Assignment 1

Kshitij Kavimandan, Pablo Alves, Pooja Mangal group 015

25 February 2024

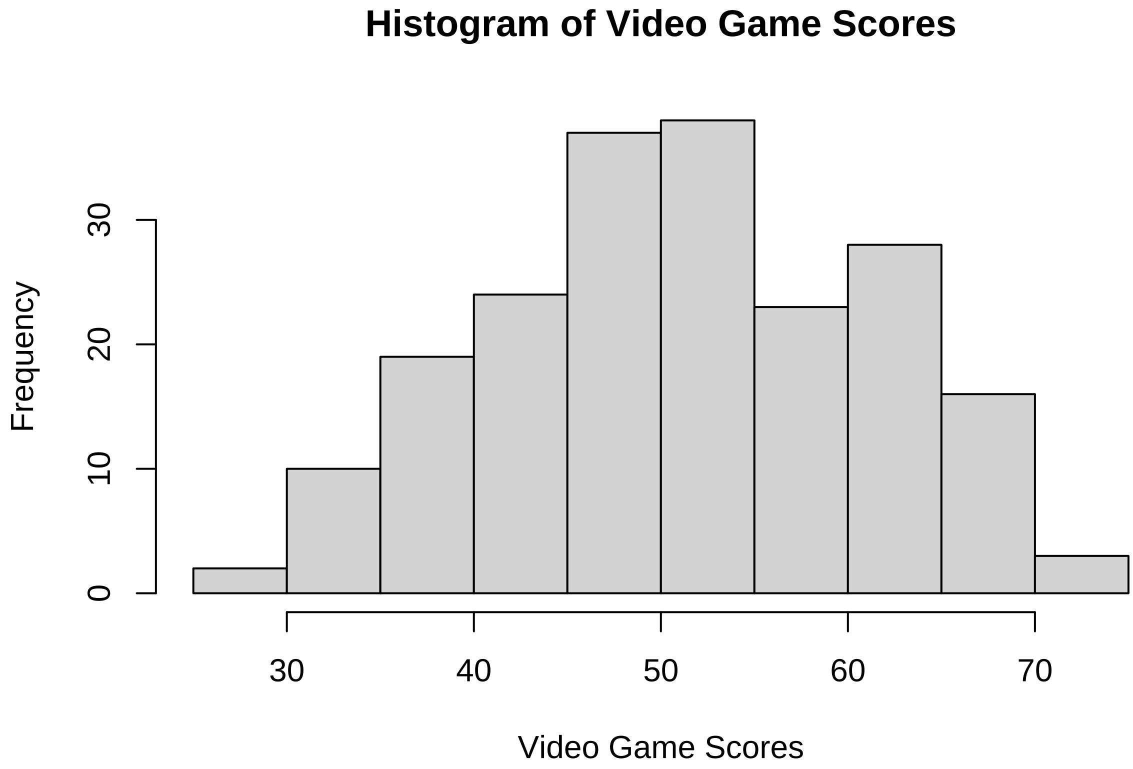
Exercise 1. Ice cream

a) Relevant plots and normality assessment:

