Homwork1

Homework01Gr6

2025-02-03

NOTE: only those who contributes and fully participates in the work will get credit

Scribe:

Moderator:

All contributors:

Q1:

The barista at "t-test espresso'' is told that the optimal serving temperature for coffee is 180 F. Five temperatures are taken of the served coffee: 175, 185, 170, 184, and 175 degrees.

TASK: Find a 90% confidence interval of the form $(-\infty, b)$ for the mean temperature.(one side CI)

DATA:

```r
temp <- c(175, 185, 170, 184, 175)
n <- length(temp)
alpha <- 0.1
x_bar <- mean(temp)
s <- sd(temp)
```

Part 1

Since the number of samples is not large enough, we calculate the one-sided CI using t-distribution: Type the formula below

$$\mu \pm t_{\alpha,n-1} \cdot \frac{s}{\sqrt{n}}$$

Part2

Compute one sided CI below

The corresponding t score is:

```r
#type codes here
# Given data
temp <- c(175, 185, 170, 184, 175)
n <- length(temp)
alpha <- 0.1  # 90% confidence means alpha = 0.1 for one-sided
x_bar <- mean(temp)
s <- sd(temp)

# Degrees of freedom
df <- n - 1
```

```r
# Compute the critical t-score (one-sided)
t_score <- qt(1 - alpha, df)

# Calculate the one-sided confidence interval
ci_upper <- x_bar + t_score * (s / sqrt(n))

# Output the results
cat("test score")
```

```
## test score
```

```r
t_score
```

```
## [1] 1.533206
```

```r
cat("one sided confidence interval")
```

```
## one sided confidence interval
```

```r
ci_upper
```

```
## [1] 182.2278
```

Part 3

Alternatively, using t.test with alt="less" will give this type of one-sided confidence interval:

```r
# Given data
temp <- c(175, 185, 170, 184, 175)

# Perform the one-sided t-test with alternative = "less"
t_test_result <- t.test(temp, alternative = "less", conf.level = 0.90)

# Display the result
t_test_result$conf.int
```

```
## [1]     -Inf 182.2278
## attr(,"conf.level")
## [1] 0.9
```

After class activities (this part is HW2 from the past)

Verzani BOOK, Problem 3.16, 3.17, 3.31, 8.6, 8.8, 8.12, 8.19, Devore BOOK : section 7.2 problem 16; Sec 7.3: problem 32

##Q2 Verzani Problem 3.16

```r
#type codes here
# Load the dataset (assuming the "UsingR" package is installed)
library(UsingR)
```

