SCEM Assignment 4

Vishal

2022-10-23

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.2 --
## v ggplot2 3.3.6      v purrr   0.3.5
## v tibble  3.1.8      v dplyr   1.0.10
## v tidyr   1.2.1      v stringr 1.4.1
## v readr   2.1.3      v forcats 0.5.2
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

1. Probability theory

1.1 (Q1)

$\Omega = \{a, b, c\}$ $\epsilon = \{A \subseteq B\}$ $\mathbb{P}(\{a\}) = 0.4$, $\mathbb{P}(\{b\}) = 0.2$, and $\mathbb{P}(\{c\}) = 0.3$ Then, $\mathbb{P}(\{a, b\}) = \mathbb{P}(\{a\}) + \mathbb{P}(\{b\}) = 0.6$ and $\mathbb{P}(\{b, c\}) = \mathbb{P}(\{b\}) + \mathbb{P}(\{c\}) = 0.5$

1.1 (Q2)

$\mathbb{P}(\{1\}) = 0.5$ Then, $\mathbb{P}(\{0\}) = 1 - 0.5 = 0.5$ Therefore, $\mathbb{P}(\{0, 1\}) = \mathbb{P}(\{1\}) + \mathbb{P}(\{0\}) = 1$

1.2 (Q1)

By Kolmogorov axioms, for pairwise disjoint events A1 and A2 in event space F,

$P(A_1 \cup A_2) = P(A_1) + P(A_2)$

Induction Hypothesis: Let A1,A2,…,An be pairwise disjoint events in F, i.e.

$A_i \cap A_j = \varnothing \quad \forall i, j = 1, 2, ..., n;\ i \neq j$

Then,

$P(\bigcup_{i=1}^{n} A_i) = \sum_{i=1}^{n} P(A_i)$

Now, I want to show that for pairwise disjoint events Ai = 1,2,…, n+1 $P(\bigcup_{i=1}^{n+1} A_i) = \sum_{i=1}^{n+1} P(A_i)$

Let $A_{n+1} \in F$ such that A_{n+1} is pairwise disjoint from events A_i, i = 1,2,…,n.

$B = \bigcup_{i=1}^{n} A_i$

Note that $B \cap A_{n+1} = \phi$ because if $x \in A_{n+1}$, then $x \notin A_i\ \forall i = 1, 2, ..., n \Rightarrow x \notin B$. On the other hand if $x \in B$ then $x \in A_i$ for exactly one i since A_i's are pairwise disjoint. A_{n+1} and A_i are also disjoint so $x \notin A_{n+1}$. We mentioned earlier that the probability of the union of two disjoint events is the sum of the probabilities of those events. Hence,

$P(B) \cup P(A_{n+1}) = P(B) + P(A_{n+1}) \Rightarrow (\bigcup_{i=1}^{n} A_i) \cup A_{n+1} = \sum_{i=1}^{n} P(A_i) + P(A_{n+1}) \Rightarrow (\bigcup_{i=1}^{n+1} A_i) = \sum_{i=1}^{n+1} P(A_i)$

Thus by induction, we have that for a finite collection of pairwise disjoint events A_1 through A_n in F, $P(\bigcup_{i=1}^{n} A_i) = \sum_{i=1}^{n} P(A_i)$

1.2 (Q2)

The complement rule can be derived from the axioms: the union of S(event) and its complement is Ω (Sample space)(either S happens or it does not, and there is no other possibility), so $\mathbb{P}(S \cup S^c) = \mathbb{P}(S) = 1$, by axiom 2. The event S and its complement are disjoint (if "S does not happen" happens, S does not happen; if S happens, "S does not happen" does not happen), so $\mathbb{P}(S \cup S^c) = \mathbb{P}(S) + \mathbb{P}(S^c)$ by axiom 3. Putting these together, we get $\mathbb{P}(S) + \mathbb{P}(S^c) = 1$. If we subtract $\mathbb{P}(S)$ from both sides of this equation, we get what we sought: $\mathbb{P}(S^c) = 1 - \mathbb{P}(S)$

1.2 (Q3)

if the weather forecast says that the chance of rain on Saturday is 40% and the chance of rain on Sunday is 10%, then the chance that it rains at some point during those two days is at least 40% and at most 50%.

