Independent-samples t-test Tutorial

An independent-samples t-test is used to determine if the data provides statistically significant evidence that the mean of one group is higher than the mean of the other group.

Example. We want to know if a new way of teaching is better. So, we teach one group of students the old way (the control group) and another group of students the new way (the treatment group). We give both groups of students the same quiz and record the results (our data). We want to determine if our data provides statistically significant evidence that the new way of teaching (the treatment) is effective, i.e., does it raise the average quiz score?

Here are the students' scores (see below) as they appear in a spreadsheet. Note "abs" means the student was absent and "late" means the student arrived too late to take the quiz. The yellow is just to highlight the "abs" and "late" entries.

	A	B	C	D	E	F
1	CG_id	ControlGroup		TG_id	TreatmentGroup	
2	1	abs		101	93.3	
3	2	77.4		102	84.8	
4	3	late		103	87.5	
5	4	80.5		104	85.6	
6	5	abs		105	91.2	
7	6	79.1		106	abs	
8	7	76.7		107	late	
9	8	76.6		108	86.8	
10	9	80.1		109	83.9	
11	10	78		110	82	
12	11	78.8		111	91.8	
13	12	81.4		112	84.8	
14	13	82.4		113	89.5	
15	14	71.2		114	94.4	
16	15	80.3		115	91	
17	16	80.8		116	88.1	
18	17	83.4		117	83	
19	18	79.7		118	82.9	
20	19	76.7		119	85.5	
21	20	80.3		120	abs	
22	21	77.2		121	83.9	
23	22	76.7		122	86.3	
24	23	79.2		123	78.5	
25	24	82.1		124	90.8	
26	25	83.9		125	84.4	
27	26	84.6			original data	
28	27	79.4		Treatment Group Data		
29	28	80.4				
30	29	84.6				
31		original data				
32	Control Group Data					
33						

Statistical Goal. We want to know if this data provides **statistically significant evidence**[1] that the treatment, on average, will lead to students getting a higher quiz scores. The key thing is that we are going to make an **inference** about students in general based upon how our students did.

[1] **Understanding statistical significance and p-values: an example.** Suppose you want to know who will win the next election.

Case 1. You ask three people how they will vote and two say "Democrat". This would not be considered statistically significant evidence (exact binomial test, p-value $p = 0.5 > 0.05$) that the Democrats will win because the p-value > 0.05. Explanation: even though 2 out of 3 favored the Democrats, the sample size was too small. Hence, the p-value ended up being too big.

Case 2. You ask 100 people and 51 of them say they'll vote "Democrat". This would not be considered statistically significant evidence (exact binomial test, p-value $p = 0.4602 > 0.05$) that the Democrats will win because the p-value > 0.05. Explanation: even though the sample size is not small, only 51% favored the Democrats, 51% is too close to 50%. Hence, the p-value ended up being too big.

Case 3. You ask 100 people and 59 of them say they'll vote "Democrat". This would be considered statistically significant evidence (exact binomial test, p-value $p = 0.0443 < 0.05$) that the Democrats will win because the p-value < 0.05. Explanation: the sample size is large enough and 59% is not too close to 50%. Hence, the p-value ended up being small enough.

Note. Just because you have statistically significant evidence that some claim is true (e.g., that the democrats will win), it doesn't mean the claim is true. For example, your data might not be representative of the population from which it is drawn; or your respondents might by lying; or your data might be corrupted due to faulty equipment, or your data isn't measuring what you think it is, etc.

The **p-value** is a mathematical measure of how "gullible" we need to be in order to accept a given claim as true based upon the given data. So, the smaller the p-value, the less gullible we'd need to be, and less gullible is good. The cut-off for statistical significance is a p-value of .05. **If the p-value $< .05$ we can say we have statistical significance.** On a practical level, the p-value is the first hurdle your study needs to clear. It tells your readers that if they are to dispute your results, they shouldn't say things like your sample size was too small, but rather they should attack your selection process as being biased or not-random, or that your survey was poorly worded, or that your respondents aren't answering honestly, and so on.

Analyzing the Data in R

The work flow for statistical analysis is like this. You keep your data in a data file, which for us will be a spreadsheet, e.g., Excel[2] or OpenOffice[3], which is free. When you want to analyze your data, you import it into statistics software, which for us will be the free, open-source statistical computing environment called R[4]. We will use an R script, see below, to analyze our data. R will not modify the data file, unless you explicitly tell it to.

