05-Distribution and Confidence Intervals of Clementines

2023-11-14

1 Introduction

The example aims to demonstrate estimation and interpretation of confidence intervals. At the end, the two samples are compared with respect to variance and mean values.

The experimental hypotheses was, that weight and size of two samples of Clementine fruits differ. The result is to be visualized with bar charts or box plots. We use only the weight as an example, analysis of the other statistical parameters is left as an optional exercise.

We can now derive the following statistical hypotheses **about the variance:**

- H_0: The variance of both samples is the same.
- H_a: The samples have different variance.

and about the mean:

- H_0: The mean of both samples is the same.
- H_a: The mean values of the samples are different.

2 Material and Methods

The data set consists of two samples of Clementines from the same shop. Weight, width and height of the fruits were measured with a scale and a caliper.

The statistical analysis is performed with the **R** software for statistical computing and graphics (R Core Team 2022), version 4.2.2 and the following packages:

```
library("gplots")  # contains barplot2 with error bars
library("dplyr")   # for pipelines, group_by and summarize
library("ggplot2") # modern plotting package "grammar of graphics"
```

3 Data Analysis

3.1 Prepare and inspect data

- Download the data set `fruits-2023-hse.csv` and use one of RStudio's "Import Dataset" wizards.
- A better alternative is to use `read.csv()`.

```
#  ... do it
```

- plot everything, just for testing:

```
plot(fruits)
```

- split table for box1 and box2:

```
box1 <- subset(fruits, brand == "box1")
box2 <- subset(fruits, brand == "box2")
```

- compare weight of both groups:

```
boxplot(box1$weight, box2$weight, names=c("box1", "box2"))
```

Note: It is also possible to use `boxplot` with the model formula syntax. This is the preferred way, because it does not require to split the data set beforehand:

```
boxplot(weight ~ brand, data = fruits)
```

3.2 Check distribution

We can check the shape of distribution graphically. If mean values of the samples differ much, it has to be done separately for each sample.

```
# use `hist`, `qqnorm`, `qqline`
# ...
```

3.3 Sample statistics

If we assume normal distribution of the data, we can estimate an approximate prediction interval from the sample parameters, i.e. in which size range we find 95% of the weights within one group.

We first calculate mean, sd, N and se for "box1" data set:

```
box1.mean <- mean(box1$weight)
box1.sd   <- sd(box1$weight)
box1.N    <- length(box1$weight)
box1.se   <- box1.sd/sqrt(box1.N)
```

Then we estimate the two-sided 95% prediction interval for the sample, assuming normal distribution:

```
box1.95 <- box1.mean + c(-1.96, 1.96) * box1.sd
box1.95
```

Instead of using 1.96, we could also use the quantile function of the normal distribution instead, e.g. `qnorm(0.975)`for the upper interval or `qnorm(c(0.025, 0.975))` for the lower and upper.

If the data set is large enough, we can compare the prediction interval from above with the **empirical quantiles**, i.e. take it directly from the data. Here we do not assume a normal or any other distribution.

```
quantile(box1$weight, p = c(0.025, 0.975))
```

Now we plot the data and indicate the 95% interval:

```
plot(box1$weight)
abline(h = box1.95, col="red")
```

... and the same as histogram:

```
hist(box1$weight)
abline(v = box1.95, col="red")
rug(box1$weight, col="blue")
```

3.4 Confidence interval of the mean

The **confidence interval** of the mean tells us how precise a mean value was estimated from data. If the sample size is "large enough", the distribution of the raw data does not necessarily need to be normal distributed, because then mean values tend to approximate a normal distribution due to the central limit theorem.

3.4.1 Confidence interval of the mean for the "box1" data

- Calculate the confidence interval of the mean value of the "box1" data set,
- use +/- 1.96 or (better) the quantile of the t-distribution:

```
box1.ci <- box1.mean + qt(p = c(0.025, 0.975), df = box1.N-1) * box1.se
```

Now indicate the confidence interval of the mean in the histogram.

```
abline(v = box1.ci, col="red")
```

3.4.2 Confidence interval for the mean of the "box2" data

We could now in principle do the same as above for the "box2" sample, but this would be rather cumbersome and boring. A more efficient method from package **dplyr** is shown below.

4 Compare samples with F- and t-Test

Null Hypothesis: Both samples have the same mean weight and variance.

Alternative: The mean weight (and possibly also the variance) differs. The fruits bought on Friday and on Tuesday had a different price, so we expect a different quality and probably different size.

```
t.test(weight ~ brand, data = fruits)
```

Perform also the classical t-test (`var.equal=TRUE`) and the F-test (`var.test`). Calculate absolute and relative effect size (mean differences) and interpret the results of all 3 tests.

```
# var.test(...)
# t.test(...)
# ...
```

5 Summary statistics and CI with tidyverse

The following shows how to calculate summary statistics in a more modern and efficient way.

The approach uses the **dplyr** and **ggplot2** packages from the so-called **tidyverse** family of packages. Furthermore, we use the pipeline operator `|>`, that transfers the output of one data manipulation step directly to the next.

Some slides about the use of pipelines can be found under https://tpetzoldt.github.io/elements/slides/x4-pipes-intro.html

5.1 Calculation of summary statistics with dplyr

Summarizing can be done with two functions, `group_by` that adds grouping information to a data frame and `summarize` to calculate summary statistics. In the following, we use the full data set with 4 groups.

```
library("dplyr")
fruits <- read.csv("fruits-2023-hse.csv")

stats <-
  fruits |>
    group_by(brand) |>
    summarize(mean = mean(weight), sd=sd(weight), N=length(weight), se=sd/sqrt(N),
              lwr = mean + qt(p = 0.025, df = N-1) * se,
              upr = mean + qt(p = 0.975, df = N-1) * se
             )

stats
```

5.2 Barchart and errorbars with ggplot2

We can then use the table of summary statistics directly for a bar chart.

```
library("ggplot2")
stats |>
  ggplot(aes(x=brand, y=mean, min=lwr, max=upr))  +
    geom_col() + geom_errorbar()
```

5.3 Additional tasks

Repeat the analysis with other properties of the fruits, e.g. width and height. Create box plots, analyse distribution, create bar charts.

6 References

R Core Team (2021). R: A language and environment for statistical computing. R Foundation for Statistical Computing, Vienna, Austria. URL https://www.R-project.org/

Wickham H, Averick M, Bryan J, Chang W, McGowan LD, François R, Grolemund G, Hayes A, Henry L, Hester J, Kuhn M, Pedersen TL, Miller E, Bache SM, Müller K, Ooms J, Robinson D, Seidel DP, Spinu V, Takahashi K, Vaughan D, Wilke C, Woo K, Yutani H (2019). "Welcome to the tidyverse." Journal of Open Source Software, *4*(43), 1686. doi:10.21105/joss.01686