CS544_HW2_Escandon

Professor S. Kalathur Spring 01 2021

Phillip Escandon - escandon@bu.edu

01 February, 2021

Part 1 Probability

Use the Bayes theorem to calculate the following probabilities. Show the individual steps of the Bayes theorem. You can use R for the calculations.

Suppose that in a particular state, among 10000 people surveyed, 4250 people are in the age group 18-34 years, 2850 people are in the age group 35-49 years, 1640 people are in the age group 50-64 years, and the remaining are 65 years & over.

Out of those in the age group 18-34 years, 1062 people had a BMI of above 30. Of those in the age group 35-49 years, 1710 people had a BMI of above 30. Among those in the 50-64 years range, 656 people had a BMI of above 30. In the last age group, 189 people had a BMI of above 30.

```r
# Get my population figures
totalPop <- 10000
ageGroupPop <- c(4250,2850,1640,1260)
byPercent <- ageGroupPop/totalPop

# Get my BMI figures

bmiGT30 <- c(1062,1710,656,189)
bmiGT30Percent <- bmiGT30 / ageGroupPop

df <- data.frame(byPercent,bmiGT30Percent)
df
```

```
##   byPercent bmiGT30Percent
## 1     0.425      0.2498824
## 2     0.285      0.6000000
## 3     0.164      0.4000000
## 4     0.126      0.1500000
```

a. What is the probability that a randomly selected person in this survey will have a BMI above 30?

```r
# P(B) = P(B|A1)*P(A1)   +  P(B|A2)*P(A2)  + P(B|A3)*P(A3)
pb <- sum(byPercent*bmiGT30Percent)
glue("Probability of randomly chosen person having a BMI > 30 is ",pb)
```

```
## Probability of randomly chosen person having a BMI > 30 is 0.3617
```

b. If a randomly selected person had a BMI of above 30, what is the probability of that person being in the age group 18-34 years?

```r
# P(A1|B) = P(B|A1)* P(A1)/P(B)

PA1B=(byPercent * bmiGT30Percent)[1] / pb
glue("Probability of randomly chosen person having a BMI>30 being in the 18-34
     age group is ", round(PA1B,digits=4))
```

```
## Probability of randomly chosen person having a BMI>30 being in the 18-34
## age group is 0.2936
```

c. If a randomly selected person had a BMI of above 30, what is the probability of that person being in the age group 35-49 years?

```
PA2B=(byPercent * bmiGT30Percent)[2] / pb
glue("Probability of randomly chosen person having a BMI>30 being in the 35-49
     age group is ",round(PA2B,digits= 4))
```

```
## Probability of randomly chosen person having a BMI>30 being in the 35-49
## age group is 0.4728
```

d. If a randomly selected person had a BMI of above 30, what is the probability of that person being in the age group 50-64 years?

```
PA3B=(byPercent * bmiGT30Percent)[3] / pb
glue("Probability of randomly chosen person having a BMI>30 being in the 50-64
     age group is ", round(PA3B,digits = 4))
```

```
## Probability of randomly chosen person having a BMI>30 being in the 50-64
## age group is 0.1814
```

e. If a randomly selected person had a BMI of above 30, what is the probability of that person being in the 65 years & over?

```
PA4B=(byPercent * bmiGT30Percent)[4] / pb
glue("Probability of randomly chosen person having a BMI>30 being in the 50-64
     age group is ", round(PA4B,digits = 4))
```

```
## Probability of randomly chosen person having a BMI>30 being in the 50-64
## age group is 0.0523
```

Part 2. Random Variables

```
Consider a game which involves rolling three dice. Write the R code for the following.
Using the rollDie function from the prob library, setup the sample space for this experiment with t
For each of the following scenarios from a) through e), show the corresponding outcomes and the prob
```

a. The sum of the rolls is greater than 10.

```
## Roll 3 dice and see all of the outcomes
S<- rolldie(3, makespace = TRUE)
dieSum <- subset(S,(X1+X2+X3>10) )
Prob(dieSum)
```

```
## [1] 0.5
```

```
#check
# dieSum <- sum(dieSum$probs)
# head(dieSum,n=3)
```

b. All the three rolls are identical.

```
## Roll 3 dice and see all of the outcomes
S<- rolldie(3, makespace = TRUE)
A<-subset(S,(X1==X2) & (X1==X3) & (X2 == X3))
Prob(A)
```

```
## [1] 0.02777778
```

```
# check
# sum(A$probs)
# head(A,n=3)
```

