Lab4.R

daniellesenechal

2019-10-06

########################################################  
# Name: Danielle Senechal  
# CSC-315  
# Lab #4: Probability   
########################################################  
   
library(gtools)  
library(ggplot2)  
library(dplyr)

##   
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':  
##   
## filter, lag

## The following objects are masked from 'package:base':  
##   
## intersect, setdiff, setequal, union

##########################################################################  
# Add R code to the script below and create a Notebook to complete  
# the steps and explicitly answer the following questions  
##########################################################################  
  
# Basic Probability Questions -- Use R as a calculator and specify the  
# answers below  
  
# A standard deck of cards contains 52 cards, with 13 cards from each   
# suit (hearts, clubs, diamonds, and spades).  
   
#1. If one card is selected at random, what is the probability that   
  
# (a) the card is the ace of spades?  
 1/52

## [1] 0.01923077

# (b) the card is NOT the ace of spades?  
 51/52

## [1] 0.9807692

# (c) the card is an ace (of any suit)?  
 4/52

## [1] 0.07692308

# (d) the card is an ace OR a 4?  
 8/52

## [1] 0.1538462

# Use R to answer the remaining questions. You MUST use R to   
# enumerate and analyze the sample space or to carry out   
# probability experiments (simulations) in order to calculate  
# an empirical probability.  
  
  
#2. This question looks at the probability of rolling two dice  
# (each with values 1 - 6) and getting a sum of 7.  
# You will answer this question in parts.  
  
# (a) Use the 'permutations' function from 'gtools'  
# to enumerate the sample space obtained by rolling two dice.  
# (Note: the correct sample space has 36 outcomes)  
   
 S <- permutations(6 ,2, c(1:6), repeats = TRUE)  
 head(S)

## [,1] [,2]  
## [1,] 1 1  
## [2,] 1 2  
## [3,] 1 3  
## [4,] 1 4  
## [5,] 1 5  
## [6,] 1 6

nrow(S)

## [1] 36

# (b) Use R and your answer to (a) to find the number of outcomes   
# where the sum is 7  
   
 sumS <- rowSums(S) == 7  
 seven <- S[sumS,]   
 seven

## [,1] [,2]  
## [1,] 1 6  
## [2,] 2 5  
## [3,] 3 4  
## [4,] 4 3  
## [5,] 5 2  
## [6,] 6 1

nrow(seven)

## [1] 6

# (c) divide your answer from (b) by the size of the sample space  
# to find the probability that the sum is 7.  
   
 nrow(seven)/nrow(S)

## [1] 0.1666667

#3. Calculate the same probability in (2) but by finding the empirical   
# probability by completing the steps below.  
  
# (a) Write a function that rolls two dice and returns the sum of the   
# die rolls (this is done for you)  
  
 roll2 <- function() {  
 s <- sample(1:6, 2, replace = TRUE)  
 sum(s)  
 }  
  
# (b) Use the 'replicate' function to roll two dice 5000 times, to  
# get a vector containing the sum of die rolls for each experiment.  
   
 roll2.5000 <- replicate(5000, roll2())   
 head(roll2.5000)

## [1] 8 8 6 9 7 5

# (c) Find the number of times you rolled a seven, and divide by the   
# number of experiments to find the empirical probability  
  
 num.sevens <- sum(roll2.5000==7)  
 num.sevens

## [1] 842

num.sevens/length(roll2.5000)

## [1] 0.1684

# Definition: A probability distribution of a discrete random variable   
# gives the probability for each value that the variable can take. For   
# example, if we flip a coin three times, and let X = the number of heads,  
# then the probability distribution of X is given by the following code:  
  
pdist <- cbind(X = 0:3, 'P(X)' = c(0.125,0.375, 0.375, 0.125))  
pdist

## X P(X)  
## [1,] 0 0.125  
## [2,] 1 0.375  
## [3,] 2 0.375  
## [4,] 3 0.125

# In other words, P(X = 0) is 0.125, which says that the probability of   
# getting no heads (or all tails) is 0.125. You will derive the above  
# probability distribution in the next problem.  
  
  
#4. We will look at flipping a coin 3 times and letting X = the  
# number of heads. Find the probability distribution of X by  
# completing the steps below:  
  
# (a) Use the 'permutations' function to enumerate the sample  
# space obtained from flipping a coin 3 times.  
  
 coin <- permutations(2,3, c("H", "T"), repeats = TRUE)  
 coin

## [,1] [,2] [,3]  
## [1,] "H" "H" "H"   
## [2,] "H" "H" "T"   
## [3,] "H" "T" "H"   
## [4,] "H" "T" "T"   
## [5,] "T" "H" "H"   
## [6,] "T" "H" "T"   
## [7,] "T" "T" "H"   
## [8,] "T" "T" "T"

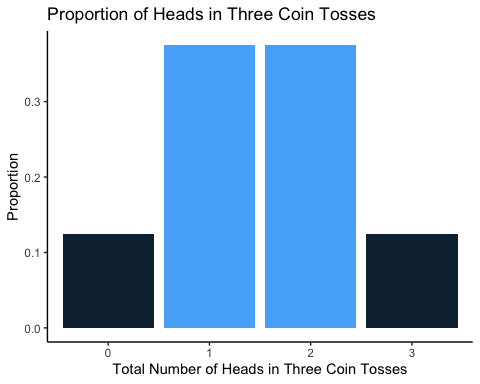
# (b) Using your sample space, find X = the number of heads for each  
# set of 3 coins.   
  
