Exercise 4 - Sample distribution and Central Limit Theorem

Konstantinos Vakalopoulos 12223236

2023-11-8

Contents

Task 1

Consider the 12 sample data points: 4.94 5.06 4.53 5.07 4.99 5.16 4.38 4.43 4.93 4.72 4.92 4.96

```
data.points <- c(4.94, 5.06, 4.53, 5.07, 4.99,
                 5.16, 4.38, 4.43, 4.93, 4.72,
                 4.92, 4.96)
```

Task 1.1

How many possible bootstrap samples are there, if each bootstrap sample has the same size as the original?

The possible bootstrap samples are n^n, n is equal to 12 (length(data.points))

Task 1.2

Compute the mean and the median of the original sample.

```
mean.orig <- mean(data.points)
median.orig <- median(data.points)

cat("The mean of the original sample is: ", mean.orig)
```

```
## The mean of the original sample is:  4.840833
```

```
cat("The median of the original sample is: ", median.orig)
```

```
## The median of the original sample is:  4.935
```

Task 1.3

Create 2000 bootstrap samples and compute their means.

First, we set a seed for reproducibility purposes (seed = 12223236 student ID) and then we use the sample() function with the parameter replace equals to true in order to perform sampling with replacement. We calculate the mean of the sample and afterwards the replicate() function is used in order to perform the same procedure 2000 times.

```
set.seed(12223236)
mean.samples <- replicate(2000, mean(sample(data.points, replace=TRUE)))
```

Task 1.3.1

Compute the mean on the first 20 bootstrap means.

```
mean.sample.20 <- mean(mean.samples[1:20])
cat("The mean of the first 20 bootstrap means is: ", mean.sample.20)
```

```
## The mean of the first 20 bootstrap means is:  4.819667
```

Task 1.3.2

Compute the mean of the first 200 bootstrap means.

```r
mean.sample.200 <- mean(mean.samples[1:200])
cat("The mean of the first 200 bootstrap means is: ", mean.sample.200)
```

```
## The mean of the first 200 bootstrap means is:  4.838292
```

Task 1.3.3

Compute the mean based on all 2000 bootstrap means.

```r
mean.sample.2000 <- mean(mean.samples[1:2000])
cat("The mean of the first 2000 bootstrap means is: ", mean.sample.2000)
```

```
## The mean of the first 2000 bootstrap means is:  4.837033
```

Task 1.3.4

Visualize the distribution all the different bootstrap means to the sample mean. Does the Central Limit Theorem kick in?

The library ggplot2 was initialized for this task. The following plot shows the distribution all the different bootstrap means to the sample mean.

```r
library(ggplot2)
```

```r
ggplot(data.frame(Means = mean.samples), aes(x = Means)) +
  geom_histogram(binwidth = 0.01, fill = "blue", color = "black") +
  geom_vline(xintercept = mean.orig, color = "red", linetype = "dashed", linewidth = 1) +
  labs(title = "Distribution of Bootstrap Sample Means vs. Sample Mean",
       x = "Sample Means",
       y = "Frequency") +
  theme_minimal()
```

According to the plot, the sample means closely follow a normal distribution, regardless of the underlying distribution of the individual sample data, which indicates that the Central Limit Theorem occurs.

Task 1.3.5

Based on the three different bootstrap sample lengths in 3. compute the corresponding 0.025 and 0.975 quantiles. Compare the three resulting intervals against each other and the "true" confidence interval of the mean under the assumption of normality. (Use for example the function t.test to obtain the 95% percent CI based on asymptotic considerations for the mean.)

In the next code chunk, based on three different bootstrap sample lengths in 3, the corresponding 0.025 and 0.975 quantiles are computed. The results are presented in a table below.