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

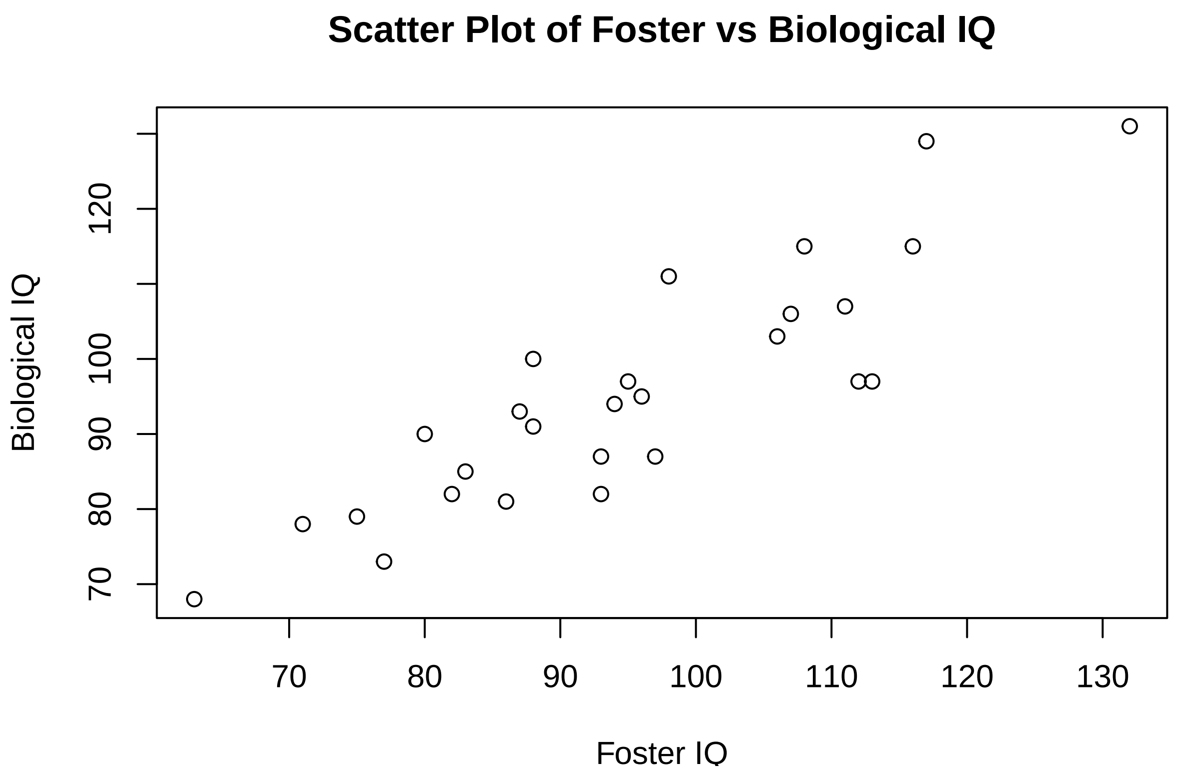
```
## Loading required package: HistData
```

```
## Loading required package: Hmisc
```

```
##
## Attaching package: 'Hmisc'
```

```
## The following objects are masked from 'package:base':
##
##     format.pval, units
```

```r
# Load the twins dataset
data(twins)

# Create a scatter plot
plot(twins$Foster, twins$Biological,
     xlab = "Foster IQ", ylab = "Biological IQ",
     main = "Scatter Plot of Foster vs Biological IQ")
```




```r
# Calculate the Pearson correlation coefficient
pearson_corr <- cor(twins$Foster, twins$Biological, method = "pearson")
cat("Pearson correlation:", pearson_corr, "\n")
```

```
## Pearson correlation: 0.8819877
```

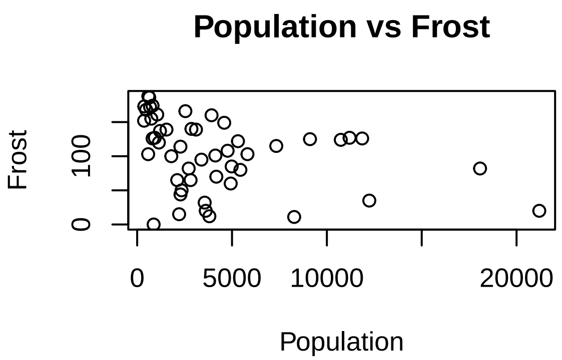
```r
# Calculate the Spearman correlation coefficient
spearman_corr <- cor(twins$Foster, twins$Biological, method = "spearman")
cat("Spearman correlation:", spearman_corr, "\n")
```

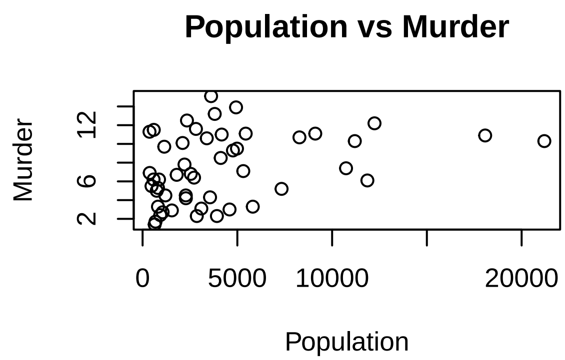
```
## Spearman correlation: 0.8858324
```

