Assignment 4

Yujie Wang

2022-10-19

```
library(tidyverse)
```

```
## -- 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 Rules of probability

1.1 (Q1) Construct probability based on the Rules of probability

Question:

Consider a sample space $\Omega = \{a, b, c\}$ and a set of events $\xi = \{A \subseteq B\}$. Based on the rules of probability, find an example of probability $P : \xi \rightarrow [0, 1]$ such that P satisfies:

$P(\{a, b\} = 0.6)\, and\, P(\{b, c\} = 0.5)$

Make sure that your function P satisfies the three rules

Answer:

Based on the rules:

$P(\{a, b\}) + P(\{c\}) = 1$
$P(\{b, c\}) + P(\{a\}) = 1$
$So,\ P(\{c\}) = 0.4, P(\{a\}) = 0.5, P(\{b\}) = 0.1$

Example:In a box,there are 10 spheres,and 5 of them are labeled "a",1 of them is labeled "b", and 4 of them are labeled "c".Every time only one sphere is drawn from the box,after that,the sphere will be returned to the bag.

1.1 (Q2) Verify that the following probability space satisfies the rules of probability.

Question:

Consider a setting in which the sample space $\Omega = \{0,1\}$, and $\xi = \{A \subseteq \Omega\} = \{\emptyset, \{0\}, \{1\}, \{0,1\}\}$. For a fixed $q \in [0,1]$, define a function $P : \xi \to [0,1]$ by

$P(\emptyset) = 0,\ P(\{0\}) = 1 - q,\ P(\{1\}) = q,\ P(\{0,1\}) = 1.$

Show that the probability space (Ω, ξ, P) satisfies the three rules of probability

Answer:

$From\, the\, description, we\, know :$
$for\, any\, A \in \xi$
$P(\emptyset) = 0 \geq 0, P(\{0\}) = 1 - q, and\, q \to [0,1],$
$So\, P(\{0\}) \geq 0, and\, P(\{1\}) = q \geq 0, P(\{0,1\}) = 1 >= 0$
$Also, P(\Omega) = P(\{0,1\}) = 1,$
$P(\Omega) = P(\{0\} \cup \{1\}) = P(\{0\}) + P(\{1\}) = 1 - q + q = 1$

So,the probability space(Ω, ξ, P) satisfies the three rules of probability.

1.2 Deriving new properties from the rules of probability

1.2 (Q1) Union of a finite sequence of disjoint events.

Question:

In Rule 3,we have:

$P(\cup_{i=1}^{\infty} A_i) = \sum_{i=1}^{\infty} P(A_i)$

for an infinite sequence of pairwise disjoint events $A_1, A_2, ...$ Show that for a finite sequence of disjoint events $A_1, A_2, ..., A_n$, for any integer n bigger than 1, the above equality holds as a consequence of Rule 3,i.e.,

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

Answer:

In an infinite sequence,$\Omega = \{A_1, A_2, ...\}$ and $P(\cup_{i=1}^{\infty} A_i) = \sum_{i=1}^{\infty} P(A_i)$

let $\xi = A_1, A_2, ..., A_n$

so $\xi \subset \Omega$,

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

1.2 (Q2) Probability of a complement.

Question:

Prove that if Ω is a sample space, $S \subseteq \Omega$ is an event and $S^c := \Omega \setminus S$ is its complement, then we have

$P(S^c) = 1 - P(S)$

Proof:

$S\,and\,S^c are\,disjoint$
$So\,P(S \cup S^c) = P(S) + P(S^c)$
$And\,P(S \cup S^c) = P(\Omega) = 1$
$So\,P(S) + P(S^c) = 1$
$P(S^c) = 1 - P(S)$

1.2 (Q3) The union bound

Question:

In Rule 3, for pairwise disjoint events $A_1, A_2, ...$, we have

$P(\cup_{i=1}^{\infty} A_i) = \sum_{i=1}^{\infty} P(A_i)$

Recall that we have also shown the union bound as a consequence of the rules of probability: for a sequence of events $S_1, S_2, ...$, we have $P(\cup_{i=1}^{\infty} S_i) \leq \sum_{i=1}^{\infty} P(S_i)$.

Given an example of a sequence of sets $S_1, S_2, ...$, such that $P(\cup_{i=1}^{\infty} S_i) \neq \sum_{i=1}^{\infty} P(S_i)$.

Answer:

In 1.1 (Q1),we know that

$P(\{a, b\}) = 0.6, P(\{b, c\}) = 0.5$
$and\,P(\{a\}) = 0.5, P(\{b\}) = 0.1, P(\{c\}) = 0.4$
$Let\,S_1 = \{a, b\}, S_2 = \{b, c\}$
$So\,P(S_1 \cup S_2) = P(\{a, b, c\}) = 1 \neq P(S_1) + P(S_2) = 0.6 + 0.5 = 1.1$

1.2 (Q4) Probability of union and intersection of events.

Question:

Show that for events $A \subseteq \Omega$ and $B \subseteq \Omega$, we have

$P(A \cup B) = P(A) + P(B) - P(A \cap B)$

Answer:

$Let\,C = A \cap B^c$
$D = A \cap B$
$E = A^c \cap B$

So
$P(A \cup B) = P(C) + P(D) + P(E)$
$A = C \cup D\,and\,C\,and\,D\,are\,disjoint$
$So\,P(A) = P(C) + P(D).$
$P(B) = P(D) + P(E).$
$So\,P(A) + P(B) = P(C) + P(E) + 2P(D)$
$= P(A \cup B) + P(D)$
$So\,P(A \cup B) = P(A) + P(B) - P(D)$
$= P(A) + P(B) - P(A \cup B)$

2. Finite probability spaces

2.1 Sample with replacement

2.1 (Q1)

Question:

Write down a mathematical expression for the probability that z out of the 22 selections were red spheres (here $z \in \{0, 1, ..., 22\}$).

Answer:

Let A is the event that z out of the 22 selections were red spheres:

$$P(A) = \frac{22!}{z!(22-z)!} \cdot (\frac{3}{10})^z \cdot (\frac{7}{10})^{22-z}$$

2.1 (Q2)

Next write an R function called prob_red_spheres() which takes z as an argument and computes the probability that z out of a total of the 22 balls selected are red.

Test:

