Confidence intervals and hypothesis tests

R provides an extensive range of tools for doing inferential statistics. The most basic of these include confidence intervals and hypothesis tests for one sample proportions.

Before getting into details of these methods, it is useful to learn how to work with normal distributions in R. The following examples show how to find normal probabilities (i.e., area under the normal curve) for any specified range of z-values.

Example 1: Find the following probabilities in the standard normal distribution:

(a) $P\ (z < -1.6789)$.

(b) $P\ (z > -1.6789)$.

(c) $P\ (|z| < 0.78)$.

(d) $P\ (-0.98 < z < 1.279)$

</FONT>

We will use a function named `pnorm`. For more information, type `?pnorm` to see the full set of options it offers.

In [1]:
```
# Example 1: (a) z < -1.6789
pnorm(-1.6789, lower.tail=TRUE)

# Example 1: (b) z > -1.6789
pnorm(-1.6789, lower.tail=FALSE)
# OR, another way is to leave out the "lower.tail" option and just
take the complement.
# Here's what that would look like if you want to try it:
# 1-pnorm(-1.6789)

# Example 1: (c) |z| < 0.78
pnorm(0.78) - pnorm(-0.78)
# NOTE that lower.tail=TRUE is the default.  So we can
# leave it out for brevity.

# Example 1: (d) -0.98 < z < 1.279
pnorm(1.279) - pnorm(-0.98)
```

0.0465857672770096

0.95341423272299

0.564609124828534

0.736008412726515

Next, let us look at how to do reverse lookups.

Example 2: Find the z-score corresponding to the following areas in the standard normal distribution:

(a) Area in the upper tail is 0.0845 .

(b) Area in lower tail is 0.404.

(c) Want z corresponding to the central 48% area.

We will use a function named `qnorm`, which returns the z-value based on accumulating areas from the left-end (i.e., in the lower tail). Thus, for each situation we must figure out the correct input to give

`qnorm` so that it gives us what we want.

In [2]:
```
# Example 2: (a) upper tail contains 0.0845 of the area
qnorm(1-0.0845)   # since it is in the upper tail, must do 1-0.0845

# Example 2: (b) lower tail contains 0.404 of the area
qnorm(0.404)

# Example 2: (c) Want z corresponding to the central 48% area
qnorm(0.48 + (1-0.48)/2)   # central 48% = (48 + 52/2)% area to the
left
```

```
1.37542410526545

-0.243006967409982

0.643345405392917
```

Confidence intervals for one sample proportion

Suppose we have data on a categorical variable that we can treat as having only two sides (e.g., yes or no; success or failure). Let n=sample size, and x=number of successes in the sample. Then the function named `prop.test` can be used to compute a confidence interval for the proportion of successes in the sample. The simplest usage of the function has the form: `prop.test(x, n, conf.level=k)` where k is the level of confidence we want.

Example 3: Pew Research polled a random sample of 900 U.S. teens about their Internet use. Sixty percent of those teens admitted they had misrepresented their age online to access websites and online services. Compute and interpret a 90% confidence interval for the proprtion of U.S. teens who have misrepresented their age online.

In [3]:
```
# Example 3: Here we have n=900, x=0.6*900
# The shortest (but less clear) way to do this is:
prop.test(0.6*900, 900, conf.level=0.9)
```

```
        1-sample proportions test with continuity correction

data:  0.6 * 900 out of 900, null probability 0.5
X-squared = 35.601, df = 1, p-value = 2.421e-09
alternative hypothesis: true p is not equal to 0.5
90 percent confidence interval:
 0.5723185 0.6270697
sample estimates:
  p
0.6
```

In [16]:
```
# For more clarity, use the following form:
n = 900    # sample size
x = 0.6*n  # number of successes
myout = prop.test(x, n, conf.level=0.9)
cat("The confidence interval = [", myout$conf.int, "]")    # print
left/right ends of CI
#diff(myout$conf.int)  # uncomment this to get width of CI
cat("\nInterpretation: We are 90% confident that the true proportio
n of U.S. teens who have misrepresented their age online to access
websites and services lies between", myout$conf.int[1], "and", myou
t$conf.int[2])
```

```
The confidence interval = [ 0.5723185 0.6270697 ]
Interpretation: We are 90% confident that the true proportion of U.
S. teens who have misrepresented their age online to access websites
and services lies between 0.5723185 and 0.6270697
```

To see a more general description of `prop.test` use R's builtin help utility by typing

```
?prop.test
```

and "run" it.

Hypothesis tests with one sample proportion

The same `prop.test` function used for confidence intervals can also be used for hypothesis testing, as shown in the following example.

Example 4: In Example 3 above we saw that Pew Research found 60% of a random sample of 900 teens admitted they had misrepresented their age online to access websites and online services. Extending that scenario, suppose we want to test the hypothesis that more than 55% of all teens misrepresent their age online. Carry out the test and find the P-value.

In [5]:
```
n = 900    # sample size
x = 0.6*n  # number of successes
p0 = 0.55  # null hypothesis value of the proportion
prop.test(x, n, p0, alternative="greater")
#
# OR, we can do it this way
#
myout = prop.test(x, n, p0, alternative="greater")
cat("The P-value=", myout$p.value)
```

```
        1-sample proportions test with continuity correction

data:  x out of n, null probability p0
X-squared = 8.89, df = 1, p-value = 0.001434
alternative hypothesis: true p is greater than 0.55
95 percent confidence interval:
 0.5723185 1.0000000
sample estimates:
  p
0.6

The P-value= 0.001433675
```

Student t distribution lookups

It is fairly straightforward to "lookup" values of a student t distribution with any specified degrees of freedom.