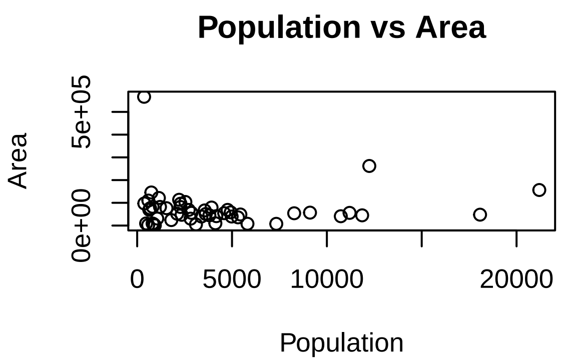
##Q3 Verzani Problem 3.17

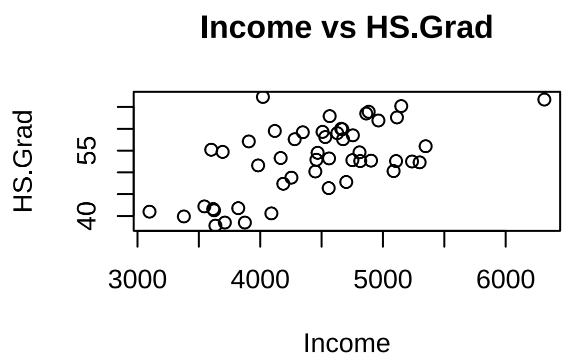
```r
# Convert the state.x77 data set into a data frame
x77 <- data.frame(state.x77)

# Create scatter plots for the specified pairs
par(mfrow = c(2, 2))  # Arrange the plots in a 2x2 grid

# Scatter plot of Population vs Frost
```

```r
plot(x77$Population, x77$Frost,
     xlab = "Population", ylab = "Frost",
     main = "Population vs Frost")

# Scatter plot of Population vs Murder
plot(x77$Population, x77$Murder,
     xlab = "Population", ylab = "Murder",
     main = "Population vs Murder")

# Scatter plot of Population vs Area
plot(x77$Population, x77$Area,
     xlab = "Population", ylab = "Area",
     main = "Population vs Area")

# Scatter plot of Income vs HS.Grad
plot(x77$Income, x77$HS.Grad,
     xlab = "Income", ylab = "HS.Grad",
     main = "Income vs HS.Grad")
```









##Check linearity in the scatter plots??

