AE 12: Conditional Probability

Dav King

2/15/2022

```
library(tidyverse)
library(knitr)
```

```
sta199 <- read_csv("sta199-fa21-year-major.csv")
```

Learning goals

- Define marginal, joint, and conditional probabilities, and calculate each "manually" and in a reproducible way
- Identify whether two events are independent
- Apply Bayes' theorem using the Hypothetical 10,000

Coming Up

- Lab 5 due Friday.
- Homework 3 assigned Thursday.
- Prep Quiz Due by 11:59 PM today

Definitions

Let A and B be events.

- **Marginal probability:** The probability an event occurs regardless of values of the other event
- *P(A)* or *P(B)*
- **Joint probability**: The probability two or more simultaneously occur
- *P(A and B)*
- **Conditional probability**: The probability an event occurs given the other has occurred
- *P(A|B)* or *P(B|A)*
- **Independent events**: Knowing one event has occurred does not lead to any change in the probability we assign to another event.
- *P(A|B) = P(A)* or *P(B|A) = P(B)*

Part 1: STA 199 years & majors

For this portion of the AE, we will continue using the data including the year in school and majors for students taking STA 199 in Fall 2021, i.e., you! The data set includes the following variables:

- `section`: STA 199 section
- `year`: Year in school
- `major_category`: Major / academic interest.
  - For the purposes of this AE, we'll call this the student's "major".

Let's start with the contingency table from the last class:

```
sta199 %>%
  count(year, major_category) %>%
  pivot_wider(id_cols = c(year, major_category),#how we identify unique obs
              names_from = major_category, #how we will name the columns
              values_from = n, #values used for each cell
              values_fill = 0) %>% #how to fill cells with 0 observations
  kable() # neatly display the results
```

year	compsci only	econ only	other	pubpol only	stat + other major	stats only	undecided
First-year	8	6	39	22	26	7	5
Junior	7	3	12	4	1	0	0
Senior	2	0	5	1	1	0	0
Sophomore	23	6	42	11	8	3	5

Try to answer the questions below using the contingency table and using code to answer in a reproducible way.

Part A: What is the probability a randomly selected STA 199 student is studying a subject in the "other" major category?

```
sta199 %>%
  count(major_category) %>%
  mutate(pMajor = n/sum(n))
```

```
## # A tibble: 7 x 3
##   major_category         n pMajor
##   <chr>              <int>  <dbl>
## 1 compsci only          40 0.162
## 2 econ only             15 0.0607
## 3 other                 98 0.397
## 4 pubpol only           38 0.154
## 5 stat + other major    36 0.146
## 6 stats only            10 0.0405
## 7 undecided             10 0.0405
```

0.397

Part B: What is the probability a randomly selected STA 199 student is a first-year?

```
sta199 %>%
  count(year) %>%
  mutate(pYear = n/sum(n))
```

```
## # A tibble: 4 x 3
##   year           n  pYear
##   <chr>      <int>  <dbl>
## 1 First-year   113 0.457
## 2 Junior        27 0.109
## 3 Senior         9 0.0364
## 4 Sophomore     98 0.397
```

0.457

Part C: What is the probability a randomly selected STA 199 student is a first year and is studying a subject in the "other" major category?

```
sta199 %>%
  mutate(frosh_other = if_else(
    year == "First-year" & major_category == "other", 1, 0)) %>%
  summarize(prop_frosh_other = mean(frosh_other))
```

```
## # A tibble: 1 x 1
##   prop_frosh_other
##              <dbl>
## 1            0.158
```

Part D: What is the probability a randomly selected STA 199 student is a first year given they are studying a subject in the "other" major category?

```
sta199 %>%
  filter(major_category == "other") %>%
  count(year) %>%
  mutate(prop_year_other = n/sum(n))
```

```
## # A tibble: 4 x 3
##   year           n prop_year_other
##   <chr>      <int>           <dbl>
## 1 First-year    39          0.398
## 2 Junior        12          0.122
## 3 Senior         5          0.0510
## 4 Sophomore     42          0.429
```

0.398

Part E: What is the probability a randomly selected STA 199 student is studying a subject in the "other" major category given they are a first-year?

```
sta199 %>%
  filter(year == "First-year") %>%
  count(major_category) %>%
  mutate(prop_other_firstyear = n/sum(n))
```

```
## # A tibble: 7 x 3
##   major_category         n prop_other_firstyear
##   <chr>              <int>                <dbl>
## 1 compsci only           8               0.0708
## 2 econ only              6               0.0531
## 3 other                 39               0.345
## 4 pubpol only           22               0.195
## 5 stat + other major    26               0.230
## 6 stats only             7               0.0619
## 7 undecided              5               0.0442
```

0.345

Part F: Are being a first-year and studying a subject in the "other" category independent events? Briefly explain.