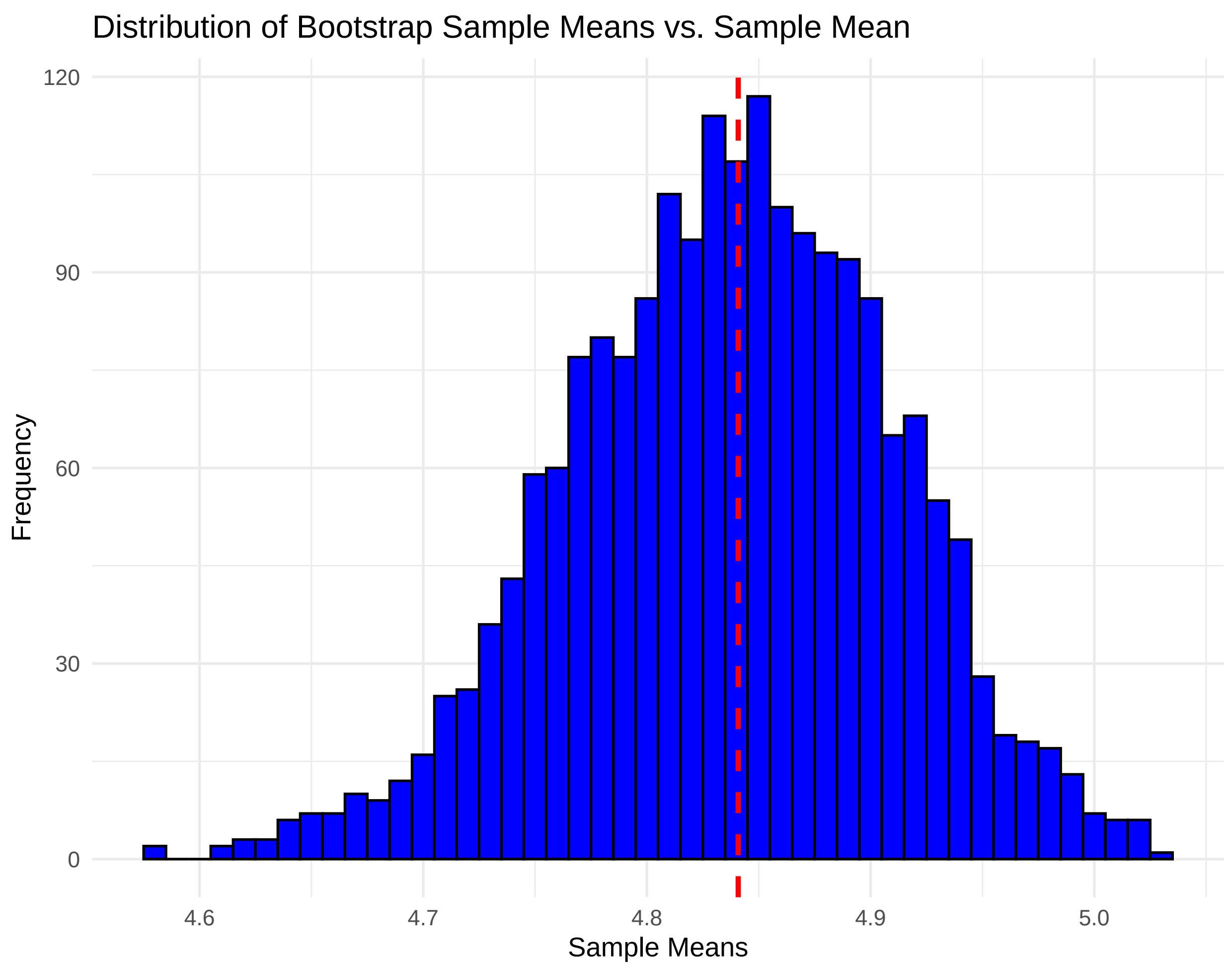

Figure 1: Distribution of the means

```
mean_sample_list <- list(mean.samples[1:20],
                         mean.samples[1:200],
                         mean.samples[1:2000])
names(mean_sample_list) <- c("Sample 20","Sample 200","Sample 2000")

quantiles <- data.frame(sapply(mean_sample_list, function(x) quantile(x, c(0.025, 0.975))))
knitr::kable(quantiles, format = "markdown",caption = "Mean 0.025 and 0.975 Quantiles")
```

