing))
```

```
[1] "Estimated Population Proportion of Skiing Preference = 0.375"
```

Question 3

In the Fisher's test, the p-value is 0.65, showing we can't really say my friend's good at telling coffee types apart. The odds ratio is 2.23, but the confidence interval is wide, so we're not very sure.

For the chi-squared test, the p-value is 0.65 too, basically saying my friend's guesses aren't different from random.

Both tests suggest my friend's not reliably telling the difference between instant and drip coffee.

Part A

```r
# Given data
observed_table <- matrix(c(7, 5, 3, 5), nrow = 2, byrow = TRUE)
colnames(observed_table) <- c("Instant", "Freshly Ground Drip")
rownames(observed_table) <- c("Instant", "Fresh Ground Drip")

# Fisher's exact test
fisher_test_result <- fisher.test(observed_table)
print(fisher_test_result)
```

```
    Fisher's Exact Test for Count Data

data:  observed_table
p-value = 0.6499
alternative hypothesis: true odds ratio is not equal to 1
95 percent confidence interval:
  0.2725276 21.9391352
sample estimates:
odds ratio
  2.234096
```

Part B

```r
# Pearson's chi-squared test
chi_squared_test_result <- chisq.test(observed_table)
print(chi_squared_test_result)
```

```
    Pearson's Chi-squared test with Yates' continuity correction

data:  observed_table
X-squared = 0.20833, df = 1, p-value = 0.6481
```