Introduction to Probability

Chapter 2, Lab 1: Template

OpenIntro Biostatistics

Topics

- Defining probability
- Rules of probability
- Simulation

Probabilities for events can be calculated via simulation by simply repeating an experiment a large number of times and then counting the number of times the event of interest occurs. According to the Law of Large Numbers, as the number of repetitions increase, the proportion $\hat{p}_n$ of occurrences converges to the probability p of that event.

The material in this lab corresponds to Section 2.1 of *OpenIntro Biostatistics*.

Introduction

Suppose that a fair coin is tossed 5 times. What is the probability of obtaining exactly 3 heads?

Before writing the code for a simulation to estimate probability, it is helpful to clearly define the experiment and event of interest. In this case, the experiment is tossing a fair coin 5 times and the event of interest is observing exactly 3 heads. Since the coin is fair, the probability of obtaining a heads is 0.5.

1. The following code illustrates the use of the `sample()` command to simulate the result for one set of 5 coin tosses.

```
#define parameters
prob.heads =
number.tosses =

#simulate the coin tosses
outcomes = sample(c(0, 1), size = number.tosses,
                      prob = c(1 - prob.heads, prob.heads), replace = TRUE)

#view the results
table(outcomes)

#store the results as a single number
total.heads = sum(outcomes)
total.heads
```

a) Using the information given about the experiment, set the parameters for `prob.heads` and `number.tosses` and run the code chunk.

b) To generate `outcomes`, the `sample()` command draws from the values `0` and `1` with probabilites corresponding to those specified by the argument `prob`. Which number corresponds to heads, and which corresponds to tails?

c) Why is it important to sample with replacement?

d) What is the advantage of representing the outcomes with the values `0` and `1`, rather than with letters like "T" and "H"?

e) Run the code chunk again to simulate another set of 5 coin tosses. Is it reasonable to expect that the results might differ from the first set of tosses?

2. The following code uses a `for` loop to repeat (i.e., replicate) the experiment and record the results of each replicate. The term `k` is an index, used to keep track of each iteration of the loop; think of it as similar to the index of summation k (or i) in sigma notation ($\sum_{k=1}^{n}$). The value `num.replicates` is set to 50, specifying that the experiment is to be repeated 50 times.

   The command `set.seed()` is used to draw a reproducible random sample; i.e., re-running the chunk will produce the same set of outcomes.

```
#define parameters
prob.heads = 0.5
number.tosses = 5
number.replicates = 50

#create empty vector to store outcomes
outcomes = vector("numeric", number.replicates)

#set the seed for a pseudo-random sample
set.seed(2018)

#simulate the coin tosses
for(k in 1:number.replicates){

  outcomes.replicate = sample(c(0, 1), size = number.tosses,
                      prob = c(1 - prob.heads, prob.heads), replace = TRUE)

  outcomes[k] = sum(outcomes.replicate)
}

#view the results
addmargins(table(outcomes))

heads.3 = (outcomes == 3)
table(heads.3)
```

a) The parameters of the experiment have already been filled in; the probability of heads remains 0.5 and the number of tosses is set to 5. This code repeats the experiment 50 times, as specified by `number.replicates`. Run the code chunk.

b) How many heads were observed in the fourth replicate of the experiment? Hint: look at

`outcomes`.

c) Out of the 50 replicates, how often were exactly 3 heads observed in a single experiment?

d) From the tabled results of the simulation, calculate an estimate of the probability of observing exactly 3 heads when a coin is tossed 5 times.

e) Using the laws of probability, calculate the exact probability of observing exactly 3 heads when a coin is tossed 5 times.

f) Compare the estimate from the simulation to the exact probability. How close are the two values?

g) Re-run the simulation with 100 replicates of the experiment; calculate a new estimate of the probability of observing exactly 3 heads when a coin is tossed 5 times. How close is this new estimate to the exact probability calculated in part e)? How close is an estimate based on 1,000 replicates to the exact probability calculated in part e)?

Mandatory Drug Testing

The simulation framework illustrated in the previous section can easily be adapted for other scenarios that may seem more complicated than coin tossing. The drug testing scenario examined in this section appears in *OI Biostat* as Example 2.30.

Mandatory drug testing in the workplace is common practice for certain professions, such as air traffic controllers and transportation workers. A false positive in a drug screening test occurs when the test incorrectly indicates that a screened person is an illegal drug user. Suppose a mandatory drug test has a false positive rate of 1.2% (i.e., has probability 0.012 of indicating that an employee is using illegal drugs when that is not the case). Given 150 employees who are in reality drug free, what is the probability that at least one will (falsely) test positive? Assume that the outcome of one drug test has no effect on the others.

3. Calculate the probability using an algebraic approach.

4. Calculate an estimate of the probability via simulation.

    a) Before writing the code to run repeated simulations, it is helpful to first think about the problem structure and how to model one run of the experiment with the `sample()` command.

        i. What are the two possible outcomes when a person is tested?

        ii. What are the associated probabilites of the possible outcomes?

        iii. Describe a reasonable way to represent the two possible outcomes with the values `0` and `1`.

    b) Simulate the result for one set of 150 tests. Use `set.seed()` so that the results are reproducible.

        i. Was there at least one employee who tested positive?

        ii. What is the proportion of employees who tested positive for drug use?

c) Based on 100,000 replicates, estimate the probability that at least one employee tests positive out of 150 drug-free employees.

Mammograms

5. The specificity of a diagnostic test refers to the probability that a test is negative in the absence of disease. Mammograms have a specificity of 95% for detecting breast cancer.

   a) Define the relationship between the specificity of a test and the probability of a false positive.

   b) Suppose a clinic conducts approximately 50 mammograms in a week. Use simulation to calculate the probability that no more than 1 woman will test positive if none of the women have breast cancer.

   c) Re-calculate the probability described in a) if the probability of a false positive result is 0.01.

```
#define parameters
prob.false.positive = 0.012
number.employees = 150
number.replicates = 100000

#create empty vector to store results
results = vector("numeric", number.replicates)

#set the seed for a pseudo-random sample
set.seed(2018)

#simulate the tests
for(k in 1:number.replicates){

  results.replicate = sample(c(0,1), size = number.employees,
                             prob = c(1 - prob.false.positive, prob.false.positive),
                                      replace = TRUE)

  results[k] = sum(results.replicate)

}

#view the results
table(results)
```

```
## results
##     0     1     2     3     4     5     6     7     8     9    10
## 16282 29847 27015 16201  7183  2524   720   169    49     7     3
```

```
at.least.1.pos = (results >= 1)
table(at.least.1.pos)
```

```
## at.least.1.pos
```

```
## FALSE  TRUE
## 16282 83718
```

```
#define parameters
prob.false.positive = 0.05
number.women = 50
number.replicates = 100000

#create empty vector to store results
results = vector("numeric", number.replicates)

#set the seed for a pseudo-random sample
set.seed(2018)

#simulate the tests
for(k in 1:number.replicates){

  results.replicate = sample(c(0,1), size = number.women,
                             prob = c(1 - prob.false.positive, prob.false.positive),
                                      replace = TRUE)

  results[k] = sum(results.replicate)

}

#view the results
table(results)
```

```
## results
##     0     1     2     3     4     5     6     7     8     9    10    11
##  7858 20162 26036 21929 13560  6716  2568   855   237    66    11     2
```

```
at.most.1.pos = (results <= 1)
table(at.most.1.pos)
```

```
## at.most.1.pos
## FALSE  TRUE
## 71980 28020
```