Table 1: Mean 0.025 and 0.975 Quantiles

	Sample.20	Sample.200	Sample.2000
2.5%	4.658521	4.702396	4.687479
97.5%	4.969792	4.973354	4.975021

The "true" confidence interval of the mean under the assumption of normality is calculated.

```
true_ci <- t.test(data.points)$conf.int
cat("The 'true' confidence interval of the mean is:\n", true_ci)
```

```
## The 'true' confidence interval of the mean is:
##  4.674344 5.007323
```

According to the results, it is observed that the values for the 3 different bootstrap samples and the original data are roughly equivalent.

Task 1.4

Create 2000 bootstrap samples and compute their medians.

The same procedure is done in task 1.4. This time instead of the mean, we use the median.

```
set.seed(12223236)
median.samples <- replicate(2000, median(sample(data.points, replace=TRUE)))
```

Task 1.4.1

Compute the mean on the first 20 bootstrap medians.

```
median.sample.20 <- median(median.samples[1:20])
```

Task 1.4.2

Compute the mean of the first 200 bootstrap medians.

```
median.sample.200 <- median(median.samples[1:200])
```

Task 1.4.3

Compute the mean based on all 2000 bootstrap medians.

```
median.sample.2000 <- median(median.samples[1:2000])
```

Task 1.4.4

Visualize the distribution all the different bootstrap medians to the sample median.

```
ggplot(data.frame(Medians = median.samples), aes(x = Medians)) +
  geom_histogram(binwidth = 0.01, fill = "blue", color = "black") +
  geom_vline(xintercept = median.orig, color = "red", linetype = "dashed",
             linewidth = 1) +
  labs(title = "Distribution of Bootstrap Sample Medians vs. Sample Median",
       x = "Sample Medians",
       y = "Frequency") +
  theme_minimal()
```

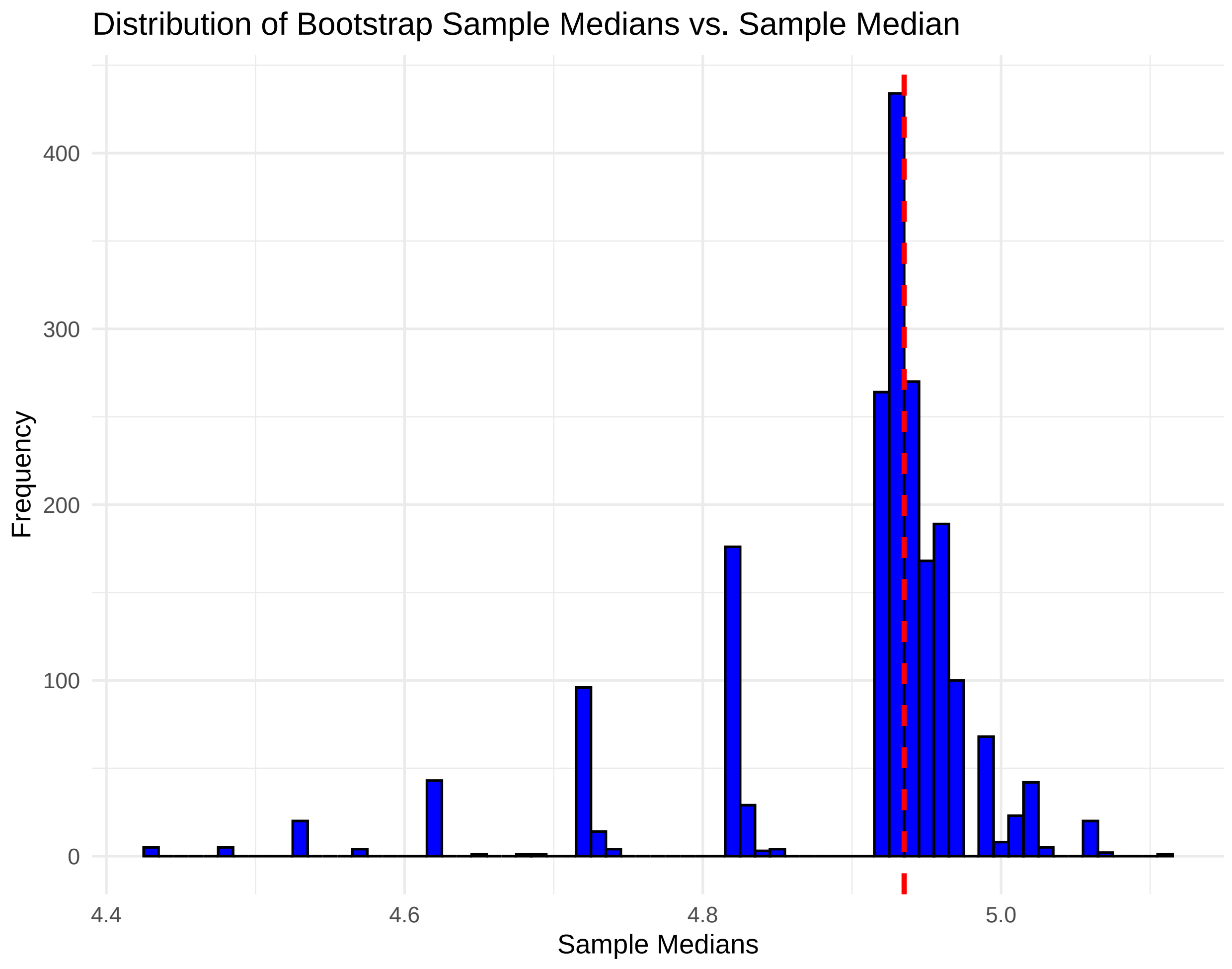

Figure 2: Distribution of the medians

Task 1.4.5

Based on the three different bootstrap sample lengths in 3. compute the corresponding 0.025 and 0.975 quantiles. Compare the three resulting intervals against each other.

```r
median_sample_list <- list(median.samples[1:20],
                           median.samples[1:200],
                           median.samples[1:2000])
names(median_sample_list) <- c("Sample 20","Sample 200","Sample 2000")

quantiles <- data.frame(sapply(median_sample_list, function(x) quantile(x, c(0.025, 0.975))))
knitr::kable(quantiles, format = "markdown",caption = "Median 0.025 and 0.975 Quantiles")
```

Table 2: Median 0.025 and 0.975 Quantiles

	Sample.20	Sample.200	Sample.2000
2.5%	4.575125	4.625	4.625
97.5%	5.015125	5.010	5.025

As mentioned in task 1.3.5, in a repetitive manner the median results are equivalent for the three samples. However, we could say that the 20 bootstrap sample data set is slightly left skewed due to the slight difference in the 2.5% quartile.

Task 2

We wish to explore the effect of outliers on the outcomes of Bootstrap Sampling.

Task 2.1

Set your seed to 1234. And then sample 1960 points from a standard normal distribution to create the vector x.clean then sample 40 observations from uniform(4,5) and denote them as x.cont. The total data is x <- c(x.clean,x.cont). After creating the sample set your seed to your immatriculation number.

In the below chunk of code, the answer is provided based on the details of the task.

```r
set.seed(1234)
x.clean <- rnorm(1960)
x.cont <- runif(40,4,5)
x <- c(x.clean,x.cont)
set.seed(12223236)
```

Task 2.2

Estimate the median, the mean and the trimmed mean with alpha = 0.05 for x and x.clean.