Example 5: Compute each of the following values for the indicated t distribution:

(a) $P\ (t < 1.967)$ with 5 df.

(b) $P\ (t > 1.967)$ with 5 df.

(c) The t-value where 97.5% area is to the left, with 5 df.

(This is the same as the t^*_5 value for a 95% confidence interval.)

(d) The t-value where 97.5% area is to the left, with 23 df.

In [6]:
```
pt(1.967, df=5)  # Find area under t-curve with 5 degrees of freedo
m for t < 1.967
pt(1.967, df=5, lower.tail=FALSE)    # Find area under same t-curve
for t > 1.967
qt(0.975, df=5)   # Inverse lookup: find t-value at 97.5 percentile
point with 5 df
qt(0.975, df=23)   # Inverse lookup: find t-value at 97.5 percentil
e point with 23 df
```

0.946834355069976

0.0531656449300238

2.57058183563631

2.06865761041905

Inferences with one sample mean

It is easiest to do confidence intervals and hypothesis tests with sample mean values if you first create a dataframe or variable containing your raw data. Once you have this, the function named `t.test` can be used for computing confidence intervals and hypothesis tests.

Example 6a: The file named "winter.csv" contains the following 4 variables: (1) name of a U.S. city; (2) the mean January temperatures in that city (in degrees F); (3) the latitude of the city; and (4) the January temperature in degrees c.

Suppose we treat those data as a random sample of cities drawn from a population that consists of all cities in the U.S. Compute a 90% confidence interval for the true mean latitude of U.S. cities.

In [8]:
```
# Read the csv file
winterdat = read.csv(file="./winter.csv", header=TRUE, sep=",")

# You can see names of the variables in that file
# by uncommenting the next line
#winterdat

# Next, define the variable you want to do inference with
x = winterdat$Latitude
t.test(x, conf.level=0.9)    # Compute a 90% CI using "x" as input
data

# If you want to see a cleaner output, use:
myout = t.test(x, conf.level=0.9)
cat("The confidence interval = [", myout$conf.int, "]")    # print
left/right ends of CI
```

```
        One Sample t-test

data:  x
t = 74.436, df = 58, p-value < 2.2e-16
alternative hypothesis: true mean is not equal to 0
90 percent confidence interval:
 38.35037 40.11235
sample estimates:
mean of x
 39.23136

The confidence interval = [ 38.35037 40.11235 ]
```

Example 6b: Continuing with data from the previous example, let us test the hypothesis that the true mean latitude of U.S. cities is no more than 38 deg. Find the P-value and draw an inference.

In [9]:
```
x = winterdat$Latitude
myout = t.test(x, mu=38, alternative="greater")
cat("The P-value=", myout$p.value)
```

```
The P-value= 0.01147574
```

Example 6b continued: The P-value is about 1.1%. If we assume a significance level of 5% (which for a 1-tailed hypothesis test is consistent with a 90% confidence level), we would reject the null hypothesis and conclude that the true mean latitude of U.S. cities is less than 38 degrees.

To see a more general description of `t.test` use R's builtin help utility by typing

```
?t.test
```

and "run" it.

Your homework exercises

The file starbucks.csv contains nutrition information on a bunch of items sold at a sample of Starbucks locations in the U.S. The variables are clearly labeled in the file (the units are grams for fat, carbohydrates, fiber and protein). Follow the instructions below to download the file from openintro.org. After that, carry out the following inference tasks:

1. Use a confidence interval to estimate the true proportion of bakery items at a typical Starbucks outlet.
2. Estimate the true mean grams of fat in items sold at Starbucks locations using a confidence interval.
3. Estimate the true mean grams of carbohydrates in items sold at Starbucks using a confidence interval.
4. Is the true mean calorie content in a typical Starbucks food item more than 300? Carry out a hypothesis test to find out.
5. Compute a matching confidence interval to check whether the mean calorie content is more than 300.
6. Is the true mean fat content in a typical Starbucks food item below 15 grams? Use a hypothesis test to infer a conclusion.

The goal of this exercise is to use R to do all the computations. However, please be sure your solution includes all the steps needed for carrying out valid inference procedures, and state a complete and correct conclusion.

In [1]:
```
download.file("https://www.openintro.org/data/csv/starbucks.csv", d
estfile="sbux.csv")
sdata = read.csv(file="./sbux.csv", header=TRUE, sep=",")
head(sdata)
```

A data.frame: 6 × 7

	item	calories	fat	carb	fiber	protein	type
	<fct>	<int>	<dbl>	<int>	<int>	<int>	<fct>
1	8-Grain Roll	350	8	67	5	10	bakery
2	Apple Bran Muffin	350	9	64	7	6	bakery
3	Apple Fritter	420	20	59	0	5	bakery
4	Banana Nut Loaf	490	19	75	4	7	bakery
5	Birthday Cake Mini Doughnut	130	6	17	0	0	bakery
6	Blueberry Oat Bar	370	14	47	5	6	bakery

In [11]:
```
t.test(sdata$calorie, mu=300, alternative="greater")
```

```
        One Sample t-test

data:  sdata$calorie
t = 3.2338, df = 76, p-value = 0.0009043
alternative hypothesis: true mean is greater than 300
95 percent confidence interval:
 318.8362      Inf
sample estimates:
mean of x
 338.8312
```

In [ ]: