Statistical Hypothesis Testing in R

Zihao Li

December 3, 2017

1. Introduction

A statistical hypothesis test, as a method of statistical inference, is often performed between two data sets (one of which might be an idealized synthetic one). Before the test, a relationship between the data sets is proposed, and the purpose of hypothesis testing is to determine whether that relationship holds with statistical significance. The proposed relationship is often called the null hypothesis, deoted as H_0, and the test determines whether there's enough evidence to reject the null hypothesis.

Since hypothesis testing is a crucial part in statistical inference, the primary motivation of this post is to inform the audience how it can be done using RStudio. The contents of this post include two kinds of hypothesis tests: t-test and Chi-square test. These are among the most common statistical tests used in daily life. For each test I describe its mathmatical theory and how it can be done in R. I've also included some plots to help visualize the results.

2. Instructions and Reproducibility

1. R version: 3.3.3. Running under: OS X Yosemite 10.10.5.
2. Required packages: ggplot2 (version 2.2.1), plyr (version 1.8.4)
3. Dependencies: there are no dependencies between packages
4. Random seeds: there are no random seeds used in this post
5. Required functions: summarySE(). This entire function is written in the 4th code chunk of this post in order to achieve full reproducibility. The original function can be found in the second reference.
6. Required data sets: all data sets are explicitly defined in this post itself. There is no need to download datasets online. The first dataset is copied from the first reference. The second dataset is a fictitious one created by myself.

This post should be entirely reproducible since it requires no outside data or functions whatsoever. There are no specific instructions to rerun or replicate the results, because all required data, functions, and packages are defined in the code chunks themselves.

3. Two-sample t-test

3.1 Background and Mathematical Theory

The general idea behind a two-sample t-test is simple: to determine whether there is any difference between the population means of based on the two data sets. For instance, we might want to know whether a new cancer treatment significantly increases patients' life expectancies, or whether a review session improves students' exam performances.

In the above scenarios, the null hypothesis (H_0) would be that the population means are not statistically different ($\mu_1 = \mu_2$). The alternative hypothesis (H_a) can be either $\mu_1 \neq \mu_2$, $\mu_1 > \mu_2$, or $\mu_1 < \mu_2$. The typical alpha level used in most statistical inferences is 0.05. This level represents the possibility of accepting H_a when H_0 is actually true.

The outcome of the t-test is a number called the "t-statistic". This number, along with the degree of freedom, can then determine the so called "p-value." Generally speaking, this value represents the probability that H_0 is true based on the data. If $p < \alpha$, we reject the null hypothesis. If $p > \alpha$, we fail to reject the null hypothesis.

Let n_1 and n_2 denote the sample sizes, Y_1, Y_2 be their corresponding sample means, and s_1, s_2 be their corresponding standard deviations, then the formula to determine the t-statistics is given as the following:

$$\begin{equation} T = \dfrac{Y_1 - Y_2}{(\dfrac{s_1^2}{n_1} - \dfrac{s_2^2}{n_2})^{1/2}} \end{equation}$$

3.2 Examples

To illustrate how a two-sample t-test is performed in RStudio, we first create a data frame that contains two samples. The data sets can be found in the first reference. For the sake of reproducibility, I have copied it to the following code chunk.

```r
# Creating sample vectors
sample1 <- c(19.7475, 19.8387, 12.6873, 17.6973, 19.0878, 30.5562,
             14.5291, 14.7627, 14.3439, 12.5745, 11.0734, 19.4998,
             18.3869, 10.7374, 18.0030, 18.1730, 18.8374, 17.9287,
             15.3563, 18.6004, 11.7280, 12.2898, 21.0552, 21.4184,
             25.5953)

sample2 <- c(17.4715, 20.0386, 12.6012, 20.4401, 22.4969, 9.8613,
             19.6289, 9.7741, 15.1119, 17.4448, 23.4827, 24.9357,
             19.9265, 7.9955, 17.6675, 13.6029, 17.8812, 16.4178,
             5.1385, 7.0984, 18.1181, 20.2681, 14.7372, 22.5915,
             16.7546)
```

The descriptive statistics are shown as the following:

```r
# Summary of first sample
summary(sample1)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max. 
##   10.74   14.34   18.00   17.38   19.50   30.56
```

```r
# Summary of second sample
summary(sample2)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max. 
##   5.138  13.600  17.470  16.460  20.040  24.940
```

The command in RStudio that calculates the two-tail t-statistic and p-value is the follwing:

```r
# Perform a two-sample t-test using the t.test() function
t.test(sample1, sample2)
```

```
## 
##  Welch Two Sample t-test
## 
## data:  sample1 and sample2
## t = 0.65571, df = 47.127, p-value = 0.5152
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -1.904233  3.746033
## sample estimates:
## mean of x mean of y 
##  17.38032  16.45942
```

Based on the output, we see that the t-statistic is -0.65571, the degree of freedom is 47.127, and the p-value is 0.5152. The p-value here is greater than the alpha level (0.05), and thus we conclude that we fail to reject the null hypothesis.