In the table below, the results of the median, mean and trimmed mean are presented.

```r
result <- data.frame(
  row.names = c("x","x_clean"),
  Median = c(median(x),median(x.clean)),
  Mean = c(mean(x), mean(x.clean)),
  Trimmed_Mean = c(mean(x, trim = 0.05), mean(x.clean, trim = 0.05))
)
knitr::kable(result, format = "markdown",
             caption = "Mean, Median, Trimmed Mean Results")
```

Table 3: Mean, Median, Trimmed Mean Results

	Median	Mean	Trimmed_Mean
x	0.0113797	0.0839551	0.0368329
x_clean	-0.0172536	-0.0059690	-0.0014626

Task 2.3

Use nonparametric bootstrap (for x and x.clean) to calculate

- standard error
- 95 percentile CI of all 3 estimators.

In this part of this exercise, the non parametric bootstrap for the two data sets (x and x.clean) will be used in order to calculate the standard error and 95 percentile CI of all 3 estimators. Essentially, in the non parametric bootstrap, we sample with replacement from the original data using the function sample() (this was done, also, in task 1). According to https://bookdown.org/compfinezbook/introcompfinr/The-Nonparametric-Bootstrap.html to avoid the simulation noise, we are going to create 10,000 bootstrap samples. Therefore, the function replicate(), where n will be equal to 10,000, is used for the three estimators (mean, median and trimmed mean).

Regarding the x data set:

```r
# x data set

# Create the median/mean/trimmed mean
x.median <- replicate(10000, median(sample(x, replace=TRUE)))
x.mean <- replicate(10000, mean(sample(x, replace=TRUE)))
x.trimmed_mean <- replicate(10000, mean(sample(x, replace=TRUE),trim = 0.05))

# Calculate the SE and the CI
sd_values <- sapply(list(x.median, x.mean, x.trimmed_mean), sd)
quantile_values <- sapply(list(x.median, x.mean, x.trimmed_mean),
                          function(x) quantile(x, c(0.025, 0.975)))

# Create a data frame for the results of x
result_x <- data.frame(
  row.names = c("x_Median", "x_Mean","x_Trimmed_Mean"),
  Standard_Error = sd_values,
  Quantile_0.025 = quantile_values[1, ],
  Quantile_0.975 = quantile_values[2, ]
)
```

Regarding the x.clean data set:

```r
# x.clean data set

# Create the median/mean/trimmed mean
x_clean.median <- replicate(10000, median(sample(x.clean, replace=TRUE)))
x_clean.mean <- replicate(10000, mean(sample(x.clean, replace=TRUE)))
x_clean.trimmed_mean <- replicate(10000, mean(sample(x.clean, replace=TRUE),trim = 0.05))

# Calculate the SE and the CI
sd_values <- sapply(list(x_clean.median, x_clean.mean, x_clean.trimmed_mean), sd)
quantile_values <- sapply(list(x_clean.median, x_clean.mean, x_clean.trimmed_mean),
                          function(x) quantile(x, c(0.025, 0.975)))


# Create a data frame for the results of x.clean
result_x_clean <- data.frame(
  row.names = c("x_clean_Median", "x_clean_Mean","x_clean_Trimmed_Mean"),
  Standard_Error = sd_values,
  Quantile_0.025 = quantile_values[1, ],
  Quantile_0.975 = quantile_values[2, ]
)
```

Task 2.4

Use parametric bootstrap (based on x and x.clean) to calculate

- standard error
- 95 percentile CI of all 3 estimators
- 95 percentile CI of all 3 estimators
- 95 percentile CI of all 3 estimators

for the mean and the trimmed mean.

When estimating the scale of the of the data in the "robust" case use the mad.

In this part of this exercise, the parametric bootstrap for the two data sets (x and x.clean) will be used in order to calculate the bias, the standard error 95 percentile CI and the bias corrected estimate of the mean and the trimmed mean.