No. If they were independent events, then P(A) should = P(A|B) and P(B) should = P(B|A). However, the probability of a student studying an "other" major is 0.397, while that proportion within first years is 0.345. Though we don't need more evidence, the proportion of 1st years is 0.457, but the proportion of 1st years within the "other" major category is 0.398. Since these probabilities are not equal, we cannot treat them as independent. Similarly, and our best test for independence, P(A & B) should = P(A) x P(B). P(A & B) = 0.158, but P(A) x P(B) = 0.181, a very different number that rejects the possibility of independence.

Part 2: Bayes' Theorem

Monty Hall Problem:

A Video: https://brilliant.org/wiki/monty-hall-problem/.

"Suppose you're on a game show, and you're given the choice of three doors: Behind one door is a car; behind the others, goats. You pick a door, say No. 1, and the host, who knows what's behind the doors, opens another door, say No. 3, which has a goat. He then says to you,"Do you want to pick door No. 2?" Is it to your advantage to switch your choice?"

We will investigate the above decision of whether to switch or not to switch.

Assumptions:

The host will always open a door not picked by the contestant.

The host will always open a door which reveals a goat (i.e. not a car).

The host will always offer the contestant the chance to switch to another door.

The door behind which the car is placed is chosen at random.

The door initially chosen by the contestant is chosen at random.

```
doors <- c(1, 2, 3)
```

```
monty_hall <- tibble(
  car_door = sample(doors, size = 10000, replace = TRUE),
  my_door = sample(doors, size = 10000, replace = TRUE)
  )
monty_hall
```

```
## # A tibble: 10,000 x 2
##    car_door my_door
##       <dbl>   <dbl>
##  1        3       1
##  2        3       3
##  3        3       1
##  4        1       3
##  5        2       1
##  6        2       3
##  7        2       2
##  8        2       1
##  9        1       3
## 10        3       1
## # ... with 9,990 more rows
```

```
monty_hall <- monty_hall %>%
  rowwise() %>%
  mutate(monty_door = if_else(car_door == my_door,
                              sample(doors[-my_door], size = 1),
                              6 - (car_door + my_door))) %>%
  ungroup()
monty_hall
```

```
## # A tibble: 10,000 x 3
##    car_door my_door monty_door
##       <dbl>   <dbl>      <dbl>
##  1        3       1          2
##  2        3       3          2
##  3        3       1          2
##  4        1       3          2
##  5        2       1          3
##  6        2       3          1
##  7        2       2          3
##  8        2       1          3
##  9        1       3          2
## 10        3       1          2
## # ... with 9,990 more rows
```

```
monty_hall <- monty_hall %>%
  mutate(switch_win = car_door != my_door,
         stay_win   = car_door == my_door)
monty_hall
```

```
## # A tibble: 10,000 x 5
##    car_door my_door monty_door switch_win stay_win
##       <dbl>   <dbl>      <dbl> <lgl>      <lgl>
##  1        3       1          2 TRUE       FALSE
##  2        3       3          2 FALSE      TRUE
##  3        3       1          2 TRUE       FALSE
##  4        1       3          2 TRUE       FALSE
##  5        2       1          3 TRUE       FALSE
##  6        2       3          1 TRUE       FALSE
##  7        2       2          3 FALSE      TRUE
```

```
##  8        2       1          3 TRUE       FALSE
##  9        1       3          2 TRUE       FALSE
## 10        3       1          2 TRUE       FALSE
## # ... with 9,990 more rows
```

```
monty_hall %>%
  summarise(switch_win_prob = mean(switch_win),
            stay_win_prob   = mean(stay_win))
```

```
## # A tibble: 1 x 2
##   switch_win_prob stay_win_prob
##             <dbl>         <dbl>
## 1           0.658         0.342
```

Some Practice using the Hypothetical 10,000

The global coronavirus pandemic illustrates the need for accurate testing of COVID-19, as its extreme infectivity poses a significant public health threat. Due to the time-sensitive nature of the situation, the FDA enacted emergency authorization of a number of serological tests for COVID-19 in 2020. Full details of these tests may be found on its website here.