3.3 Visualization

The easiest way to visualize the result of a t-test is through a barplot with error bars. The error bars represent the confidence level (1 - alpha level). If the error bars overlap with each other, then we fail to reject the null hypothesis.

One way to obtain all necessary parameters for the barchart is to write a function. Here I use a function given in the second reference, which calculates the mean, t-statistic, and standard error. All lines are explained with an original comment from the website. If that’s unclear, please feel free to go to the website and take a look. The function is given as the following:

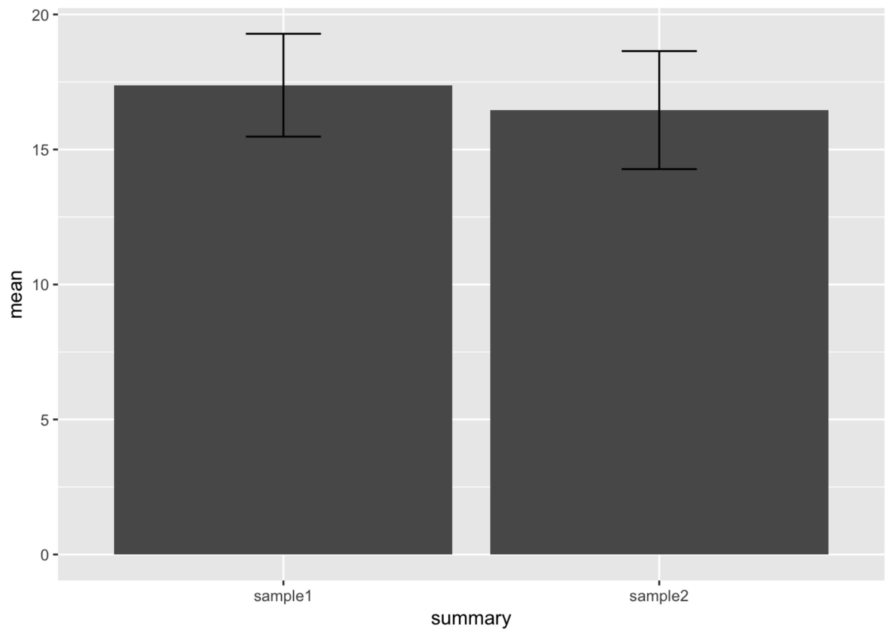
```r
# data is the input vector or data frame
# measurevar is the variable to be sumarized
# groupvars indicates which variables should be grouped together
# na.rm indicates whether to remove missing values
# conf.interval is the confidence interval (Default is 0.95)

summarySE <- function(data = NULL, measurevar, groupvars = NULL,
                      na.rm = F, conf.interval = .95) {
    library(plyr)

    # If na.rm == T, exclude NA from the vector length
    length2 <- function (x, na.rm = FALSE) {
        if (na.rm) sum(!is.na(x))
        else       length(x)
    }

    # For each group's data frame, return a vector with
    # N, mean, and sd
    # ddply() splits data frame, apply a function, then
    # return the results in a data frame

    datac <- ddply(data, groupvars,
      .fun = function(xx, col) {
        c(N    = length2(xx[[col]], na.rm=na.rm),
          mean = mean   (xx[[col]], na.rm=na.rm),
          sd   = sd     (xx[[col]], na.rm=na.rm)
        )
      },
      measurevar
    )

    # Calculate standard error of the mean
    datac$se <- datac$sd / sqrt(datac$N)

    # Confidence interval multiplier for standard error
    # Calculate t-statistic for confidence interval using qt()
    # ci is the length of the error bars

    ciMult <- qt(conf.interval/2 + .5, datac$N-1)
    datac$ci <- datac$se * ciMult

    return(datac)
}
```

The barplot of the previous data sets can then be obtained as the following (using the previous function):

```r
library(ggplot2)

# Combine the vectors to create a data frame
sample <- data.frame(sample1, sample2)

# Call the summarySE() function to obtain summaries
summary1 <- summarySE(sample, measurevar = 'sample1')
summary2 <- summarySE(sample, measurevar = 'sample2')

# Combine the summaries to modify the original data frame
sample <- data.frame(summary = c('sample1', 'sample2'),
                     mean = c(summary1$mean, summary2$mean),
                     ci = c(summary1$ci, summary2$ci))
sample
```

```
##   summary     mean       ci
## 1 sample1 17.38032 1.905027
## 2 sample2 16.45942 2.184645
```

```r
# Draw the barchart
ggplot(sample, aes(x = summary, y = mean)) +
    geom_bar(stat = "identity") +
    geom_errorbar(aes(ymin= mean - ci, ymax = mean + ci),
                  width=.2)    # generate the error bars
```



As shown, the errorbars of the barplot overlap (in terms of the y-interval). This indicates that the difference between the means of the two data sets is not statistically significant, and that we fail to reject the null hypothesis.