```
# the results of my expression
prob_red_spheres<-function(z){
  results<-choose(22,z)*(0.3)^z*(0.7)^(22-z)
  return(results)
}
prob_red_spheres(10)
```

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

2.1 (Q3)

Generate a data frame called prob_by_num_reds with two columns num_reds and prob. The num_reds column should contain numbers 1 through 22 and the prob column should give the associated probability of selecting that many reds out of a total number of 22 selections.

Answer:

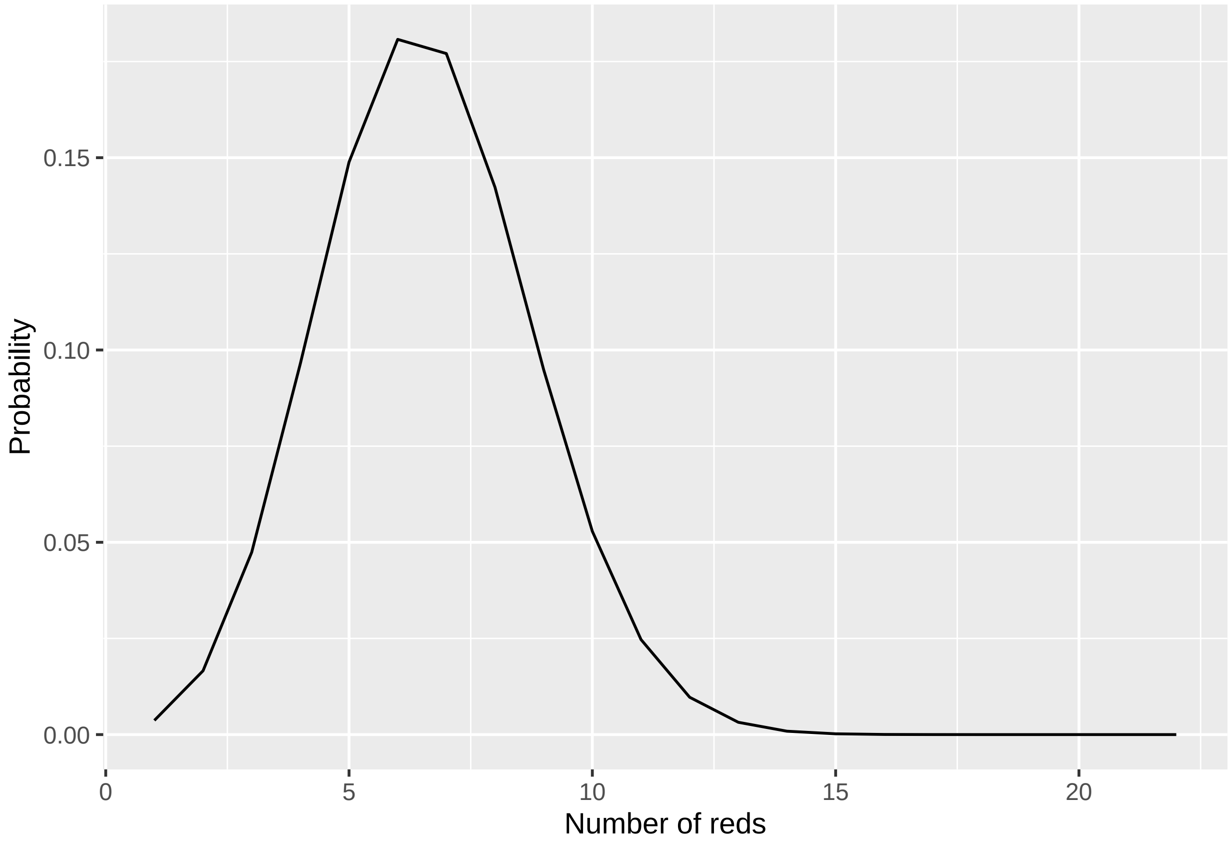
```
library(dplyr)
num_reds<-seq(22)
prob<-map_dbl(num_reds,prob_red_spheres)
prob_by_num_reds<-data.frame(num_reds,prob)
prob_by_num_reds%>%head(3)
```

