WEEK_3_t-test

Kevin Healy

2023-08-28

In this script you will calculate some p-values using some coing flipping code and you will carry out some t-tests.

Uploading data and packages

Its a good idea to load all of the packages you plan to use at the start to keep everything nice a tidy.

For this session you we will use the iris data set so we will save it as the object iris_data. We will return to using this data later after we flip some coins

```
iris_data <- iris
```

Coin flipping

In the lecture we used a coin to demonstrated p-values. In this simple case we will flip a coin 10 times and test if it was a fair coin or if it was biased coin. In this case the Null hypothesis is that its a fair coin and that the coin will have a 50% chance of showing heads up. The alternative is that the coin is biased and that it will show heads or tails more often than expected from a fair coin.

To test this hypothesis we could look at our observations and ask how likely would such a result be with a fair coin.

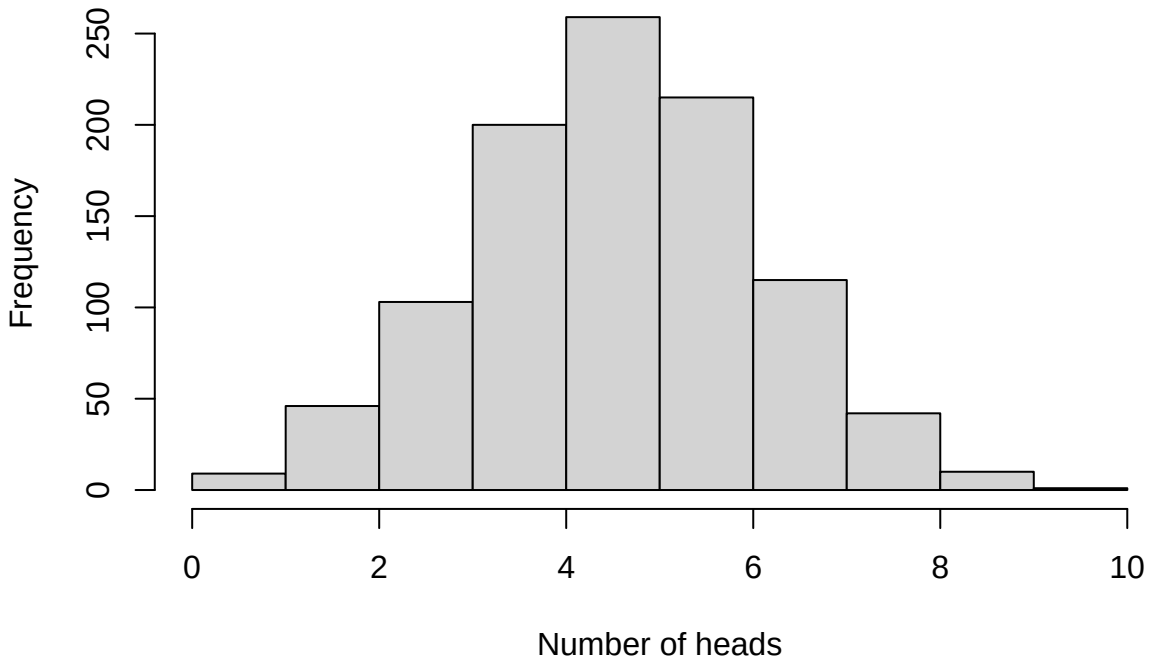
Lets simulate some data below for a fair coin and explore the issue in how to decide if something is more different than we might expect.

```
##Copy and paste this entire section of code and run it in the console

#this is object that saves the result
no_heads_fair <- c()

#this is the loop that asks R to run the code enclosed in the {} brackets
#in this case we are asking r to run it when i = 1, then when i = 2, etc
#until it runs i = 1000 and stops
for(i in 1:1000){

  #This line asks for to draw a value between "h" and "t"
  #10 times, with replacement (i.e. when we draw a value we can draw the same
  #value again next time), with a chance of 50%

  test <- sample(c("h","t"),
```

```
                  10,
                  replace = T,
                  prob = c(50,50))

  #we ask how many heads we got in each iteration of the loop

  no_heads_fair[i] <- length(test[test=="h"])

}
```

We now have 1000 trails of flipping a coin 10 times and adding up the number of heads. We could plot a distribution of this, which would give us a binomial distribution, which is very similar to a normal distribution.

```
hist(no_heads_fair,
     xlab = "Number of heads",
     main = "")
```



Notice that in most of the 1000 trails we got between 4 and 6 heads. In such cases we would be happy to say the coin if fair. The reason for that is we know that a fair coin should roughly give heads 50% of the time but also that there is natural variation. For most cases we would not be able to reject the null, which is that there is not strong enough evidence to says its not a fair coin.