To find the chance exactly, you would need the chance that it rains on both days, which you don't have. Assuming independence doesn't seem like a good idea in this setting. So you cannot compute an exact answer, and must be satisfied with bounds.

Though bounds aren't exact answers or even approximations, they can be very useful. Here is an example of a common use of Boole's Inequality in data science. It has Bonferroni's name attached to it, because Boole and Bonferroni both have related bounds on probabilities of unions.

click here

1.2 (Q4)

There is $A \cup (B \cap A^c) = (A \cup B) \cap (A \cup A^c) = A \cup B$, i.e. $A \cup B$ can be expressed as the union of two disjoint sets. Therefore, according to axiom 3, there is $\mathbb{P}(A \cup B) = \mathbb{P}(A) + \mathbb{P}(B \cap A^c)$. But $B = B \cap (A \cup A^c) = (B \cap A) \cup (B \cap A^c)$ is also the union of two disjoint sets, so there is also, $\mathbb{P}(B) = \mathbb{P}(B \cap A) + \mathbb{P}(B \cap A^c) \Rightarrow \mathbb{P}(B \cap A^c) = \mathbb{P}(B) - \mathbb{P}(B \cap A)$. Substituting the latter expression into the one above gives, $\mathbb{P}(A \cup B) = \mathbb{P}(A) + \mathbb{P}(B) - \mathbb{P}(A \cap B)$. . . . .

2. Finite probability spaces

2.1 (Q1)

Probability of picking red sphere from a bag containing 10 spheres is $\frac{1}{10}$. If we replace the sphere every time then the probability of drawing the red ball 22 times is,

$\mathbb{P}(A) = \frac{3}{10} * \frac{3}{10} * ... * \frac{3}{10}$ i.e. $\mathbb{P}(A) = \left(\frac{3}{10}\right)^{22}$ Probability that z out of 22 selections were red is can be given by **binomial distribution.** $P(X = z) = \binom{n}{z} p^k (1-p)^{n-z}$ i.e. for $z = 10$ $P(X = 10) = \binom{22}{10} \left(\frac{3}{10}\right)^k \left(\frac{7}{10}\right)^{12}$

2.1 (Q2)

```r
prob_red_spheres <- function(z) {
  return(choose(22, z) * ((0.3) ^ z) * ((0.7) ^ (22 - z)))
}

prob_red_spheres(10)
```

```
## [1] 0.05285129
```

2.1 (Q3)

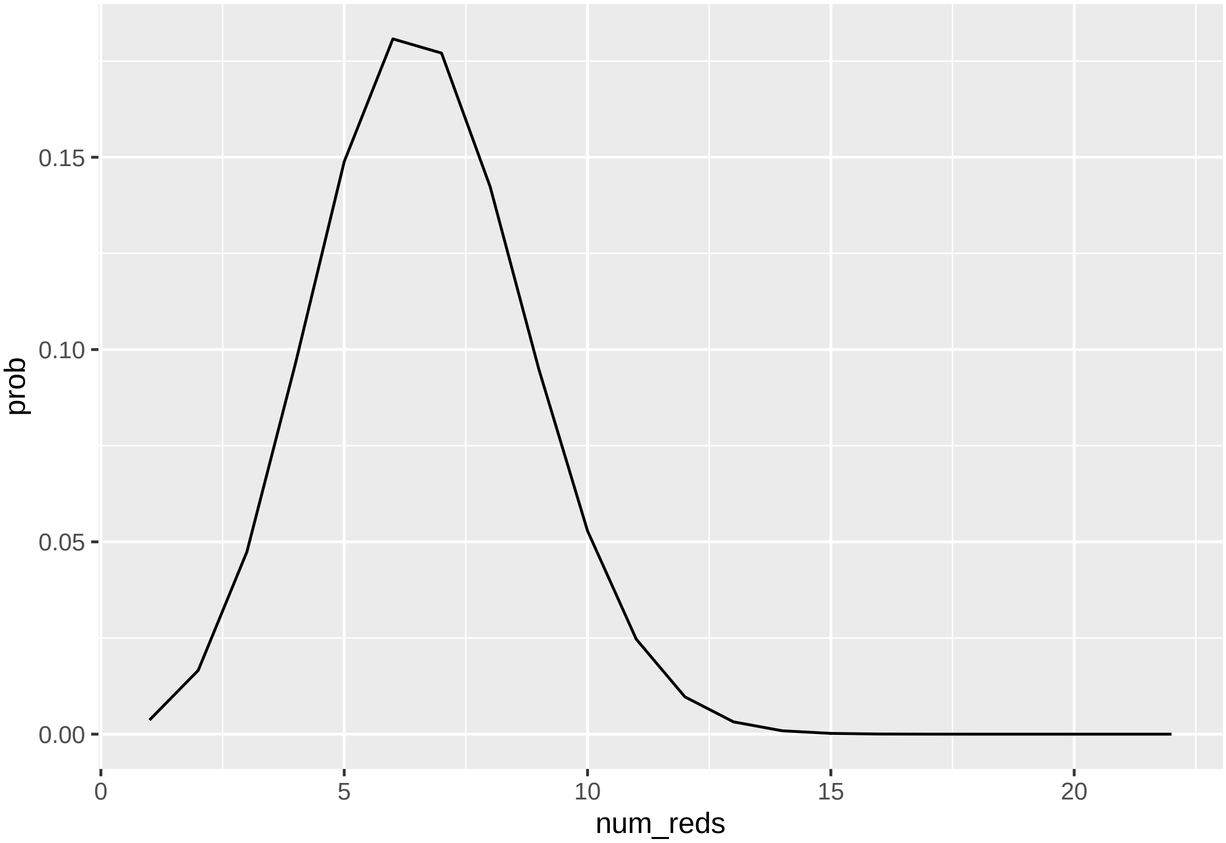
```
num_reds <- seq(1, 22, by = 1)
prob <- unlist(lapply(num_reds, prob_red_spheres))

prob_by_num_reds <- data.frame(num_reds, prob)
head(prob_by_num_reds)
```

```
##   num_reds        prob
## 1        1 0.003686403
## 2        2 0.016588812
## 3        3 0.047396606
## 4        4 0.096485948
## 5        5 0.148864035
## 6        6 0.180763470
```