```
##   num_reds        prob
## 1        1 0.003686403
## 2        2 0.016588812
## 3        3 0.047396606
```

2.1 (Q4)

Now use the geom_line() function within the ggplot2 library, in conjunction with your data frame to display a plot of the probability as a function of the number of reds.

```
library(ggplot2)
ggplot(data=prob_by_num_reds,aes(x=num_reds,y=prob))+geom_line()+xlab('Number of reds')+ylab('Probabili
```



2.1 (Q5)

Next we shall explore the sample() function within R. Let's suppose we want to simulate a random experiment in which we sample with replacement from a collection of 10 objects, and repeat this process 22 times.

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

```
##  [1]  1  8  9  6 10  7  9  9  3 10  9 10 10  6  8  1  7  7  1  4  6  6
```

In the process,by setting the function *set.seed()* we can get the same output every time.

```
# Setting the random seed just once
set.seed(0)
for(i in 1:5){
  print(sample(100,5,replace=FALSE))
}
```

```
## [1] 14 68 39  1 34
## [1] 87 43 14 82 59
## [1] 51 97 85 21 54
## [1] 74  7 73 79 85
## [1]  37  89 100  34  99
```

```
# Resetting the random seed every time
for(i in 1:5){
  set.seed(1)
  print(sample(100,5,replace=FALSE))
}
```

```
## [1] 68 39  1 34 87
## [1] 68 39  1 34 87
## [1] 68 39  1 34 87
## [1] 68 39  1 34 87
## [1] 68 39  1 34 87
```

We shall now use the *sample()* to construct a simulation study to explore the probability of selecting z red balls from a bag of size 10, with 3 red and 7 blue balls, when sampling 22 balls with replacement.

1. set a random seed.

2. create a data frame called *sampling_with_replacement_simulation* consisting of two columns. The first is called *trial* and contains numbers 1 through 1000. The second is called *sample_balls* and corresponds to random samples of size 22 from a bag of size 10, with replacement.

3. Now add a new column called *num_reds* such that, for each row, *num_reds* contains an integer which gives the number of items within the sample for that row (the entry in the sample_balls column) which are less than or equal to three (assuming that the three red balls are labelled by {1, 2, 3}).

For example, suppose that in some row of the data frame, the sample_balls column contains the following list:

4 10 4 10 8 3 5 5 5 5 10 7 1 2 1 10 5 6 5 7 1 10

Then the corresponding row of the num_reds column should contain the number 5, since 5 of these values are less than equal to 3.