The idea behind the parametric bootstrap is that instead of sample data from the original data set, essentially we simulate k equals to 10,000 samples of n observations (n is the length of x or x.clean data set) from an assumed distribution (in our case normal distribution). In a repetitive manner, the function replicate() is used in order to create all the samples and at the same time to calculate the mean and the trimmed mean.

Below, the function parametric_bootstrap() is responsible for creating the bootstrap sample. It takes as input the x or x.clean data sets, the number of samples (10,000 samples) and the alpha = 0.05 for the confidence intervals. Inside the function, the bias, the standard error, the 95 percentile CI and the corrected bias estimation are calculated. It is worth mentioning that the bias and the corrected bias are equal to mean(means of the bootstrap samples) - mean(original data) and mean(data) - bias, respectively. Finally, the function returns a list with all the results.

```r
parametric_bootstrap <- function(data, n_bootstraps, alpha = 0.05) {
  boot_means <- replicate(n_bootstraps, mean(rnorm(data)))
  boot_trimmed_means <- replicate(n_bootstraps, mean(rnorm(data), trim = 0.05))

  # bias
  bias_mean <- mean(boot_means) - mean(data)
  bias_trimmed_mean <- mean(boot_trimmed_means) - mean(data)

  # standard error
  se_mean <- sd(boot_means)
  se_trimmed_mean <- sd(boot_trimmed_means)

  # 95 percentile CI
  ci_mean <- quantile(boot_means, c(alpha / 2, 1 - alpha / 2))
  ci_trimmed_mean <- quantile(boot_trimmed_means, c(alpha / 2, 1 - alpha / 2))

  # bias-corrected estimate
  corrected_mean <- mean(data) - bias_mean
  corrected_trimmed_mean <- mean(data) - bias_trimmed_mean

  return(list(
    bias_mean = bias_mean,
    se_mean = se_mean,
    ci_mean = ci_mean,
    corrected_mean = corrected_mean,
    bias_trimmed_mean = bias_trimmed_mean,
    se_trimmed_mean = se_trimmed_mean,
    ci_trimmed_mean = ci_trimmed_mean,
    corrected_trimmed_mean = corrected_trimmed_mean
  ))
}
```

Here, the results for the two data sets are presented in a data frame form.

```
set.seed(12223236)
n_bootstraps <- 10000

result_x2 <- parametric_bootstrap(x, n_bootstraps, alpha = 0.05)
result_x_clean2 <- parametric_bootstrap(x.clean, n_bootstraps, alpha = 0.05)

result_x2_df <- data.frame(
  row.names = c("x_Mean","x_Trimmed_Mean"),
  Bias = c(result_x2$bias_mean,result_x2$bias_trimmed_mean),
  Standard_Error = c(result_x2$se_mean,result_x2$se_trimmed_mean) ,
  Quantile_0.025 = c(result_x2$ci_mean[1], result_x2$ci_trimmed_mean[1]),
  Quantile_0.975 = c(result_x2$ci_mean[2], result_x2$ci_trimmed_mean[2]),
  Corrected_Bias = c(result_x2$corrected_mean,result_x2$corrected_trimmed_mean)
)

result_x_clean2_df <- data.frame(
  row.names = c("x_Mean","x_Trimmed_Mean"),
  Bias = c(result_x_clean2$bias_mean,result_x_clean2$bias_trimmed_mean),
  Standard_Error = c(result_x_clean2$se_mean,result_x_clean2$se_trimmed_mean) ,
  Quantile_0.025 = c(result_x_clean2$ci_mean[1], result_x_clean2$ci_trimmed_mean[1]),
  Quantile_0.975 = c(result_x_clean2$ci_mean[2], result_x_clean2$ci_trimmed_mean[2]),
  Corrected_Bias = c(result_x_clean2$corrected_mean,result_x_clean2$corrected_trimmed_mean)
)
```