But notice that there are cases were our fair coin gives heads either 1 or 9 times or maybe even 10 times. These cases would suggest the coin is biased but we know its not. However, they are rare enough that we just accept that sometimes will reject the null hypothesis and say its a biased coin. We can calculate the odds of these types of events.

```
#Since our simulation is fair coin we can ask how many times did we get more
#than 8 heads in one trial
head_over_8 <- length(no_heads_fair[no_heads_fair > 8])

#since we ran 1000 trails the probability of such an event is roughly
```

```
prop_over_8 <- head_over_8/1000
prop_over_8
```

```
## [1] 0.011
```

Since getting more than 8 heads out of 10 is rare we can be confident if we saw a coin do that we would say its biased. This is were the idea of p-value comes in. We set a probability, or alpha value, typically at 5%, where we say that if some observation has a probability (or p-value) less than that value under the null case we reject the null. Basically we say that the observation was very unlikely to happen if the Null was true so we reject the null as the explanation leaving the alternative as the remaining explanation.

What about if we saw a coin give 7 heads out of 10 would we reject that. We can check like before.

```
#Since our simulation is a fair coin we can ask how many times did we get more
#than 7 heads in one trial
head_over_7 <- length(no_heads_fair[no_heads_fair > 7])

#since we ran 1000 trails the probability of such an event is roughly
prop_over_7 <- head_over_7/1000
prop_over_7
```

```
## [1] 0.053
```

We can see that the probability of a coin getting 7 heads or more is greater than 5% (you might get slightly less than 0.05 in some cases) so in that case we cannot reject the Null, leaving us to accept the coin is fair.

Overall this means in science sometimes we reject the null when we shouldn't and other times we fail to reject the null when we should (i.e. because even though we see a difference we are not sure enough the difference could have been generated by chance.)

Unfair coin

Lets simulate an unfair coin and see how that compares. Like above we will run the same trail of flipping a coin 10 times and counting the number of heads. We will repeate this 1000 times but now we will make the coin give heads 75% of the time.

```
##Copy and paste this entire section of code and run it in the console

#this is object that saves the result
no_heads_unfair <- c()

#this is the loop that asks R to run the code enclosed in the {} brackets
#in this case we are asking r to run it when i = 1, then when i = 2, etc
#until it runs i = 1000 and stops
for(i in 1:1000){

  #This line asks for to draw a value between "h" and "t"
  #10 times, with replacement (i.e. when we draw a value we can draw the same
  #value again next time), with a chance of 75%

  test <- sample(c("h","t"),
                 10,
```

```
                  replace = T,
                  prob = c(75,25))

  #we ask how many heads we got in each iteration of the loop

  no_heads_unfair[i] <- length(test[test=="h"])

}
```

We saw above that the probability of having 8 or more heads less was less than 5% with a fair coin so we can see how often we find 8 or more heads with our biased coin.

```
no_unfair_over7 <- length(no_heads_unfair[no_heads_unfair > 7])

#what percent of the time do we get 8 or more heads
no_unfair_over7/1000
```

```
## [1] 0.533
```

We can see that about 50% of the trails have 8 or more heads and so in all of those trials we would have rejected the null as the probability of getting 8 or more heads on a fair coin is about 1.5% (which is less than our 5% alpha value). However, it also shows that for the rest of the trials we could not reject the null as the number of heads in those trials are not extreme enough to be sure its really different enough and not just a fair coin giving a slightly unlikely event. These are cases of a type II error.