```
num_trials<-1000
set.seed(0)
sampling_with_replacement_simulation<-data.frame(trials=1:num_trials)%>%mutate(sample_balls=map(.x=trial
compute_num<-function(x,y,z){
  new<-as.vector(unlist(x))
  return (sum(new<=z & new>=y))
}
sampling_with_replacement_simulation<-sampling_with_replacement_simulation%>%mutate(num_reds=map_dbl(.x=
sampling_with_replacement_simulation%>%head(10)
```

```
##    trials                                                         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
## 7       7 2, 1, 7, 9, 10, 10, 2, 2, 3, 1, 2, 3, 3, 3, 8, 9, 2, 10, 8, 10, 3, 5
## 8       8     9, 5, 7, 5, 6, 4, 2, 1, 3, 8, 9, 6, 1, 4, 5, 9, 5, 8, 4, 1, 9, 5
## 9       9  1, 5, 4, 10, 10, 9, 8, 5, 5, 6, 6, 2, 2, 8, 4, 10, 8, 5, 5, 8, 8, 7
## 10     10    4, 4, 1, 10, 4, 9, 9, 9, 9, 6, 6, 4, 3, 3, 9, 9, 7, 9, 5, 7, 4, 4
##    num_reds
## 1         5
## 2         3
## 3         7
## 4         6
## 5         6
## 6        10
## 7        12
## 8         5
## 9         3
## 10        3
```

2.1 (Q6)

Next we shall add a new column called *predicted_prob* to our existing data frame *prob_by_num_reds* which gives the number of times (that we observed the corresponding number of reds within our simulation) divided by the total number of observations. This aims to estimate the probability of the event that z out of a total of the 22 balls selected are red (for z = 1, 2, . . . , 22).

```
num_reds_in_simulation<-sampling_with_replacement_simulation%>%
  pull(num_reds)
#we extract a vector corresponding to the number of reds in each trial
prob_by_num_reds<-prob_by_num_reds%>%mutate(predicted_prob=map_dbl(.x=num_reds,~sum(num_reds_in_simulati
#add a column which gives the number of trials with a given number of reds
prob_by_num_reds%>%head(10)
```

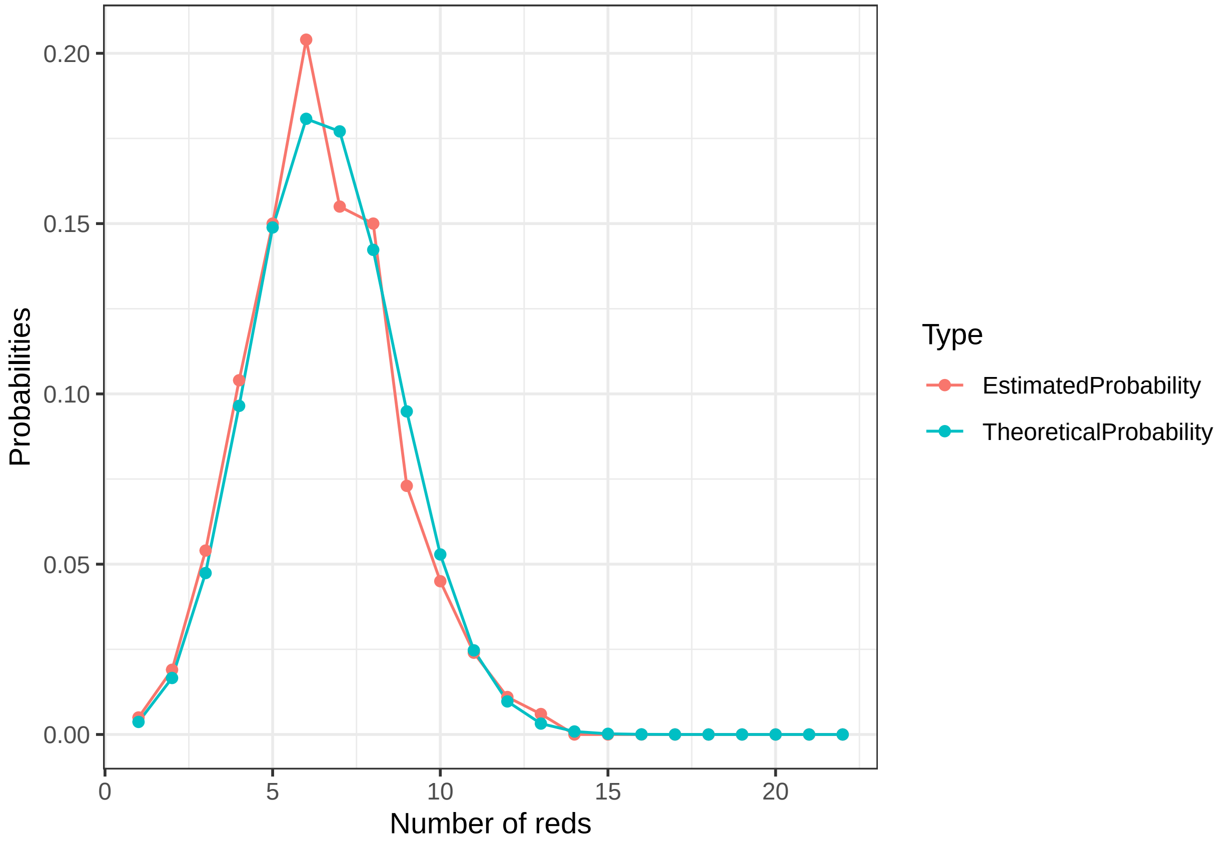
```
##    num_reds        prob predicted_prob
## 1         1 0.003686403          0.005
## 2         2 0.016588812          0.019
## 3         3 0.047396606          0.054
## 4         4 0.096485948          0.104
## 5         5 0.148864035          0.150
## 6         6 0.180763470          0.204
## 7         7 0.177074420          0.155
## 8         8 0.142291945          0.150
## 9         9 0.094861296          0.073
## 10       10 0.052851294          0.045
```

2.1 (Q7)

Finally, create a plot which compares the results of your simulation with your probability formula.