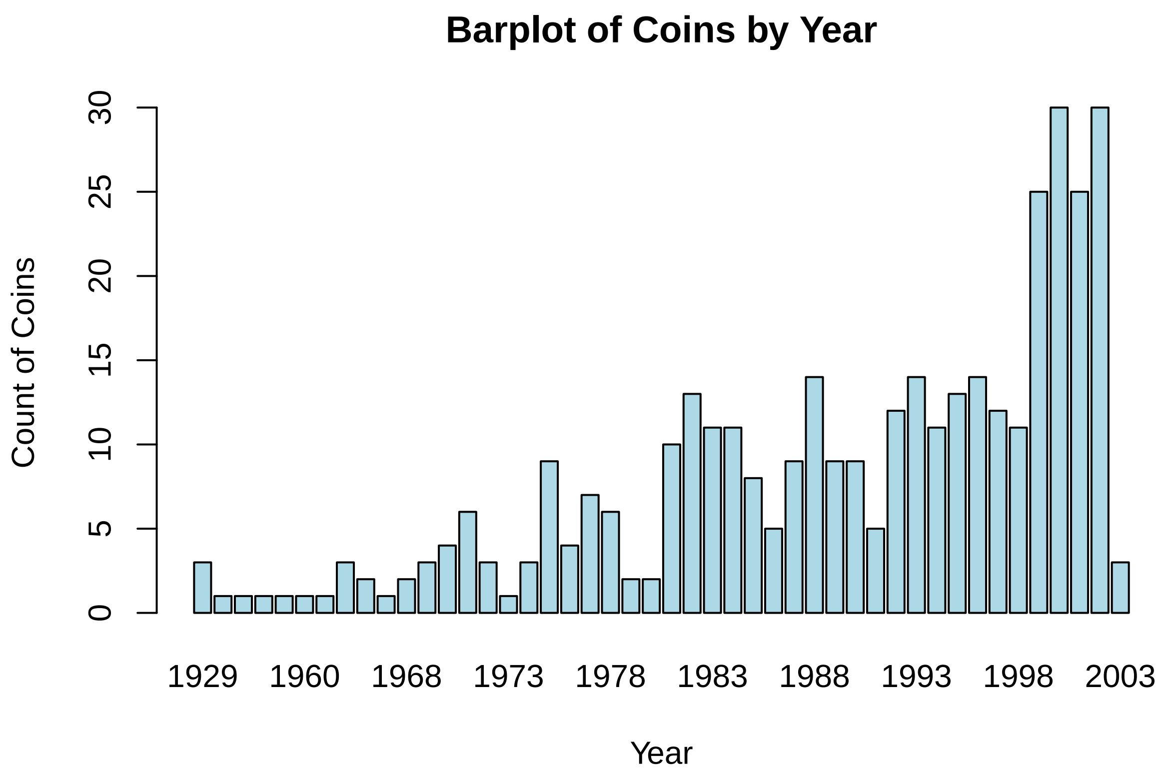
##Q4 Verzani Problem 3.31

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

# Load the coins data set
data(coins)

# 1. How much money is in the change bin?
```

```r
# Assuming the 'coins' dataset contains a 'count' and 'value' columns:
money_in_bin <- sum(coins$count * coins$value)
cat("Total money in the change bin: $", money_in_bin, "\n")
```

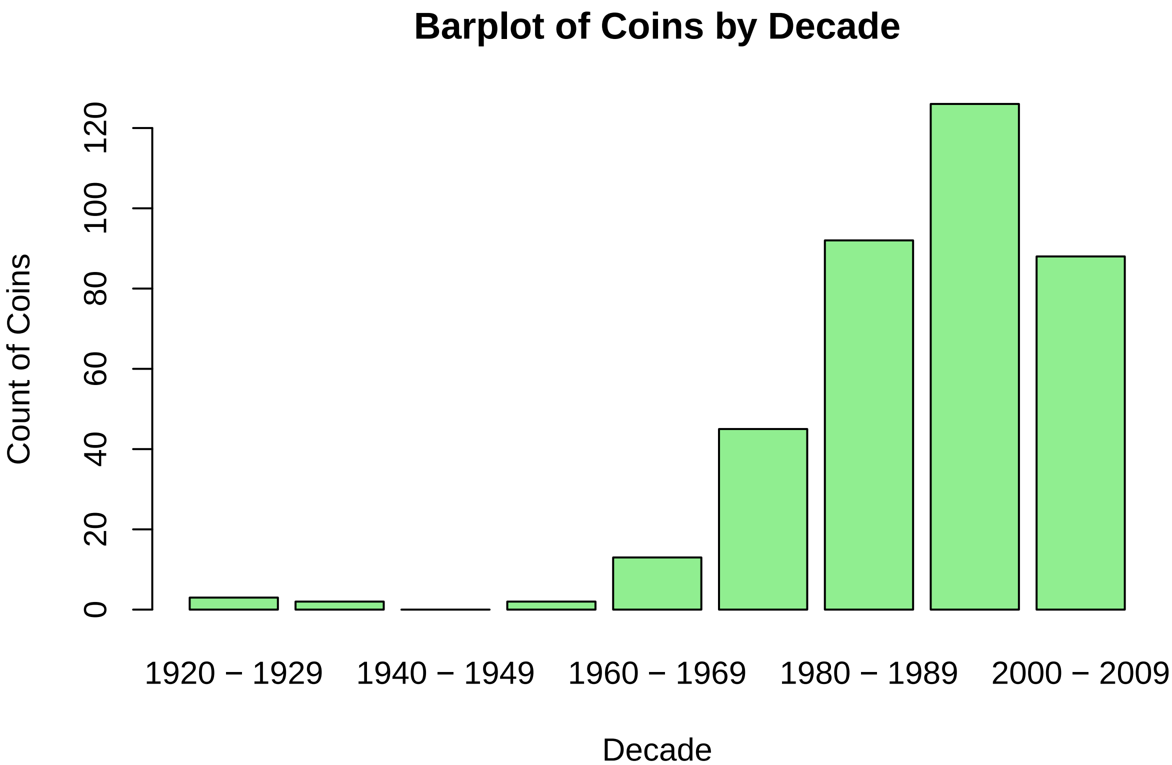
```
## Total money in the change bin: $ 0
```

```r
# 2. Make a barplot of the years. Is there a trend?
barplot(table(coins$year),
        xlab = "Year",
        ylab = "Count of Coins",
        main = "Barplot of Coins by Year",
        col = "lightblue")
```




```r
# 3. Use cut to construct a barplot by decade
# Create a new variable grouping years into decades

decade_labels <- seq(from = floor(min(coins$year) / 10) * 10,
                     to = ceiling(max(coins$year) / 10) * 10 - 1,
                     by = 10)

coins$decade <- cut(coins$year,
                    breaks = c(decade_labels, Inf),  # Inf is used to include the last decade
                    labels = paste(decade_labels, "-", decade_labels + 9),
                    right = FALSE)