In this case used a 1 tailed test as we only calculated a p-value of the chance of getting more heads than expected. A two tailed test would include the probabilities of either getting extremely high numbers of head results or extremely low numbers of heads. Since a fail coin should give symmetrical results the p-value for the above case is just 0.015 (1.5%) by two so it would be 0.03 (3%), which is still lower than the alpha value of 5% so we can still reject the NULL.

In reality each of the 1000 test only have 10 observations which is very low. If we increased the number coin flips (i.e. observations) in a given trail we would have more data to help us be more confident there are differences. Try play with the code and see if you can increase the number of flips to 100 and how that changes the number of times we can reject the null.

T-tests

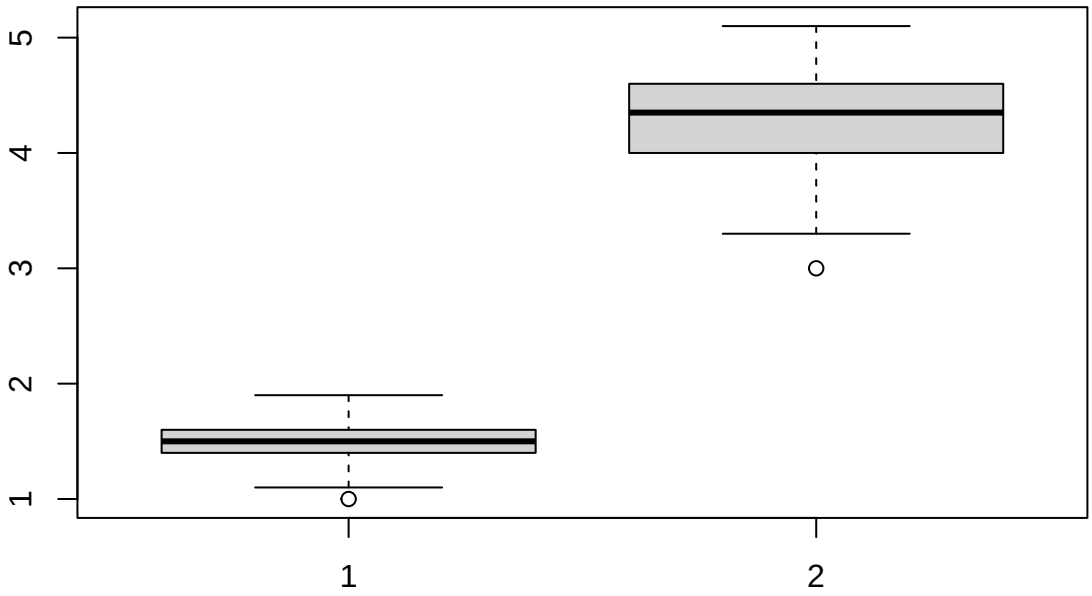
We can use P-values to decide if two groups are significantly different. Lets compare 2 groups using t-tests. A t-test will compare the mean values of 2 groups and, along with the variance, will test if the groups are different compared to our Null expectation.

Using the Iris dataset lets compare the petal length between 2 species.

```
#petal length for setosa
setosa_petal <- iris_data[iris_data$Species == "setosa", "Petal.Length"]

#petal length for versicolor
versicolor_petal <- iris_data[iris_data$Species == "versicolor", "Petal.Length"]
```

When doing any analysis always plot the data so you can see if there is anything odd about it.

```
boxplot(setosa_petal, versicolor_petal)
```



We can visually see that petal length is clearly different in both groups. We could also calculate the means in each species.

```
#mean petal length of setosa
mean(setosa_petal)
```

```
## [1] 1.462
```

```
#mean petal length of versicolor
mean(versicolor_petal)
```

```
## [1] 4.26
```

While the means are different this could just be by chance (like we saw with the fair coin). To test if they are significantly different, that is the chance that we see the difference is lower than 5% given that the null is true, we use a t-test. The student t-test is the simplest so lets start there.

```
t.test(setosa_petal, versicolor_petal, var.equal = T)
```

```
##
##  Two Sample t-test
##
## data:  setosa_petal and versicolor_petal
## t = -39.493, df = 98, p-value < 2.2e-16
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -2.938597 -2.657403
## sample estimates:
## mean of x mean of y
##     1.462     4.260
```

We can see that the t-test estimates the mean of each group and also gives a t-value and degrees of freedom (df) which it uses to calculate the p-value.