We will use the following R script to analyze the data from the example and to perform an independent-samples t-test.

```
# tTestIndSoTL.R
# independent samples t test for SoTL

# Import Data From Spreadsheet (Step 1)
# ------------------------------------
# Data: numeric data, blank cells are ok, but no cells with words.
ControlGroupData = read.table(file = "clipboard", sep = "\t", header = TRUE)[,1];
ControlGroupData ;      # display control group data in console.

TreatmentGroupData = read.table(file = "clipboard", sep = "\t", header = TRUE)[,1];
TreatmentGroupData ;    # display treatment group data in console.

# Find the means and standard deviations (Step 2)
# ----------------------------------------------
mean(ControlGroupData);     # the mean of the Control group
sd(ControlGroupData);       # the standard deviation of the Control group

mean(TreatmentGroupData);   # the mean of the Treatment Group
sd(TreatmentGroupData);     # the standard deviation of the Treatment Group

# Create boxplots of the groups' data (Step 3)
# -------------------------------------------
boxplot(ControlGroupData, TreatmentGroupData,
 main = "Boxplots of Control and Treatment Groups",
 names = c("Control Group", "Treatment Group"),
 col = c("grey","violet"),
 ylab = "Quiz scores"
);

# Do the independent samples t-test mu = 0 (Step 4)
# -------------------------------------------------
t.test(TreatmentGroupData, ControlGroupData,  alternative = "greater", mu = 0);

# End of Script
#----------------------------------------------
```

See the next pages for how to use the above R script.

[2]Microsoft Office

[3]https://www.openoffice.org/ An excellent, free, open-source version of Microsoft Office.

[4]https://www.r-project.org/ We will explore and analyze our data using the very powerful, open source statistical software R. R is probably the most widely used statistics software by professional statisticians. You can **download R for free** and find **R tutorials** at: https://www.r-project.org/. R is very easy to use. You can easily find free resources online to help you learn R. Also, see the Appendix of these notes for a short introduction to using R.

Step 0. Cleaning the data before exporting it to R

It is important to clean the data before importing it into R. This means deleting the non-numerical information like "abs" or "late." There are ways to use R to automate data cleaning, however, unless you have a lot of data, it is easier to just do it by hand in Excel or whatever spreadsheet you are using. The cleaned version of the data is on the right.

before cleaning

	A	B	C	D	E
1	CG_id	ControlGroup		TG_id	TreatmentGroup
2	1	abs		101	93.3
3	2	77.4		102	84.8
4	3	late		103	87.5
5	4	80.5		104	85.6
6	5	abs		105	91.2
7	6	79.1		106	abs
8	7	76.7		107	late
9	8	76.6		108	86.8
10	9	80.1		109	83.9
11	10	78		110	82
12	11	78.8		111	91.8
13	12	81.4		112	84.8
14	13	82.4		113	89.5
15	14	71.2		114	94.4
16	15	80.3		115	91
17	16	80.8		116	88.1
18	17	83.4		117	83
19	18	79.7		118	82.9
20	19	76.7		119	85.5
21	20	80.3		120	abs
22	21	77.2		121	83.9
23	22	76.7		122	86.3
24	23	79.2		123	78.5
25	24	82.1		124	90.8
26	25	83.9		125	84.4
27	26	84.6			original data
28	27	79.4		Treatment Group Data	
29	28	80.4			
30	29	84.6			
31		original data			
32	Control Group Data				

after cleaning

I	J	K	L	M
CG_id	ControlGroup		TG_id	TreatmentGroup
1			101	93.3
2	77.4		102	84.8
3			103	87.5
4	80.5		104	85.6
5			105	91.2
6	79.1		106	
7	76.7		107	
8	76.6		108	86.8
9	80.1		109	83.9
10	78		110	82
11	78.8		111	91.8
12	81.4		112	84.8
13	82.4		113	89.5
14	71.2		114	94.4
15	80.3		115	91
16	80.8		116	88.1
17	83.4		117	83
18	79.7		118	82.9
19	76.7		119	85.5
20	80.3		120	
21	77.2		121	83.9
22	76.7		122	86.3
23	79.2		123	78.5
24	82.1		124	90.8
25	83.9		125	84.4
26	84.6			cleaned data
27	79.4		Treatment Group Data	
28	80.4			
29	84.6			
	cleaned data			
Control Group Data				

See next page.

Step 1. Import the data into R by copy and paste. Open the `tTestIndSoTL.R` script file in R. Then, in the spreadsheet copy the cleaned ControlGroup data column to the "clipboard". When you copy the column, make sure to include the **header**, i.e., the word "ControlGroup". See image below (left).