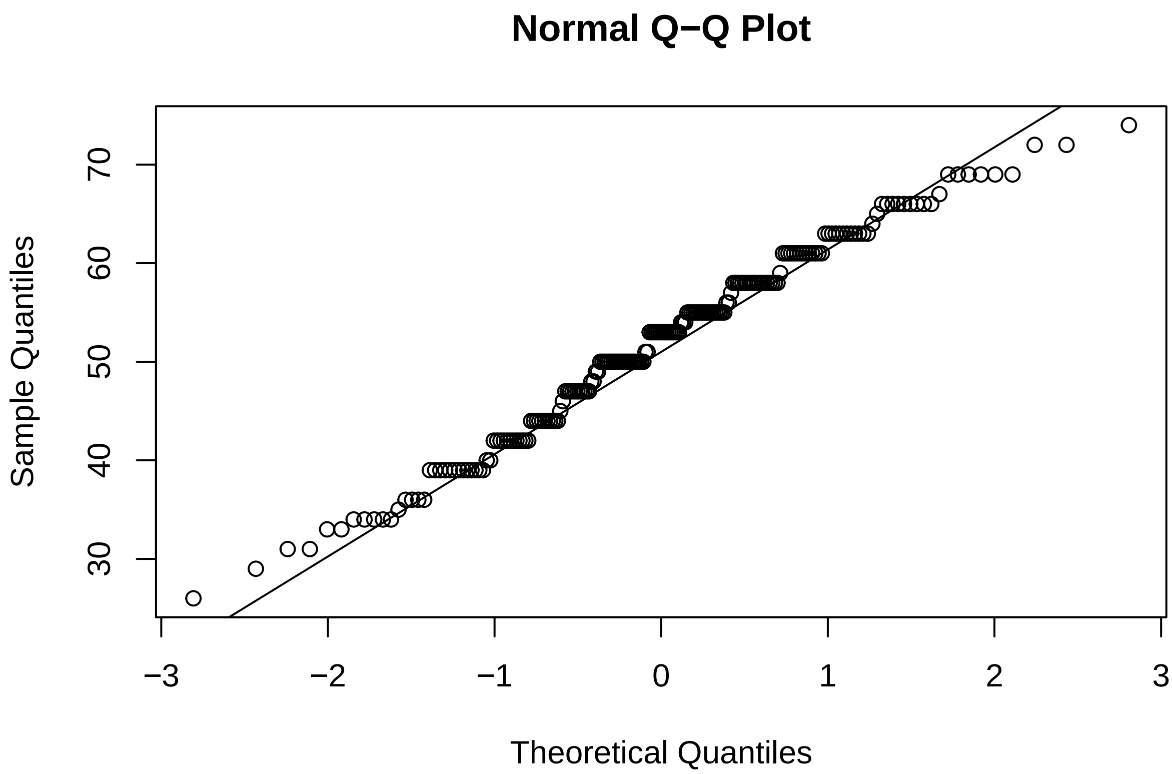
We will create a histogram and a Q-Q plot of the sample of video game scores to visually assess normality.

```r
# Load the data
data <- read.csv("Ice_cream.csv")

# Plot histogram
hist(data$video, main="Histogram of Video Game Scores", xlab="Video Game Scores")
```



```r
# Plot Q-Q plot
qqnorm(data$video)
qqline(data$video)
```



After examining the plots, we can comment on the normality of the video game scores.

Constructing a bounded 97%-CI for μ:

```r
# Calculate the mean and standard deviation of the sample
sample_mean <- mean(data$video)
sample_sd <- sd(data$video)

# Calculate the margin of error
margin_of_error <- qt(0.985, df = length(data$video) - 1) * (sample_sd / sqrt(length(data$vide

# Construct the CI
lower_bound <- sample_mean - margin_of_error
upper_bound <- sample_mean + margin_of_error
```

Sample size needed for a 97%-CI with a maximum length of 3:

```r
# Calculate the required sample size
required_sample_size <- (qt(0.985, df = 199) * sample_sd / 3) ^ 2
```

Bootstrap 97%-CI for μ:

```r
# Bootstrap function
bootstrap_mean <- function(data, sample_size) {
  bootstrap_samples <- replicate(10000, mean(sample(data, sample_size, replace = TRUE)))
  ci_lower <- quantile(bootstrap_samples, 0.015)
  ci_upper <- quantile(bootstrap_samples, 0.985)
  return(c(ci_lower, ci_upper))
}

# Apply bootstrap function
bootstrap_ci <- bootstrap_mean(data$video, length(data$video))
```

b) T-test to verify mean score:

```r
# Perform t-test
t_test <- t.test(data$video, mu = 50, alternative = "greater")

# Print t-test results and explanation of CI
print(t_test)
```

```
##
##  One Sample t-test
##
## data:  data$video
## t = 2.6425, df = 199, p-value = 0.004442
## alternative hypothesis: true mean is greater than 50
## 95 percent confidence interval:
##  50.69305      Inf
## sample estimates:
## mean of x
##     51.85
```

Explanation of CI: The CI in the output represents the 95% confidence interval for the difference between the sample mean and the hypothesized mean (μ_0). If the entire interval is above μ_0, it suggests evidence for the alternative hypothesis.