In this case the p-value is less than 0.05 so we can reject the null hypothesis.

Notice also that the degrees of freedom is 98, with is the samples size (100) minus the number of parameters (2 means).

When reporting such a result in a results section of the thesis or paper we would write something like. "We find that the petal length of setosa (mean = 1.462, SD = 0.17, n = 50) and versicolor (mean = 4.26, SD = 0.47, n = 50) are significantly different ($t = -39.50$, $p < 0.05$)"

Here we assumed the variances were equal in the two groups using the "var.equal = T" argument. Lets check if that assumption actually holds.

```
var(setosa_petal)
```

```
## [1] 0.03015918
```

```
var(versicolor_petal)
```

```
## [1] 0.2208163
```

We can see the variances are quite different so lets use the Welch two sample t-test which does not make that assumption. This is the default t-test in R and should be typically used instead of the student t-test

```
#We can set var.equal = F to ask for a Welch Two Sample t-test
t.test(setosa_petal, versicolor_petal, var.equal = F)
```

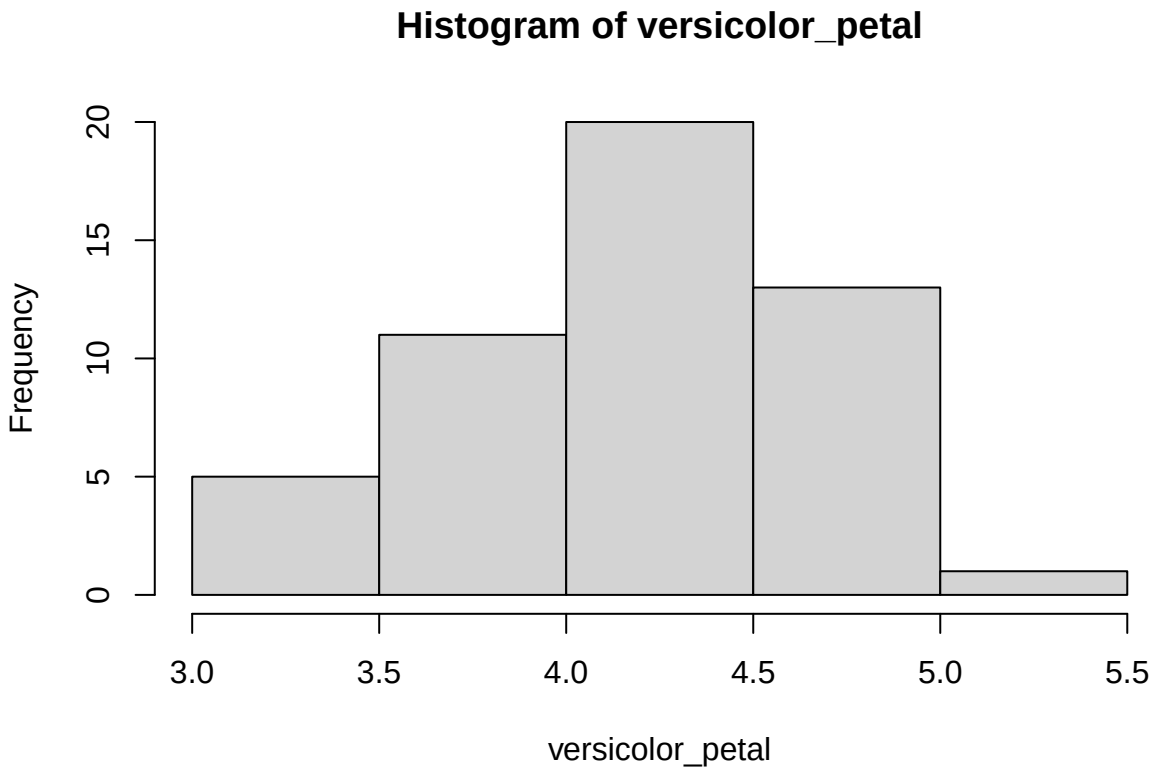
```
##
##  Welch Two Sample t-test
##
## data:  setosa_petal and versicolor_petal
## t = -39.493, df = 62.14, p-value < 2.2e-16
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -2.939618 -2.656382
## sample estimates:
## mean of x mean of y
##     1.462     4.260
```

Notice that the degree of freedom is now lower (62) but that the p-value is still much lower than 0.05. This lower df shows that this test requires higher samples sizes compared to the student t-test, however, this trade-off is nearly always worth it.