2.1 (Q4)

```
prob_by_num_reds %>%
  ggplot(aes(num_reds, prob)) +
  geom_line()
```



2.1 (Q5)

```
sample(10, 22, replace = T)
```

```
##  [1] 7 3 8 4 7 9 5 4 1 7 8 2 1 3 8 2 6 3 3 1 1 5
```

```r
num_trials <- 1000
set.seed(0)
sampling_with_replacement_simulation <- data.frame(trial = 1:num_trials) %>%
  mutate(sample_balls = map(.x = trial, ~sample(10, 22, replace = T)))

head(sampling_with_replacement_simulation)
```

```
##   trial                                                         sample_balls
## 1     1   9, 4, 7, 1, 2, 7, 2, 3, 1, 5, 5, 10, 6, 10, 7, 9, 5, 5, 9, 9, 5, 5
## 2     2 2, 10, 9, 1, 4, 3, 6, 10, 10, 6, 4, 4, 10, 9, 7, 6, 9, 8, 9, 7, 8, 6
## 3     3   10, 7, 3, 10, 6, 8, 2, 2, 6, 6, 1, 3, 3, 8, 6, 7, 6, 8, 7, 1, 4, 8
## 4     4  9, 9, 7, 4, 7, 6, 1, 5, 6, 1, 9, 7, 7, 3, 6, 2, 10, 10, 7, 3, 2, 10
## 5     5 1, 10, 10, 8, 10, 5, 7, 8, 5, 6, 8, 1, 3, 10, 3, 1, 6, 6, 4, 9, 5, 1
## 6     6     3, 6, 3, 7, 3, 3, 1, 9, 2, 8, 6, 1, 2, 7, 7, 4, 9, 8, 3, 5, 3, 4
```

```r
sampling_with_replacement_simulation <-
  sampling_with_replacement_simulation %>%
  mutate(num_reds = (map_dbl(.x = sample_balls, ~sum(.x <= 3))))

head(sampling_with_replacement_simulation)
```

```
##   trial                                                         sample_balls
## 1     1   9, 4, 7, 1, 2, 7, 2, 3, 1, 5, 5, 10, 6, 10, 7, 9, 5, 5, 9, 9, 5, 5
## 2     2 2, 10, 9, 1, 4, 3, 6, 10, 10, 6, 4, 4, 10, 9, 7, 6, 9, 8, 9, 7, 8, 6
## 3     3   10, 7, 3, 10, 6, 8, 2, 2, 6, 6, 1, 3, 3, 8, 6, 7, 6, 8, 7, 1, 4, 8
## 4     4  9, 9, 7, 4, 7, 6, 1, 5, 6, 1, 9, 7, 7, 3, 6, 2, 10, 10, 7, 3, 2, 10
## 5     5 1, 10, 10, 8, 10, 5, 7, 8, 5, 6, 8, 1, 3, 10, 3, 1, 6, 6, 4, 9, 5, 1
## 6     6     3, 6, 3, 7, 3, 3, 1, 9, 2, 8, 6, 1, 2, 7, 7, 4, 9, 8, 3, 5, 3, 4
##   num_reds
## 1        5
## 2        3
## 3        7
## 4        6
## 5        6
## 6       10
```