c) Sign test and test based on ranks:

```r
# Perform sign test
sign_test <- binom.test(sum(data$video > 50), length(data$video), p = 0.5, alternative = "grea

# Perform Wilcoxon signed-rank test
wilcox_test <- wilcox.test(data$video, mu = 50, alternative = "greater")

# Test for fraction of scores less than 42
binom_test <- binom.test(sum(data$video < 42), length(data$video), p = 0.25, alternative = "le
```

d) Bootstrap test with test statistic T:

```r
# Bootstrap test function
bootstrap_test <- function(data, num_samples) {
  bootstrap_samples <- replicate(num_samples, min(sample(data, length(data), replace = TRUE)))
  p_value <- mean(bootstrap_samples < min(data) | bootstrap_samples > 100)
  return(p_value)
}

# Apply bootstrap test
p_value <- bootstrap_test(data$video, 10000)

# Apply Kolmogorov-Smirnov test
ks_test <- ks.test(data$video, "pnorm", mean = 50, sd = 10)
```

```
## Warning in ks.test.default(data$video, "pnorm", mean = 50, sd = 10): ties
## should not be present for the Kolmogorov-Smirnov test
```

e) Tests for male and female students:

```r
# Separate scores for male and female students
male_scores <- data$video[data$female == 0]
female_scores <- data$video[data$female == 1]

# Perform two-sample t-test
t_test_gender <- t.test(male_scores, female_scores, alternative = "greater")

# Perform Mann-Whitney test
mannwhitney_test <- wilcox.test(male_scores, female_scores, alternative = "greater")

# Perform Kolmogorov-Smirnov test
ks_test_gender <- ks.test(male_scores, female_scores)
```

f) Correlation and comparison between video game and puzzle scores:

```r
# Investigate correlation
correlation <- cor(data$video, data$puzzle)

# Test if puzzle scores are higher than video game scores
wilcox_test_puzzle <- wilcox.test(data$puzzle, data$video, alternative = "greater")
```

Exercise 2. Hemoglobin in trout

a) R-code for randomization process:

```r
# Load the data
hemoglobin_data <- read.table("hemoglobin.txt", header = TRUE)

# Create a data frame to store the combinations of rate and method
combinations <- expand.grid(rate = c(1, 2, 3, 4), method = c("A", "B"))

# Randomly assign 80 fishes to the combinations
set.seed(123)  # Set seed for reproducibility
hemoglobin_data$fish <- sample(rep(1:80, length.out = nrow(hemoglobin_data)), replace = FALSE)

# Merge the combinations with the hemoglobin data
final_data <- merge(combinations, hemoglobin_data, by = "rate", all = TRUE)
```

b) Two-way ANOVA:

```r
# Perform two-way ANOVA
anova_result <- aov(hemoglobin ~ rate * method, data = hemoglobin_data)

# Print ANOVA table
print(summary(anova_result))
```

```
##             Df Sum Sq Mean Sq F value   Pr(>F)    
## rate         1  27.93  27.931  11.933 0.000905 ***
## method       1   2.42   2.415   1.032 0.312963    
## rate:method  1   1.24   1.243   0.531 0.468373    
## Residuals   76 177.90   2.341                     
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

c) Influence of factors and combination yielding highest hemoglobin:

```r
# Calculate means for rate and method
mean_hemoglobin <- aggregate(hemoglobin ~ rate + method, data = hemoglobin_data, FUN = mean)

# Find combination yielding highest hemoglobin
```

```r
max_hemoglobin <- mean_hemoglobin[which.max(mean_hemoglobin$hemoglobin), ]

# Estimate mean hemoglobin value for rate 3 by using method A
mean_hemoglobin_rate3_methodA <- mean_hemoglobin$hemoglobin[which(mean_hemoglobin$rate == 3 & r

# Estimate mean hemoglobin value for each rate
mean_hemoglobin_rate <- aggregate(hemoglobin ~ rate, data = hemoglobin_data, FUN = mean)
```

d) One-way ANOVA:

```r
# Perform one-way ANOVA ignoring the variable method
anova_result_rate <- aov(hemoglobin ~ rate, data = hemoglobin_data)

# Print ANOVA table
print(summary(anova_result_rate))
```

```
##             Df Sum Sq Mean Sq F value   Pr(>F)    
## rate         1  27.93  27.931      12 0.000867 ***
## Residuals   78 181.55   2.328                     
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

e) Kruskal-Wallis test:

```r
# Perform Kruskal-Wallis test
kruskal_test_result <- kruskal.test(hemoglobin ~ rate, data = hemoglobin_data)

# Print Kruskal-Wallis test result
print(kruskal_test_result)
```

```
## 
##  Kruskal-Wallis rank sum test
## 
## data:  hemoglobin by rate
## Kruskal-Wallis chi-squared = 34.224, df = 3, p-value = 1.777e-07
```