# Create a barplot of coins by decade
barplot(table(coins$decade),
        xlab = "Decade",
        ylab = "Count of Coins",
        main = "Barplot of Coins by Decade",
```

```r
        col = "lightgreen")
```




```r
# 4. Make a contingency table of year and value
contingency_table <- table(coins$year, coins$value)
cat("Contingency table of Year and Value:\n")
```

```
## Contingency table of Year and Value:
```

```r
print(contingency_table)
```

```
##
##         0.01 0.05 0.1 0.25
##   1929     2    1   0    0
##   1936     0    0   1    0
##   1939     0    0   1    0
##   1955     0    0   0    1
##   1959     1    0   0    0
##   1960     1    0   0    0
##   1964     1    0   0    0
##   1965     2    1   0    0
##   1966     1    1   0    0
##   1967     0    1   0    0
##   1968     1    0   0    1
##   1969     2    0   1    0
##   1970     1    1   0    2
##   1971     3    0   2    1
##   1972     2    0   0    1
##   1973     1    0   0    0
##   1974     1    1   1    0
##   1975     4    1   0    4
```

```
## 1976 2 1 0 1
## 1977 4 2 0 1
## 1978 5 1 0 0
## 1979 1 1 0 0
## 1980 1 1 0 0
## 1981 7 1 2 0
## 1982 9 1 2 1
## 1983 7 3 1 0
## 1984 7 3 0 1
## 1985 2 4 1 1
## 1986 3 0 1 1
## 1987 7 1 1 0
## 1988 7 3 2 2
## 1989 4 0 3 2
## 1990 3 3 1 2
## 1991 2 1 1 1
## 1992 6 0 0 6
## 1993 8 3 1 2
## 1994 7 1 2 1
## 1995 6 2 3 2
## 1996 7 3 0 4
## 1997 6 2 1 3
## 1998 5 1 0 5
## 1999 16 2 6 1
## 2000 15 4 3 8
## 2001 13 5 0 7
## 2002 19 3 5 3
## 2003 1 0 0 2
```

```r
# Interpretation of the contingency table:
cat ("Value of 0.01 is heavily concentrated near 1999, 2000,2001 and 2002")
```

```
## Value of 0.01 is heavily concentrated near 1999, 2000,2001 and 2002
```

##Q5 Verzani Problem 8.6

```r
# Given data
n <- 100    # Total number of students surveyed
x <- 5      # Number of left-handed students

# Sample proportion
p_hat <- x / n
p_hat
```

```
## [1] 0.05
```

```r
# Standard error of the sample proportion
SE <- sqrt(p_hat * (1 - p_hat) / n)
SE
```

```
## [1] 0.02179449
```

```r
# Critical value for a 95% confidence interval
z_alpha <- 1.96

# Confidence interval calculation
CI_lower <- p_hat - z_alpha * SE
```

```r
CI_upper <- p_hat + z_alpha * SE
#Lower bound of confidence interval
CI_lower
```

```
## [1] 0.00728279
```

```r
#Upper bound of confidence interval
CI_upper
```

```
## [1] 0.09271721
```

##Q6 Verzani Problem 8.8

```r
# Given values
z_alpha <- 1.96  # critical value for 95% confidence
p_hat <- 0.54     # sample proportion
margin_error <- 0.02  # margin of error