Control Group

I	J
CG_id	ControlGroup
1	
2	77.4
3	
4	80.5
5	
6	79.1
7	76.7
8	76.6
9	80.1
10	78
11	78.8
12	81.4
13	82.4
14	71.2
15	80.3
16	80.8
17	83.4
18	79.7
19	76.7
20	80.3
21	77.2
22	76.7
23	79.2
24	82.1
25	83.9
26	84.6
27	79.4
28	80.4
29	84.6
	cleaned data
Control Group Data	

Treatment Group

L	M
TG_id	TreatmentGroup
101	93.3
102	84.8
103	87.5
104	85.6
105	91.2
106	
107	
108	86.8
109	83.9
110	82
111	91.8
112	84.8
113	89.5
114	94.4
115	91
116	88.1
117	83
118	82.9
119	85.5
120	
121	83.9
122	86.3
123	78.5
124	90.8
125	84.4
	cleaned data
Treatment Group Data	

Then, back in R, in the `tTestIndSoTL.R` script, run the following lines of code:

```
# Import Data From Spreadsheet (Step 1)
# -------------------------------------
# Data: numeric data, blank cells are ok, but no cells with words.
ControlGroupData = read.table(file = "clipboard", sep = "\t", header = TRUE)[,1];
ControlGroupData ;      # display control group data in console.
```

Then copy the treatment group data column to the clipboard (see image above, right) and then, back in R, run the following lines of code:

```
TreatmentGroupData = read.table(file = "clipboard", sep = "\t", header = TRUE)[,1];
TreatmentGroupData ;    # display treatment group data in console.
```

Step 2. Find the mean and standard deviation of quiz scores. We will need this information when we report our results.

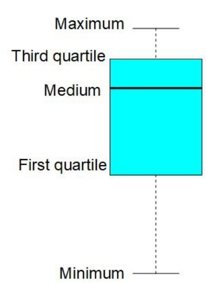
```
# Find the means and standard deviations (Step 2)
# -----------------------------------------------
mean(ControlGroupData);     # the mean of the Control group
sd(ControlGroupData);       # the standard deviation of the Control group

mean(TreatmentGroupData);   # the mean of the Treatment Group
sd(TreatmentGroupData);     # the standard deviation of the Treatment group
```

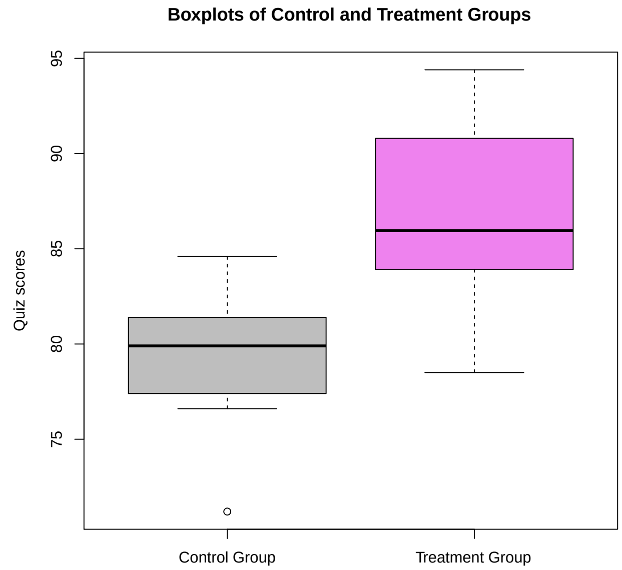
R returns:

```
> mean(ControlGroupData);     # the mean of the Control group
[1] 79.67308
> sd(ControlGroupData);       # the standard deviation of the Control group
[1] 2.988586
> mean(TreatmentGroupData);   # the mean of the Treatment Group
[1] 86.81818
> sd(TreatmentGroupData);     # the standard deviation of the Treatment Group
[1] 4.041527
```

Step 3. Produce boxplots of the groups. Perhaps the quickest way to get a feel for how the control and treatment groups perform is to compare boxplots of the quiz scores. Also, boxplots are visually appealing. They will make your article look nicer!




```
# Create boxplots of the groups' data (Step 3)
# --------------------------------------------
boxplot(ControlGroupData, TreatmentGroupData,
 main = "Boxplots of Control and Treatment Groups",
 names = c("Control Group", "Treatment Group"),
 col = c("grey","violet")
);
```



Step 4. Do the independent-samples t-test.

```
# Do the independent samples t-test mu = 0 (Step 4)
# --------------------------------------------------
t.test(TreatmentGroupData, ControlGroupData,  alternative = "greater", mu = 0);
```

Note about the above syntax. The above R command does the t-test to determine if the data provides statistically significant evidence that the average quiz score of "treatment group" students (who will be taught with the new method) will be greater than the average of the "control group" students (who were taught with the old method).