Exercise 3. Sour cream

a) Analyzing the data in a three-way experiment without interactions:

```r
# Load the data
cream_data <- read.table("cream.txt", header = TRUE)

# Fit the model without interactions
model <- lm(acidity ~ starter + batch + position, data = cream_data)
```

```
# Summary of the model
summary(model)
```

```
## 
## Call:
## lm(formula = acidity ~ starter + batch + position, data = cream_data)
## 
## Residuals:
##     Min      1Q  Median      3Q     Max 
## -2.9442 -1.2164 -0.0976  0.8296  3.7508 
## 
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)    
## (Intercept)   7.3330     1.3373   5.484 1.93e-05 ***
## starter       0.1992     0.2483   0.802    0.431    
## batch         0.3116     0.2483   1.255    0.223    
## position     -0.0674     0.2483  -0.271    0.789    
## ---
## Signif. codes:  0 ’***’ 0.001 ’**’ 0.01 ’*’ 0.05 ’.’ 0.1 ’ ’ 1
## 
## Residual standard error: 1.756 on 21 degrees of freedom
## Multiple R-squared:  0.09839,    Adjusted R-squared:  -0.03041 
## F-statistic: 0.7639 on 3 and 21 DF,  p-value: 0.527
```

From the summary output, we can examine the coefficients and corresponding p-values for the effect of each factor. The p-value associated with the coefficient for starter 1 compared to starter 2 indicates whether there is a significant difference between the effects of these two starters on acidity. If the p-value is less than the chosen significance level (e.g., 0.05), we consider the difference to be statistically significant.

b) Removing insignificant block variables and performing ANOVA:

```
# Fit the model with significant factors only
model <- lm(acidity ~ starter, data = cream_data)

# Perform ANOVA for the fixed effects model
anova_result <- anova(model)

# Print ANOVA table
print(anova_result)
```

```
## Analysis of Variance Table
## 
## Response: acidity
##           Df Sum Sq Mean Sq F value Pr(>F)
## starter    1  1.984  1.9840  0.6535 0.4272
## Residuals 23 69.830  3.0361               
```

c) Applying the Friedman test:

```r
# Convert "acidity" column to numeric
cream_data$acidity <- as.numeric(as.character(cream_data$acidity))

# Conduct the Friedman test
friedman_test_result <- friedman.test(cream_data$acidity, cream_data$starter, cream_data$batch

# Print the Friedman test result
print(friedman_test_result)
```

```
##
##  Friedman rank sum test
##
## data:  cream_data$acidity, cream_data$starter and cream_data$batch
## Friedman chi-squared = 13.212, df = 4, p-value = 0.01028
```

d) Performing a mixed effects analysis:

```r
# Load the lme4 package
library(lme4)
```

```
## Loading required package: Matrix
```

```r
# Fit the mixed effects model
mixed_model <- lmer(acidity ~ (1|batch) + (1|position), data = cream_data)
```

```
## boundary (singular) fit: see help('isSingular')
```

```r
# Summary of the mixed effects model
summary(mixed_model)
```

```
## Linear mixed model fit by REML ['lmerMod']
## Formula: acidity ~ (1 | batch) + (1 | position)
##    Data: cream_data
##
## REML criterion at convergence: 97
##
## Scaled residuals:
##     Min      1Q  Median      3Q     Max
## -1.6541 -0.4914 -0.1351  0.3222  2.0694
##
## Random effects:
##  Groups   Name        Variance Std.Dev.
##  batch    (Intercept) 0.4085   0.6392
##  position (Intercept) 0.0000   0.0000
##  Residual             2.6518   1.6284
## Number of obs: 25, groups:  batch, 5; position, 5
##
## Fixed effects:
##             Estimate Std. Error t value
## (Intercept)   8.6632     0.4333   19.99
## optimizer (nloptwrap) convergence code: 0 (OK)
## boundary (singular) fit: see help('isSingular')
```

In the mixed effects model, we model the block variables (batch and position) as random effects. We compare the results from the mixed effects model to those from the fixed effects model to assess the impact of modeling block variables as random effects. This comparison helps evaluate the variability between batches and positions and provides insight into whether the fixed effects model adequately captures the underlying structure of the data.