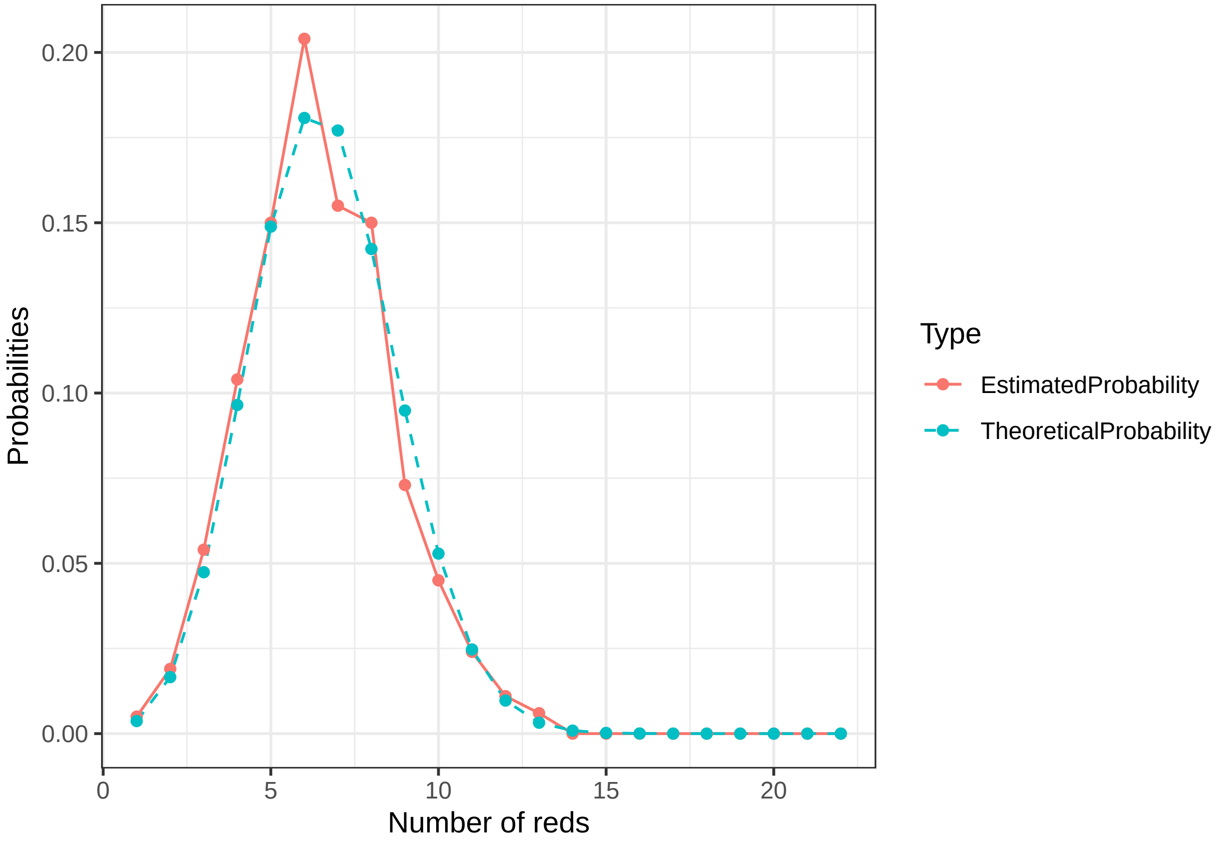
```r
num_reds_in_simulation <- sampling_with_replacement_simulation %>%
  pull(num_reds)

prob_by_num_reds <- prob_by_num_reds %>%
  mutate(predicted_prob = map_dbl(.x = num_reds, ~sum(num_reds_in_simulation == .x)) / num_trials)

head(prob_by_num_reds)
```

```
##   num_reds        prob predicted_prob
## 1        1 0.003686403          0.005
## 2        2 0.016588812          0.019
## 3        3 0.047396606          0.054
## 4        4 0.096485948          0.104
## 5        5 0.148864035          0.150
## 6        6 0.180763470          0.204
```

```r
prob_by_num_reds %>%
  rename(TheoreticalProbability = prob, EstimatedProbability = predicted_prob) %>%
  pivot_longer(cols = c("EstimatedProbability", "TheoreticalProbability"),
               names_to = "Type", values_to = "count") %>%
  ggplot(aes(num_reds, count)) +
  geom_line(aes(linetype = Type, color = Type)) +
  geom_point(aes(color = Type)) +
  scale_linetype_manual(values = c("solid", "dashed")) +
  theme_bw() +
  xlab("Number of reds") +
  ylab("Probabilities")
```



