Contents

This chapter focuses on the material covered in *OI Biostat* Chapter 2, Section 2.5. While simple conditional probability problems are easily solved by hand with a basic calculator, R can be useful for more elaborate scenarios like those that involve Bayes' Theorem.

What are the chances that a woman with a positive mamogram has breast cancer? This question can be rephrased as the conditional probability that a woman has breast cancer, given that her mammogram is abnormal, otherwise known as the **positive predictive value** of a mammogram. Two methods discussed in this chapter, using Bayes' Theorem and creating a contingency table, are also explained in *OI Biostat*. This chapter will also cover an additional approach: modeling the problem scenario by running a simulation.

> ***OI Biostat* Example 2.37.** In Canada, about 0.35% of women over 40 will develop breast cancer in any given year. A common screening test for cancer is the mammogram, but it is not perfect. In about 11% of patients with breast cancer, the test gives a **false negative**: it indicates a woman does not have breast cancer when she does have breast cancer. Similarly, the test gives a **false positive** in 7% of patients who do not have breast cancer: it indicates these patients have breast cancer when they actually do not. If a randomly selected woman over 40 is tested for breast cancer using a mammogram and the test is positive – that is, the test suggests the woman has cancer – what is the probability she has breast cancer?

1 Bayes' Theorem

Bayes' Theorem states that the conditional probability $P(A_1|B)$ can be identified as the following fraction:

$$\frac{P(A_1 \text{and } B)}{P(B)} = \frac{P(B|A_1)P(A_1)}{P(B|A_1)P(A_1) + P(B|A_2)P(A_2) + \cdots + P(B|A_k)P(A_k)}$$

where A_2, A_3, ..., and A_k represent all other possible outcomes of the first variable.

The expression can also be written in terms of diagnostic testing language, where $D = \{\text{has disease}\}$, $D^c = \{\text{does not have disease}\}$, $T^+ = \{\text{positive test result}\}$, and $T^- = \{\text{negative test result}\}$.

$$\begin{aligned} P(D|T^+) &= \frac{P(D \text{ and } T^+)}{P(D)} \\ &= \frac{P(T^+|D) \times P(D)}{[P(T^+|D) \times P(D)] + [P(T^+|D^c) \times P(D^c)]} \\ \text{PPV} &= \frac{\text{sensitivity } \times \text{ prevalence}}{[\text{sensitivity } \times \text{ prevalence}] + [(1 \text{ - specificity}) \times (1 \text{ - prevalence})]} \end{aligned}$$

R can be used to store values for prevalence, sensitivity, and specificity so that calculations are less tedious. Recall that the **sensitivity** is the probability of a positive test result when disease is present, which is the complement of a false negative. The **specificity** is the probability of a negative test result when disease is absent, which is the complement of a false positive.

```
prevalence = 0.0035
sensitivity = 1 - 0.11
specificity = 1 - 0.07

ppv.num = (sensitivity*prevalence)  ## numerator
ppv.den = ppv.num + ((1-specificity)*(1-prevalence))  ## denominator
ppv = ppv.num / ppv.den

ppv

## [1] 0.04274736
```

2 Contingency Table

The PPV can also be calculated by constructing a two-way contingency table for a hypothetical population and calculating conditional probabilities by conditioning on rows or columns. While this method results in an estimate of PPV, using a large enough population size such as 100,000 produces an empirical estimate that is very close to the exact value found through using Bayes' Theorem.

	D+	D-	Total
T+			
T-			
Total			100,000

First, calculate the expected number of disease cases and non-disease cases in the population:

```
population.size = 100000
expected.cases = prevalence * population.size
expected.cases

## [1] 350

expected.noncases = (1 - prevalence) * population.size
expected.noncases

## [1] 99650
```

	D+	D-	Total
T+			
T-			
Total	350	99,650	100,000

Next, calculate the expected number of cases of true positives and the expected number of cases of false positives:

```
expected.true.positives = expected.cases * sensitivity
expected.true.positives

## [1] 311.5

expected.false.positives = expected.noncases * (1 - specificity)
expected.false.positives

## [1] 6975.5

total.expected.positives = expected.true.positives + expected.false.positives
total.expected.positives

## [1] 7287
```

	D+	D-	Total
T+	311.5	6,975.5	7,287
T-			
Total	350	99,650	100,000

Finally, calculate the positive predictive value:

```
ppv = expected.true.positives/total.expected.positives
ppv

## [1] 0.04274736
```

3 Simulation

R can be used to simulate a population of 100,000 individuals that fits the parameters specified by the problem, i.e., a population where 0.35% of women have breast cancer. Afterwards, using the known specificity and sensitivity of the diagnostic test, individuals can be assigned a test result of either positve or negative. This results in a simulated dataset of 100,000 individuals that each have a disease status and test result.