 num.heads <- rowSums(coin == "H")  
 num.heads

## [1] 3 2 2 1 2 1 1 0

# (c) Create a relative frequency table for X, which is the probability  
# distribution of X = the number of heads in 3 coin tosses.  
   
 num.heads <- table(num.heads) %>% prop.table() %>% data.frame()  
 num.heads

## num.heads Freq  
## 1 0 0.125  
## 2 1 0.375  
## 3 2 0.375  
## 4 3 0.125

# (d) Create a bar graph of the relative frequencies, using ggplot and  
# labeling the x-axis, y-axis, and title.   
   
 num.heads <- data.frame(num.heads)  
 ggplot(num.heads) + geom\_col(aes(num.heads, Freq, fill = Freq)) +  
 labs(x = "Total Number of Heads in Three Coin Tosses", y = "Proportion",   
 title = "Proportion of Heads in Three Coin Tosses") +  
 theme\_classic() + theme(legend.position = "none")



#5. Find the empirical distribution of X = the number of heads in 3  
# coin tosses by completing the steps below.   
  
# (a) Write a function that flips a coin 3 times and returns the number  
# of heads  
  
 flip.three <- function() {  
 f <- sample(c("H", "T"), 3, replace = TRUE)  
 count <- sum(f=="H")  
 }  
   
# (b) Use the 'replicate' function to repeat 3 coin tosses 5000 times, to  
# get a vector containing the number of heads for each experiment  
  
 res <- replicate(5000, flip.three())   
 head(res)

## [1] 2 2 1 2 2 1

# (c) Create a relative frequency table for the number of heads. This is  
# the empirical probability distribution of X = the number of heads  
# in 3 coin tosses.  
   
 num.heads2 <- table(res) %>% prop.table() %>% data.frame()   
 num.heads2

## res Freq  
## 1 0 0.1252  
## 2 1 0.3788  
## 3 2 0.3696  
## 4 3 0.1264

######################################################################  
# Poker Time! The commands below enumerate the sample space of  
# all possible poker hands. Here we ignore the suit because is not   
# needed for the questions below. We also use combinations   
# instead of permutations. Combinations should be used when the  
# order does not matter (e.g., which is true for the order of cards  
# in a hand). The cards are sampled WITHOUT replacement   
# (repeats.allowed = FALSE) because we cannot include  
# the same card twice in one hand. Finally, we specify 'set = FALSE' to   
# allow for duplicate values in the deck vector. Each combination   
# (hand) is equally likely, so classical probability can be used.  
######################################################################  
  
deck <- rep(1:13,4)  
hands <- combinations(52, 5, deck, repeats.allowed = FALSE, set = FALSE)  
  
#6. How many possible poker hands are there?  
  
nrow(hands)

## [1] 2598960

#There are 2598960 possible poker hands  
  
#7. The function below takes a vector (corresponding to a hand of cards)  
# and returns TRUE if the hand contains a four-of-a-kind  
# Use 'apply' to apply this function to each hand, in order to show  
# that the probability of being dealt a four-of-a-kind is  
# approximately 0.00024 (or 1/4165). Note: You MUST use the  
# apply function and the four.of.a.kind function below to find this.   
# Because the hands matrix contains more than 2.5 million rows,   
# this calculation may take several minutes. You should therefore test   
# your code on a subset of the hands matrix first. There are two 4-of-a-kinds  
# if you look at the first 20,000 rows.  
  
##############################################################  
# this function returns true if a hand contains a 4-of-a-kind  
##############################################################  
four.of.a.kind <- function(x) {  
 t <- table(x) # frequency table for cards in the hand  
 m <- max(t) # how frequent is the most common card?  
 if (m == 4) return (TRUE)  
 return (FALSE)  
}  
mf <- head(hands, n = 20000)  
fours <- apply(mf, 1, four.of.a.kind)  
head(fours) #first 20000

## [1] FALSE FALSE FALSE FALSE FALSE FALSE

fours <- apply(hands, 1, four.of.a.kind)  
head(fours) #all

## [1] FALSE FALSE FALSE FALSE FALSE FALSE

count.four <- sum(fours == T)  
count.four

## [1] 624

count.four/length(hands)

## [1] 4.801921e-05

#8. Create a function that determines whether a vector 'x' contains   
# a full house (i.e., 3 of a kind and 1 pair). You can assume  
# that 'x' includes exactly 5 cards.  
full.house <- function(x) {  
 t <- table(x)   
 m <- max(t)   
 d <- min(t)  
 if (m == 3 && d == 2) return (TRUE)  
 return (FALSE)  
}  
  
#9. Show that the probability of being dealt a full house is   
# approximately 0.00144 (roughly 1/694). Note for testing purposes,  
# that there are 18 full houses in the first 20,000 rows of the  
# hands matrix  
  
mf <- head(hands, n = 20000)  
fh <- apply(mf, 1, full.house)  
head(fh) #first 20000

## [1] FALSE FALSE FALSE FALSE FALSE FALSE

fh <- apply(hands, 1, full.house)  
head(fh) #all

## [1] FALSE FALSE FALSE FALSE FALSE FALSE

count.fh <- sum(fh == T)  
count.fh

## [1] 3744

count.fh/length(hands)

## [1] 0.0002881152