# Sample size calculation
n <- (z_alpha^2 * p_hat * (1 - p_hat)) / margin_error^2
n
```

```
## [1] 2385.634
```

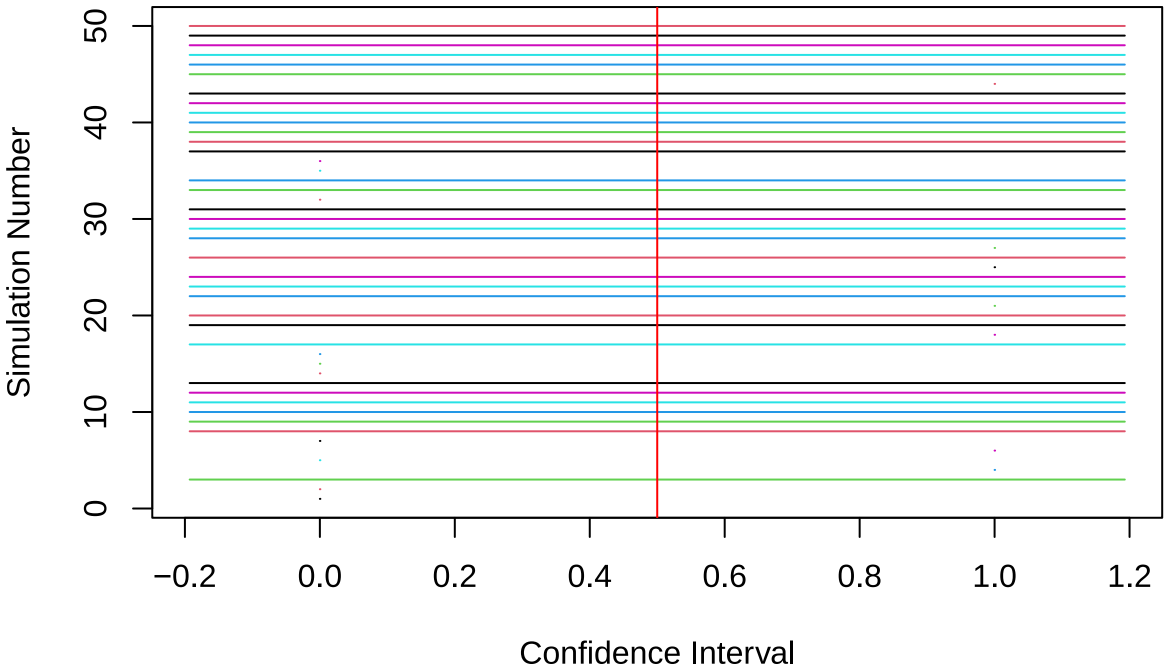
##Q7 Verzani Problem 8.12

```r
# Set parameters for the simulation
M <- 50  # Number of simulations
n <- 2   # Number of coin tosses per trial
p <- 0.5 # True proportion
alpha <- 0.05 # For 95% confidence interval

# Critical value for a 95% confidence interval
zstar <- qnorm(1 - alpha / 2)

# Generate M random samples of size n with probability p (binomial distribution)
phat <- rbinom(M, n, p) / n

# Compute the standard error for each sample proportion
SE <- sqrt(phat * (1 - phat) / n)

# Compute the confidence intervals
lower_bound <- phat - zstar * SE
upper_bound <- phat + zstar * SE

# Check how many of the confidence intervals contain the true proportion p = 0.5
contained <- sum(lower_bound < p & p < upper_bound)

# Calculate the percentage of intervals that contain p
percentage_contained <- (contained / M) * 100

# Output the result
percentage_contained
```

```
## [1] 66
```

```r
# Optional: Plot the confidence intervals
matplot(rbind(lower_bound, upper_bound),
```

```r
        rbind(1:M, 1:M), type = "l", lty = 1,
        xlab = "Confidence Interval", ylab = "Simulation Number")
abline(v = p, col = "red")  # Vertical line indicating true proportion p = 0.5
```



##Q7 Verzani Problem 8.19

```r
# Load necessary libraries
library(HistData)

# Load the Macdonell dataset
data(Macdonell)

# Expand the data based on frequency
finger <- with(Macdonell, rep(finger, frequency))
head(finger)  # Preview the expanded data
```

```
## [1] 10.0 10.3  9.9 10.2 10.2 10.3
```

```r
# Set the random seed for reproducibility
set.seed(123)