```
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")
```

```
## Warning: Ignoring unknown aesthetics: lineType
```



2.2 Sampling without replacement

Let’s suppose we have a large bag containing 100 spheres. There are 50 red spheres, 30 blue spheres and 20 green spheres. Suppose that we sample 10 spheres from the bag without replacement.

2.2 (Q1)

What is the probability that one or more colours are missing from your selection? First aim to answer this question via a simulation study using ideas from the previous question.

1. First set a random seed
2. Next set a number of trials (e.g., 10 or 1000. Try this initially with a small number of simulations.Increase your number of simulations to about a relatively large number to get a more accurate answer, once everything seems to be working well), and a sample size (10);

```
set.seed(0)

sample_size<-10
```

3. Now use a combination of the functions sample(), mutate() and map() to generate your samples. Here you are creating a sample of size 10 from a collection of 100 balls—the sampling is done without replacement;

```
number_trials<-1000
sampling_without_replacement_simulation<-data.frame(trials=1:number_trials)%>%
  mutate(samples=map(.x=trials,~sample(100,10,replace=FALSE)))
sampling_without_replacement_simulation%>%head(10)
```

```
##    trials                                  samples
## 1       1  14, 68, 39, 1, 34, 87, 43, 100, 82, 59
## 2       2   51, 97, 85, 21, 54, 74, 7, 73, 79, 98
## 3       3 37, 89, 100, 34, 99, 44, 79, 33, 84, 35
## 4       4 70, 74, 42, 38, 20, 28, 96, 44, 87, 100
## 5       5   40, 44, 25, 70, 39, 51, 42, 6, 24, 32
## 6       6   14, 2, 45, 18, 22, 78, 65, 70, 87, 93
## 7       7  75, 81, 13, 40, 89, 48, 96, 23, 84, 29
## 8       8  13, 22, 93, 28, 48, 33, 45, 21, 31, 17
## 9       9  73, 87, 83, 90, 48, 64, 94, 60, 51, 34
## 10     10   10, 1, 43, 59, 26, 15, 58, 29, 24, 42
```

4. Now compute the number of "reds", "greens" and "blues" in your sample using the map_dbl() and mutate() functions.