4. Chi-square test

4.1 Background and Mathematical Theory

Chi-square (goodness of fit) test is mostly used to test whether a given data set follows a specific distribution. A classic example would be to determine whether the types of candies in a bag follow a discrete uniform distribution.

The idea of the Chi-square test is similar to that of the two-sample t-test. We claim a null hypothesis and an alternative hypothesis, and then obtain a test statistic, called "χ^2-statistic". We then obtain a p-value based on the χ^2-statistic and the degree of freedom, and compare it to the alpha level. We reject the null hypothesis if $p < α$, and fail to reject if $p > α$.

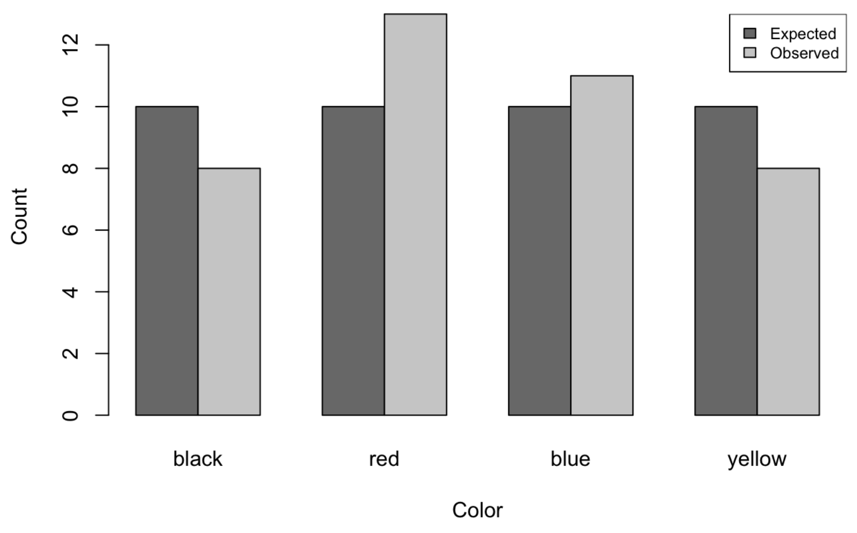
Let O and E represent the observed and expected frequencies, respectively. The formula to determine χ^2 is given as the following:

\begin{equation} χ^2 = \sum \dfrac{(O-E)^2}{E} \end{equation}

4.2 Examples

In order to demonstrate how to perform the test in RStudio, let's make up a fictional data set. Say that a bag of M&M is expected to have 10 black ones, 10 red ones, 10 blue ones, and 10 yellow ones. We observe that it has 8 black ones, 13 red ones, 11 blue ones, and 8 yellow ones. The data can be summarized and graphed as the following:

```r
# Create the corresponding vectors and matrix
name = c('black', 'red', 'blue', 'yellow')
Expected <- c(10, 10, 10, 10)
Observed <- c(8, 13, 11, 8)
M_and_M <- rbind(Expected, Observed)
colnames(M_and_M) = name

# Graph the expected and observed numbers of candies
barplot(M_and_M, beside = T, xlab = 'Color', ylab = 'Count',
        col = c('gray48', 'gray81'))
legend('topright', legend = rownames(M_and_M), cex = 0.75,
       fill = c('gray48', 'gray81'))
```



```
# Perform the Chi-square test using chisq.test() function
# rescale.p rescales the counts before applying the test
chisq.test(M_and_M, rescale.p = T)
```

```
##
##  Pearson's Chi-squared test
##
## data:  M_and_M
## X-squared = 0.88337, df = 3, p-value = 0.8294
```

As shown, the χ^2 value is 0.88337, the degree of freedom is 3, and the p-value is 0.8294. Since the p-vaue is greater than the alpha level, we fail to reject the null hypothesis.

5. Take Home Message

Doing statistical hypothesis tests is easy in R. There are usually packages such as t.test() or chisq.test() specifically designed for such tests. The results can also be easily formatted and visualized by the techniques above.

6. References

https://www.unm.edu/~marcusj/2Sampletex1.pdf

http://www.cookbook-r.com/Manipulating_data/Summarizing_data/

http://www.cookbook-r.com/Graphs/Plotting_means_and_error_bars_(ggplot2)/

https://en.wikipedia.org/wiki/Statistical_hypothesis_testing

http://www.itl.nist.gov/div898/handbook/eda/section3/eda353.htm

https://ncss-wpengine.netdna-ssl.com/wp-content/themes/ncss/pdf/Procedures/NCSS/Two-Sample_T-Test.pdf

http://blog.minitab.com/blog/adventures-in-statistics-2/understanding-hypothesis-tests%3A-significance-levels-alpha-and-p-values-in-statistics

https://statistics.berkeley.edu/computing/r-t-tests

http://www.mas.ncl.ac.uk/~ndw/teaching/MAS1403/notes4.pdf

/rcompanion.org/rcompanion/b_03.html