# Generate 750 samples of size 4
samples <- replicate(750, sample(finger, size = 4, replace = TRUE))

# Compute the sample means
sample_means <- apply(samples, 2, mean)

# Compute the 95% confidence interval using quantile
CI_quantile <- quantile(sample_means, c(0.025, 0.975))
CI_quantile
```

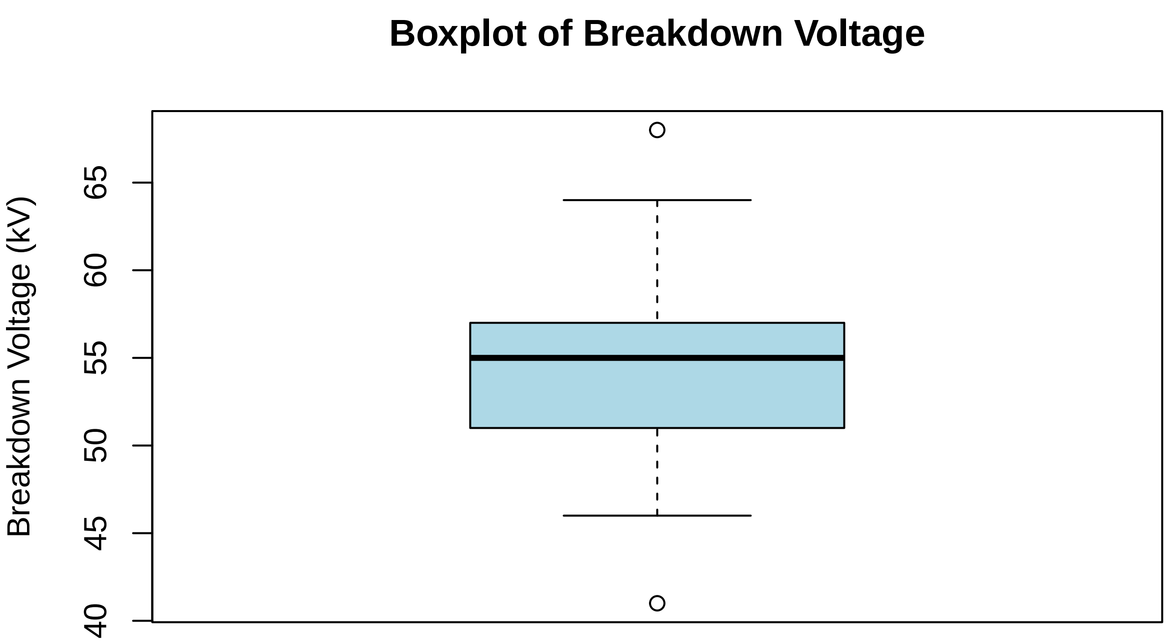
```
##   2.5%  97.5%
## 11.025 12.050
```

```r
# Select one sample of size 4
single_sample <- sample(finger, size = 4)

# Perform the t-test
t_test_result <- t.test(single_sample)

# Extract the confidence interval from the t-test result
CI_t_test <- t_test_result$conf.int
CI_t_test
```

```
## [1] 11.58632 12.06368
## attr(,"conf.level")
## [1] 0.95
```

##Devore 7.2 prob 16

```r
# Breakdown voltage data
voltage <- c(62, 50, 53, 57, 41, 53, 55, 61, 59, 64, 50, 53, 64, 62, 50, 68,
             54, 55, 57, 50, 55, 50, 56, 55, 46, 55, 53, 54, 52, 47, 47, 55,
             57, 48, 63, 57, 57, 55, 53, 59, 53, 52, 50, 55, 60, 50, 56, 58)
# Boxplot
boxplot(voltage, main="Boxplot of Breakdown Voltage", ylab="Breakdown Voltage (kV)", col="lightblue")
```




```r
# Sample statistics
n <- length(voltage)
x_bar <- mean(voltage)
s <- sd(voltage)

# Critical value for t-distribution with 95% confidence level
t_alpha <- qt(0.975, df=n-1)

# Margin of error
margin_error <- t_alpha * (s / sqrt(n))

# Confidence interval
```

```r
CI_lower <- x_bar - margin_error
CI_upper <- x_bar + margin_error

CI_lower
```

```
## [1] 53.1895
```

```r
CI_upper
```

```
## [1] 56.22716
```

```r
# Desired margin of error
E <- 1  # 1 kV for margin of error

# Sample size calculation
sample_size <- (t_alpha * s / E)^2
sample_size
```

```
## [1] 110.7284
```

##Devore 7.3 prob 32

```r
# Given data
x_bar <- 1584    # Sample mean
s <- 607         # Sample standard deviation
n <- 20          # Sample size

# Degrees of freedom
df <- n - 1

# Critical t value for 99% confidence level
t_alpha <- qt(0.995, df = df)

# Standard error of the mean
SE <- s / sqrt(n)

# Margin of error
margin_error <- t_alpha * SE

# Confidence interval
CI_lower <- x_bar - margin_error
CI_upper <- x_bar + margin_error

CI_lower
```

```
## [1] 1195.687
```

```r
CI_upper
```

```
## [1] 1972.313
```