Interpreting the t-test results. After we run the t-test in R:

```
# Do the independent samples t-test mu = 0 (Step 4)
# --------------------------------------------------
t.test(TreatmentGroupData, ControlGroupData,  alternative = "greater", mu = 0);
```

R outputs the results summary:

```
        Welch Two Sample t-test

data:  TreatmentGroupData and ControlGroupData
t = 6.8564, df = 38.081, p-value = 1.904e-08
alternative hypothesis: true difference in means is greater than 0
95 percent confidence interval:
5.388261      Inf
sample estimates:
mean of x mean of y
86.81818  79.67308
```

The most important part of the summary is the p-value. In this example the `p-value = 1.904e-08`, which is less than 0.05. This tells us that our data provides statistically significant evidence that the treatment is effective. Typically, peer reviewed journals will allow us to say we have statistically significant evidence only if the p-value < 0.05.

See next page for how to report the results.

How to report the results of the paired-samples t-test in your article:

Students being taught by the new method had higher quiz scores (M = 86.82, SD = 4.042) than did those being taught by the old methods (M = 79.67, SD = 2.989), t(38.081) = 6.8564, p < .0001.

Notes. We report: $t(38.081) = 6.8564$ because in the summary it says `t = 6.8564, df = 38.081`. If the p-value < .0001 we just report the p-value as p < .0001, if the p-value is between .0001 and .05 we'd report the actual p-value. If the p-value > .05, then our data doesn't provide statistically significant evidence that the new method is better than the old method,

It is traditional to round the mean (M) to one more decimal place than appears in the data, and to round the standard deviation (SD) to one more decimal place than the mean. We calculated the means and standard deviations in Step 2.

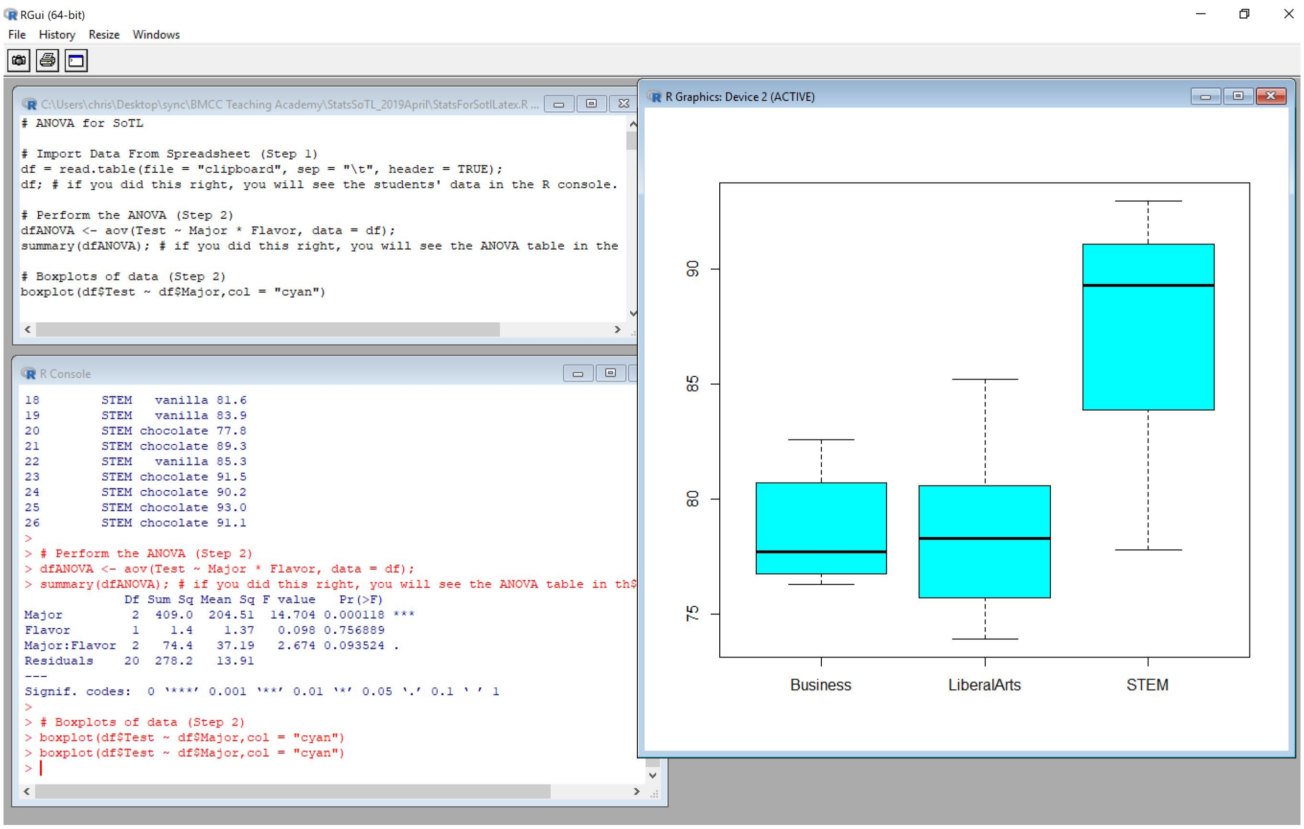
R calls the t-test that we just did the "Welch Two Sample t-test" because we aren't assuming equal variances. Don't worry about that.