We will define the following events:

- **Pos**: The event the Alinity test returns positive.
- **Neg**: The event the Alinity test returns negative.
- **Covid**: The event a person has COVID
- **No Covid**: The event a person does not have COVID

The Abbott Alinity test has an estimated sensitivity of 100%, *P(Pos | Covid) = 1*, and specificity of 99%, *P(Neg | No Covid) = 0.99*.

Suppose the prevalence of COVID-19 in the general population is about 2%, *P(Covid) = 0.02*.

Bayes Theorem and the Hypothetical 10,000.

Part A: Use the Hypothetical 10,000 to calculate the probability a person has COVID given they get a positive test result, i.e. *P(Covid | Pos)*.

	Covid	No Covid	Total
Pos			
Neg			
Total			10000

```
covid <- c(1:100)

covid_table <- tibble(
  patient = sample(covid, size = 10000, replace = T),
  test = sample(covid, size = 10000, replace = T)
)
covid_table
```

```
## # A tibble: 10,000 x 2
##    patient  test
##      <int> <int>
##  1       3    38
##  2      59    92
##  3      17    11
##  4      34    75
##  5      77    87
##  6      54    60
##  7      54    38
##  8      75    86
##  9      65    65
## 10      95    19
## # ... with 9,990 more rows
```

```
covid_table2 <- covid_table %>%
  mutate(cov_status = if_else(patient > 2, "No Covid", "Covid")) %>%
  mutate(test_spec = case_when(
    cov_status == "No Covid" & test == 1 ~ "Positive",
    cov_status == "Covid" ~ "Positive",
    T ~ "Negative"
  ))

covid_table2 %>%
  count(cov_status, test_spec) %>%
  pivot_wider(id_cols = c(cov_status, test_spec),
              names_from = cov_status, values_from = n, values_fill = 0) %>%
  kable()
```

test_spec	Covid	No Covid
Positive	193	106
Negative	0	9701

```
covid_table2 %>%
  mutate(outcome = case_when(
   cov_status == "Covid" & test_spec == "Positive" ~ "hit",
   cov_status == "Covid" & test_spec == "Negative" ~ "miss",
   cov_status == "No Covid" & test_spec == "Positive" ~ "false positive",
   cov_status == "No Covid" & test_spec == "Negative" ~ "correct reject")) %>%
  count(outcome) %>%
  mutate(prob_outcome = n/sum(n)) %>%
  filter(outcome == "false positive" | outcome == "hit") %>%
  mutate(population = sum(n)) %>%
  filter(outcome == "hit") %>%
  summarize(answer = n/population)
```

```
## # A tibble: 1 x 1
##   answer
##    <dbl>
## 1  0.645
```

Note: I am aware a) that I didn't have to do this in code and b) that I couldn't figure out how to make it sum the totals (I tried hard, and would appreciate advice if anybody reads this). This is clunky code, but my reproducible answer is given above.

Part B: Use Bayes' Theorem to calculate *P(Covid/Pos)*.

```
covid_table2 %>%
  mutate(pos = if_else(cov_status == "No Covid", 0, 1)) %>%
  mutate(testPos = if_else(test_spec == "Positive", 1, 0)) %>%
  mutate(pPos = mean(pos)) %>%
  mutate(pTestPos = mean(testPos)) %>%
  filter(pos == 1) %>%
  mutate(pPosGivenCovid = mean(testPos)) %>%
  summarize(answer = (pPosGivenCovid * pPos)/pTestPos)
```

```
## # A tibble: 193 x 1
##    answer
##     <dbl>
##  1  0.645
##  2  0.645
##  3  0.645
##  4  0.645
##  5  0.645
##  6  0.645
##  7  0.645
##  8  0.645
##  9  0.645
## 10  0.645
## # ... with 183 more rows
```