The next assumption is that the data is normally distributed. Lets plot some histograms to test this.

```
hist(setosa_petal)
```


```
hist(versicolor_petal)
```



These both look normal enough so in reality we would stick with the Welch test. However, if they were not normal we would use the Mann–Whitney U test which is a non-parametric version of the t-test. This means it does not assume a normal distribution in the data.

```
wilcox.test(setosa_petal, versicolor_petal)
```

```
##
##  Wilcoxon rank sum test with continuity correction
##
## data:  setosa_petal and versicolor_petal
## W = 0, p-value < 2.2e-16
## alternative hypothesis: true location shift is not equal to 0
```

This looks slightly different to the output of the other t-test as it does not compare the means of the distributions but the rank sums of them calculating the U or W value (see lecture notes on what this means). We also see that the p-value is well below 0.05 so we can still reject the null. Out of all the t-test the Mann–Whitney U is the least powerful in terms of requiring more data to reject the null compared to the others. However, it requires the least amount of assumptions so is the most conservative of the tests and, quite frankly, my fav t-test.

Paired T tests

Finally, imagine a case were our data is paired in some way. For example, we might run an experiment and compare the same group before and after some intervention. Lets say we give the entire class a surprise test on R. After the test we could than have some intervention, say a tutorial session, and give the test again some days later. Since its the same people doing the test we know that they are paired. For example, a students that did well in the first test is likely to also do well in the second test so it makes more sense to compare these 2 data points more directly. We can give R this information using the paired argument when running a t-test.

First lets make up some data for 10 students taking a test before (first_test_scores) and after (second_test_scores) some intervention.

```
#Student ID
student_ID <- c("a", "b", "c", "d", "e", "f", "g", "h", "i", "j")

#Scores from first test
first_test_scores <- c(35, 50, 33, 78, 49, 60, 70, 20, 58, 48)

#Scores from second test
second_test_scores <- c(40, 59, 47, 82, 60, 65, 77, 38, 66, 55)

#We can put them all into 1 dataset to keep it tidier
test_data <- data.frame(student_ID,
                        first_test_scores,
                        second_test_scores)
```

We should be careful to make sure that each row corresponds to the same individual as when we use a paired t-test R assumes that entries are paired in the order that they appear in the variables. That is the first entry of first_test_scores is paired with the first entry of second_test_scores etc.

```
t.test(test_data$first_test_scores,
       test_data$second_test_scores,
       var.equal = F,
       paired = TRUE)
```

```
##
##  Paired t-test
##
## data:  test_data$first_test_scores and test_data$second_test_scores
## t = -6.3, df = 9, p-value = 0.000141
## alternative hypothesis: true mean difference is not equal to 0
## 95 percent confidence interval:
##  -11.95983  -5.64017
## sample estimates:
## mean difference
##            -8.8
```

We can see from this that there is a significant difference between the mean test scores of 50.1 (SD = 17.54) before the intervention and 58.9 (SD = 15.54) after the intervention (mean difference = -8.8, n = 20, t = -6.3, $p < 0.05$).

Lets compare this to a normal independent t-test where we don't pair up the observations.

```
t.test(test_data$first_test_scores,
       test_data$second_test_scores,
       var.equal = F,
       paired = F)
```

```
##
##  Welch Two Sample t-test
##
## data:  test_data$first_test_scores and test_data$second_test_scores
## t = -1.2219, df = 17.401, p-value = 0.238
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -23.967621   6.367621
## sample estimates:
## mean of x mean of y
##      50.1      58.9
```

Now we find that we don't find a significant difference between the two groups (t = -1.22, n = 20, p = 0.24) as the p-value is above 0.05. The reason for this is that comparing the same individual before and after in a paired t-test is more powerful than just comparing the overall means of the group before and after in a more standard t-test.