See next page for the Appendix: Getting Started with R.

Appendix

Getting Started with R

This section is quick introduction to R to help you get started.
You can download R for free from https://www.r-project.org/.



The above shows a screen shot of **R on a Windows PC**. R on an Apple will look slightly different. When you start R the "R Console" will appear (see above, bottom left). You can type commands directly into the R Console, however, it is better to run commands from the "R Editor" (top left). In the R Editor you can create "R scripts" which are just R commands in a text file. When you run R commands from the R Editor you'll see the output in the "R Console" or in an "R Graphics" window (right).

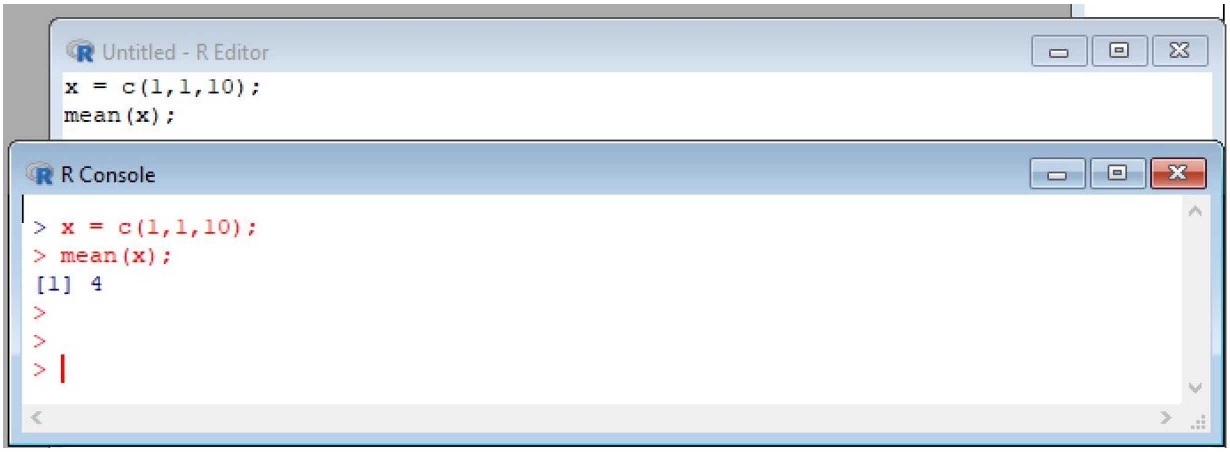
Two minute R Tutorial to Get You Started.

Finding the average of the numbers 1, 1, 10 in R.

1. Open R.
2. Inside of R create the script file by doing the folowing. Go to the menu, then File, then New script. A blank script will appear in the R Editor. Type, or copy and paste, the following R commands into your R script:

```
x = c(1,1,10);
mean(x);
```

3. Run the script. **On a Windows PC**, to run a line of code in an R script, put your cursor anywhere in that line and do **Ctrl R**. To run a bunch of lines, highlight the lines by doing **Ctrl Shift** and the **Up** or **Down** key, then do **Ctrl R** to run them.
   On an Apple Computer, to run a line of code in an R script, put your cursor anywhere in that line and press **Command Enter** at the same time.
4. The results you should get. After you run this script you should see, in the R console window, the commands from the script displayed together with the number 4, since the mean of 1, 1, 10 is $\frac{1+1+10}{3} = \frac{12}{3} = 4$.



Notes:

1. If you want to save your script file you should save it with the .R extension. Something like `MyScript.R`.

2. To load your R script file into R you need to open R and then from inside of R, go to the menu, File, Open. Double clicking on the R file typically won't load it into R.

3. If you don't know how to do something in R just Google it. R is so widely used that just about any question you might have will have already been asked and answered somewhere online.

4. If you want to put a comment in your R script use the `#` symbol. Anything on a line after the `#` symbol is considered by R to be "comment", and won't be compiled.

End of R tutorial.