....

2.2 (Q1)

1

```
set.seed(19)
```

2

```
num_trials <- 1000
```

3

```
sampling_without_replacement_simulation <- data.frame(trial = 1:num_trials) %>%
  mutate(sample_balls = map(.x = trial, ~sample(100, 10, replace = F)))

head(sampling_without_replacement_simulation)
```

```
##   trial                          sample_balls
## 1     1  54, 3, 5, 82, 57, 67, 45, 15, 33, 22
## 2     2 43, 68, 84, 24, 48, 63, 5, 85, 11, 91
## 3     3  73, 11, 48, 98, 19, 20, 60, 1, 54, 8
## 4     4 80, 31, 91, 61, 6, 51, 13, 33, 93, 25
## 5     5 36, 65, 42, 73, 62, 14, 26, 39, 5, 63
## 6     6 84, 32, 83, 100, 44, 96, 4, 74, 98, 8
```

4

```
sampling_without_replacement_simulation <-
  sampling_without_replacement_simulation %>%
  mutate(num_reds = (map_dbl(.x = sample_balls, ~sum(.x <= 50)))) %>%
  mutate(num_blues = (map_dbl(.x = sample_balls, ~sum(.x > 50 & .x <= 80)))) %>%
  mutate(num_greens = (map_dbl(.x = sample_balls, ~sum(.x > 80))))

head(sampling_without_replacement_simulation)
```

```
##   trial                          sample_balls num_reds num_blues num_greens
## 1     1  54, 3, 5, 82, 57, 67, 45, 15, 33, 22        6         3          1
## 2     2 43, 68, 84, 24, 48, 63, 5, 85, 11, 91        5         2          3
## 3     3  73, 11, 48, 98, 19, 20, 60, 1, 54, 8        6         3          1
## 4     4 80, 31, 91, 61, 6, 51, 13, 33, 93, 25        5         3          2
## 5     5 36, 65, 42, 73, 62, 14, 26, 39, 5, 63        6         4          0
## 6     6 84, 32, 83, 100, 44, 96, 4, 74, 98, 8        4         1          5
```

5

```
sampling_without_replacement_simulation <-
  sampling_without_replacement_simulation %>%
  mutate(min_count = pmin(num_reds, num_blues, num_greens))
head(sampling_without_replacement_simulation)
```

```
##   trial                          sample_balls num_reds num_blues num_greens
## 1     1  54, 3, 5, 82, 57, 67, 45, 15, 33, 22        6         3          1
## 2     2 43, 68, 84, 24, 48, 63, 5, 85, 11, 91        5         2          3
## 3     3  73, 11, 48, 98, 19, 20, 60, 1, 54, 8        6         3          1
## 4     4 80, 31, 91, 61, 6, 51, 13, 33, 93, 25        5         3          2
## 5     5 36, 65, 42, 73, 62, 14, 26, 39, 5, 63        6         4          0
## 6     6 84, 32, 83, 100, 44, 96, 4, 74, 98, 8        4         1          5
##   min_count
## 1         1
## 2         2
## 3         1
## 4         2
## 5         0
## 6         1
```

6

```
# Proportion of rows where minimum number of three counts is 0.
sum(sampling_without_replacement_simulation$min_count %in% 0) / nrow(sampling_without_replacement_simula
```

```
## [1] 0.118
```