Task 2.5

Compare and summarize your findings with tables and graphically.

```
knitr::kable(result_x, format = "markdown",
             caption = "Non Parametric Standard Error and
                        Confidence Interval
                        for x data set")
```

Table 4: Non Parametric Standard Error and Confidence Interval
for x data set

	Standard_Error	Quantile_0.025	Quantile_0.975
x_Median	0.0278481	-0.0466360	0.0637379
x_Mean	0.0261076	0.0329840	0.1355108
x_Trimmed_Mean	0.0234615	-0.0089172	0.0832424

```
knitr::kable(result_x_clean, format = "markdown",
             caption = "Non Parametric Standard Error and
                        Confidence Interval
                        for x.clean data set")
```

Table 5: Non Parametric Standard Error and Confidence Interval
for x.clean data set

	Standard_Error	Quantile_0.025	Quantile_0.975
x_clean_Median	0.0276840	-0.0679086	0.0398049
x_clean_Mean	0.0220990	-0.0493758	0.0371035
x_clean_Trimmed_Mean	0.0225564	-0.0448631	0.0430289

According to table 4 for the x data set, the standard errors for the mean, median and the trimmed mean are not far apart. However, it is noticeable the differences in the 95% confidence interval. Specifically, the values of the median and the trimmed mean are close but the values for the mean are different in a large extend. This indicates that the distribution of mean samples are right skewed compared to the median and trimmed mean samples.

On the other hand, according to table 5 and the values of the x.clean data set, there are no any significant differences.

```
knitr::kable(result_x2_df, format = "markdown",
             caption = "Parametric Standard Error and
                        Confidence Interval
                        for x data set")
```

Table 6: Parametric Standard Error and Confidence Interval for x data set

	Bias	Standard_Error	Quantile_0.025	Quantile_0.975	Corrected_Bias
x_Mean	-0.0837872	0.0223951	-0.0433021	0.0442574	0.1677423
x_Trimmed_Mean	-0.0840748	0.0224970	-0.0438004	0.0447830	0.1680299

```
knitr::kable(result_x_clean2_df, format = "markdown",
             caption = "Parametric Standard Error and
                    Confidence Interval
                    for x.clean data set")
```

Table 7: Parametric Standard Error and Confidence Interval for x.clean data set

	Bias	Standard_Error	Quantile_0.025	Quantile_0.975	Corrected_Bias
x_Mean	0.0060577	0.0225127	-0.0440331	0.0443924	-0.0120267
x_Trimmed_Mean	0.0061368	0.0229583	-0.0447521	0.0451467	-0.0121058

According to the tables 6 and 7, there are, also, no noticeable differences. However, what is evident from the two tables is that the values of bias and corrected bias differ between the two data sets.

Task 3

Based on the above tasks and your lecture materials, explain the methodology of bootstrapping for the construction of confidence intervals and parametric or non-parametric tests.

Bootstrapping is a valuable technique for replicating existing data or information. Non-parametric bootstrapping doesn't rely on assuming a specific data distribution; instead, it involves creating multiple replicates of the data to make statistical inferences. In Task 1, it was observed that as the number of bootstraps increased, the mean of these bootstraps approached the true mean, demonstrating an example of the central limit theorem.

Parametric bootstrapping, on the other hand, is used when we are interested in the distribution of a sample and assume that it follows a particular distribution. It involves estimating the parameters of that distribution and replicating it multiple times to gain insights into the original sample. However, if the data is not well-distributed, as seen in Task 2 with data like x or if there are outliers and the calculated metrics are sensitive to outliers, such as the untrimmed mean, this approach can lead to issues.

In the lecture, two types of confidence intervals (CI) were discussed. The first is the "normal bootstrap CI," which uses the standard error of the estimated parameter and a quantile from the assumed distribution of our original sample. The other type is the "percentile Bootstrap CI," which derives confidence intervals by considering the actual quantiles of the metrics from each bootstrap, making it more suitable when the number of bootstraps, denoted as m (number of samples) is large.