1. Define the parameters of the simulation: disease prevalence, test sensitivity, test specificity, and hypothetical population size.

2. In order for the results of the simulation to be reproducible, it is necessary to use `set.seed()` to associate the particular set of results with a "seed". Any integer can be used as the seed. Different seeds will produce a different set of results.

3. Using the `vector()` command, create two empty lists that will be used to store the results of the simulation: one will store disease status and the other will store test outcome. The results will be in the form of numbers; specifically, either `0` or `1`.

4. Assign disease status by using the `sample()` command, which takes a sample of a specified size from a list `x`. In this context, the goal is to sample between `0` and `1` 100,000 times, where `0` represents an individual without breast cancer and `1` an individual with breast cancer. The argument `prob` defines the probability that either number is sampled; a `1` should be sampled with the same probability as the prevalence, since the prevalence indicates how many members of the population have disease. Similarly, a `0` should be sampled with probability `(1 - prevalence)`. The argument `replace = TRUE` allows for sampling with replacement; in other words, allowing the numbers `0` and `1` to be assigned multiple times.

5. Assign test result by using the `sample()` command in combination with a `for` loop and conditional statements. In this step, a test result is assigned with different probability depending on whether the disease status is a `0` or a `1`; this relates to the sensitivity and specificity of the diagnostic test. The loop allows for this process to be repeated for each of the 100,000 individuals being simulated.

   - The first line sets up the loop, with a variable `ii` that starts at `1` and finishes at `population.size`, which was specified earlier as 100,000. This allows for each test result to be assigned one at a time.
   - Two `if` statements are in the loop which direct R to take a different action depending on the value of `disease.status`. The double equals signs == imply a conditional statement, allowing `if(disease.status[ii] == 1)` to make the statement "For this loop, if disease status is equal to 1, then do the following...".
   - Depending on disease status, R will execute one of the two `sample()` functions in the loop. One sample function has probabilities weighted based on sensitivity and the other has probabilities defined by specificity.

6. Make calculations using the two lists, `disease.status` and `disease.outcome`, which are now filled with values. Since the value `1` was assigned to a positive test result and to an individual with disease, the `sum()` command can be used to determine the total number of individuals with disease and the number of positive test results.

```
## 1. set parameters
prevalence = 0.0035
sensitivity = 1 - 0.11
specificity = 1 - 0.07
population.size = 100000
```

```r
## 2. set seed
set.seed(2016)

## 3. create empty lists
disease.status = vector("numeric", population.size)
test.outcome = vector("numeric", population.size)

## 4. assign disease status
disease.status = sample(x = c(0,1), size = population.size,
                        prob = c(1 - prevalence, prevalence),  ## matches order of x
                        replace = TRUE)

## 5. assign test result
for (ii in 1:population.size) {
  if (disease.status[ii] == 1) {  ## note the ==
    test.outcome[ii] = sample(c(0, 1), size = 1, ## test results assigned 1 at a time
                              prob = c(1 - sensitivity, sensitivity))}
  if (disease.status[ii] == 0) {  ## note the ==
    test.outcome[ii] = sample(c(0, 1), size = 1, ## test results assigned 1 at a time
                              prob = c(specificity, 1 - specificity))}

}

## 6. calculate ppv
num.disease = sum(disease.status)  ## total number of individuals with disease
num.pos.test = sum(test.outcome)   ## total number of positive tests

# identify the number of individuals with disease who tested positive
num.true.pos = sum(test.outcome[disease.status == 1])

# calculate ppv
ppv = num.true.pos / num.pos.test
ppv

## [1] 0.04273973
```