```
sampling_without_replacement_simulation<-sampling_without_replacement_simulation%>%
  mutate(reds=map_dbl(.x=samples,~compute_num(.x,1,50)),
         blues=map_dbl(.x=samples,~compute_num(.x,51,80)),
         greens=map_dbl(.x=samples,~compute_num(.x,81,100)))
sampling_without_replacement_simulation%>%head(10)
```

```
##    trials                                  samples reds blues greens
## 1       1  14, 68, 39, 1, 34, 87, 43, 100, 82, 59    5     2      3
## 2       2   51, 97, 85, 21, 54, 74, 7, 73, 79, 98    2     5      3
## 3       3 37, 89, 100, 34, 99, 44, 79, 33, 84, 35    5     1      4
## 4       4 70, 74, 42, 38, 20, 28, 96, 44, 87, 100    5     2      3
## 5       5   40, 44, 25, 70, 39, 51, 42, 6, 24, 32    8     2      0
## 6       6   14, 2, 45, 18, 22, 78, 65, 70, 87, 93    5     3      2
## 7       7  75, 81, 13, 40, 89, 48, 96, 23, 84, 29    5     1      4
## 8       8  13, 22, 93, 28, 48, 33, 45, 21, 31, 17    9     0      1
## 9       9  73, 87, 83, 90, 48, 64, 94, 60, 51, 34    2     4      4
## 10     10   10, 1, 43, 59, 26, 15, 58, 29, 24, 42    8     2      0
```

5. Compute the minimum of the three counts using the pmin() function. When this minimum is zero, then one of the three colours is missing. It is recommended that you look up the difference between pmin() and min() here;

```r
sample_head<-sampling_without_replacement_simulation%>%head(10)
pmin(sample_head$reds,sample_head$blues,sample_head$greens)
```

```
##  [1] 2 2 1 2 0 2 1 0 2 0
```

```r
min(sample_head$reds,sample_head$blues,sample_head$greens)
```

```
## [1] 0
```

```r
min_color<-pmin(sampling_without_replacement_simulation$reds,
                sampling_without_replacement_simulation$blues,
                sampling_without_replacement_simulation$greens)
```

6. Compute the proportion of rows for which the minimum number of the three counts is zero.

```r
## Estimated Probability
prob_zero<-sum(min_color==0)/number_trials
prob_zero
```

```
## [1] 0.124
```

2.2 (Q2)

Description of Question

Let's derive a mathematical expression for the probability that we considered in (Q1): You can try and use "combinations" with $\binom{n}{k}$ to compute the probability directly.

1. First aim to compute the number of subsets of size 10 from 100 which either entirely miss out one of the subsets $Red = \{1, ..., 50\}, Blues = \{51, ..., 80\}, Greens = \{81, ..., 100\}$.
2. Then compute the number of subsets of size 10 from 100 which miss out two of the subsets Red, Blues, Green.
3. Be careful not to double count some of these subsets!
4. Once you have computed all such subsets, combine them with the formula for the total number of subsets of size 10 from a set of 100,to compute the probability of missing a colour.

Once you have the mathematical expression for the probability, you can check how close the probability computed in (Q1) to the theoretical values.

Answers:

Let A is the event that one or more colours are missing from the selection.

$$P(A) = \frac{\sum_{i=1}^{10}((\binom{Blues}{i} \cdot \binom{Greens}{10-i}) + \binom{Greens}{i} \cdot \binom{Reds}{10-i}) + \binom{Reds}{i} \cdot \binom{Blues}{10-i}))}{\binom{100}{10}}$$

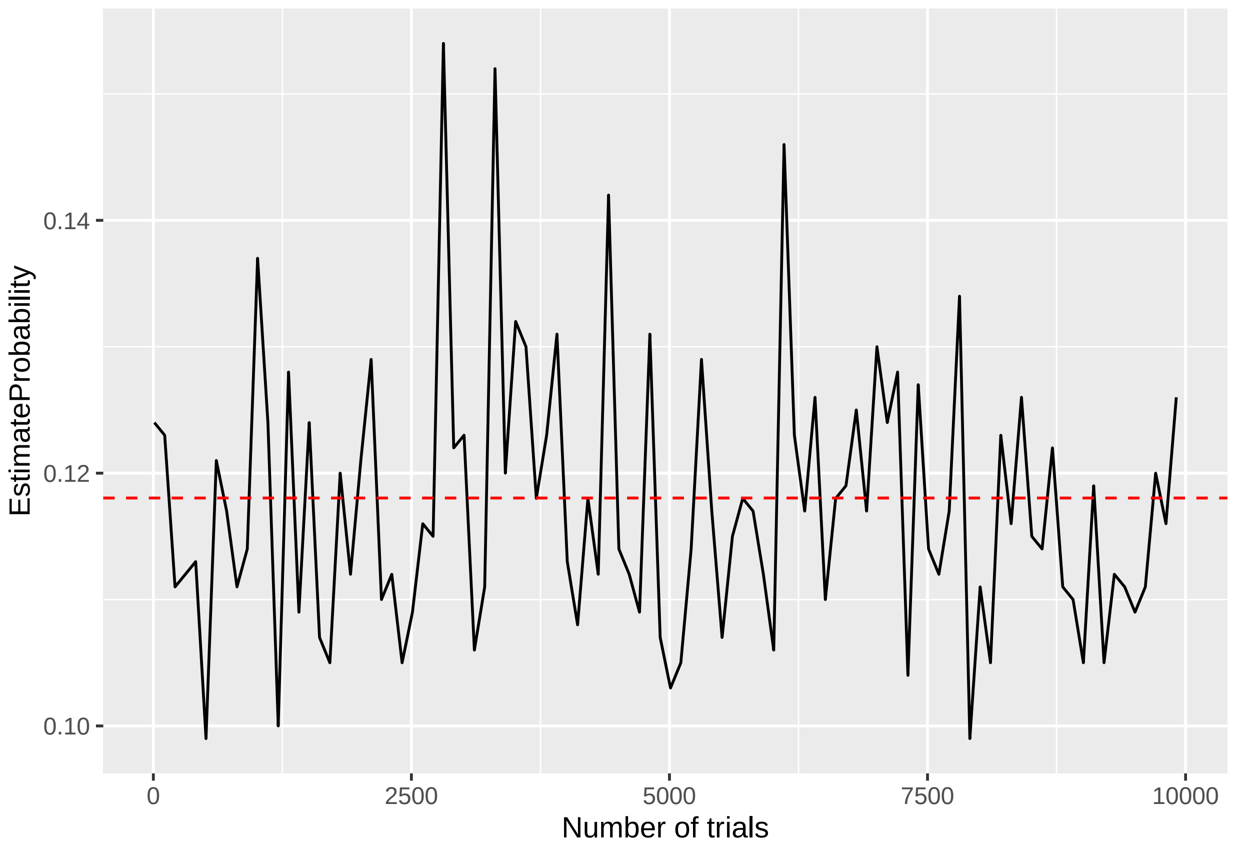
Test:

```r
num_sum<-function(){
  num<-0
  for (i in 1:10){
    num=num+(choose(30,i)*choose(20,10-i)+choose(20,i)*choose(50,10-i)+choose(50,i)*choose(30,10-i))
  }
  return(num)
}
#Theoretical Probability
PA<-num_sum()/choose(100,10)
PA
```

```
## [1] 0.1180318
```

```r
## Simulative Probability
set.seed(0)
PS<-function(num_trials){
  sample_size<-10
  sampling_without_replacement_simulation<-data.frame(trials=1:number_trials)%>%
    mutate(samples=map(.x=trials,~sample(100,10,replace=FALSE)))
  sampling_without_replacement_simulation<-sampling_without_replacement_simulation%>%
    mutate(reds=map_dbl(.x=samples,~compute_num(.x,1,50)),
           blues=map_dbl(.x=samples,~compute_num(.x,51,80)),
           greens=map_dbl(.x=samples,~compute_num(.x,81,100)))
  min_color<-pmin(sampling_without_replacement_simulation$reds,
                  sampling_without_replacement_simulation$blues,
                  sampling_without_replacement_simulation$greens)
  prob_zero<-sum(min_color==0)/number_trials
  return(prob_zero)
}

PSline<-map_dbl(seq(10,10000,by=100),PS)
num_trial<-seq(10,10000,by=100)
Table<-data.frame(num_trial,PSline)
ggplot(data=Table,aes(x=num_trial,y=PSline))+
  geom_line()+
  xlab("Number of trials")+
  ylab("EstimateProbability")+
  geom_hline(aes(yintercept=PA),color="red",linetype="dashed")
```


0.14
0.12
0.10
EstimateProbability
0
2500
5000
7500
10000
Number of trials