c. Only two of the three rolls are identical.

```
S<- rolldie(3, makespace = TRUE)


# NOW take the UNION of dieEqual with A.  A comes from prob 2b.
# this will remove the x1=x2=x3 attributes

# create 3 subsets and take the union

# X1==X2, X3 not included
t<- subset(S,X1==X2 & X1 != X3)

# X1==X3, X2 not included
u<- subset(S,X1==X3 & X1 != X2)
```

```r
# X2==X3, X1 not included
v<- subset(S,X2==X3 & X1 != X2)

Prob(union(t,u,v))
```

```
## [1] 0.2777778
```

```r
#check
# union(t,u,v)
```

d. None of the three rolls are identical

```r
S<- rolldie(3, makespace = TRUE)
#S
dieNotEqual <- subset(S, (X1 != X2) & (X1 != X3) & (X2 != X3) )
Prob(dieNotEqual)
```

```
## [1] 0.5555556
```

```r
# sum(dieNotEqual$probs)
# head(dieNotEqual,n=3)
```

e. Only two of the three rolls are identical given that the sum of the rolls is greater than 10.

```r
S<- rolldie(3, makespace = TRUE)

# X1==X2 or X1==X3 or X2==X3
A <- subset(S, (X1 == X2) | (X1 == X3) | X2 == X3  )
B <- subset(S,(X1 + X2 + X3 > 10))
Prob(intersect(A,B))
```

```
## [1] 0.2222222
```

```r
#check
# u<- intersect(A,B)
# head(u,n=3)
```

Part 3. Functions

Using a for loop or a while loop, write your own R function, sum_of_first_N_odd_squares (n), that returns the sum of the squares of the first n odd numbers.

sum_of_first_N_odd_squares()

```
#' sum_of_first_N_odd_squares
#'
#' @param x Int that defines the first X odd numbers to be operated on
#'
#' @return Sum of the squares of the odd values
#' @export
#'
#' @examples  sum_of_first_N_odd_squares - returns 1^2 + 3^2 + 5^2 + 7^2 + 9^2 = 165
sum_of_first_N_odd_squares <- function(x){
#create a seq for x - but DOUBLE the size since we are taking every other one
  x<-seq(1:(x*2))

  #create an empty vector for our output
  k<- c()
  for(n in x){
    if (n %% 2 == 1){  # n %% 2 == 1 searches for the odd number
      k <- c(k,n)      # and stores them in k
    }
  }
  return (sum(k^2))
}
```

```
sum_of_first_N_odd_squares(2)
```

```
## [1] 10
```

```
sum_of_first_N_odd_squares(5)
```

```
## [1] 165
```

```
sum_of_first_N_odd_squares(10)
```

```
## [1] 1330
```

sum_of_first_N_odd_squaresV2()

```
#' sum_of_first_N_odd_squaresV2
#'
#' @param x Int that defines the first X odd numbers to be operated on
#'
#' @return Sum of the squares of the odd values
```

```r
#' @export
#'
#' @examples  sum_of_first_N_odd_squares - returns 1^2 + 3^2 + 5^2 + 7^2 + 9^2 = 165
sum_of_first_N_odd_squaresV2 <- function(x){
#create a seq for x double the size since we are taking every other one
  x<-seq(1,(x*2),2)
  return (sum(x^2))
}
```

The output for the Version 2 function

```r
sum_of_first_N_odd_squaresV2(2)
```

```
## [1] 10
```

```r
sum_of_first_N_odd_squaresV2(5)
```

```
## [1] 165
```

```r
sum_of_first_N_odd_squaresV2(10)
```

```
## [1] 1330
```

4. Using R

`Initialize the Dow Jones Industrials daily closing data, dow, using the read.csv function with the link: http://people.bu.edu/kalathur/datasets/DJI_2020.csv`

Store the result of the summary function for the Close attribute as the variable sm. Change

the names of this variable so that the output appears as shown below.

```
dow <- read_csv("http://people.bu.edu/kalathur/datasets/DJI_2020.csv",col_types = list(col_character(),c
head(dow)
```

```
## # A tibble: 6 x 2
##   Date   Close
##   <chr>  <dbl>
## 1 1/2/20 28869
## 2 1/3/20 28635
## 3 1/6/20 28703
## 4 1/7/20 28584
## 5 1/8/20 28745
## 6 1/9/20 28957
```

4a. Store the result of the summary() for the *close* attribute as the variable sm.

```
sm <- summary(dow$Close)
names(sm)<-c("Min","Q1","Q2","Mean","Q3","Max")
sm
```

```
##   Min    Q1    Q2  Mean    Q3   Max
## 18592 23466 24826 25544 28862 29551
```

```
paste("First Quartile variation is ",sm[2]-sm[1])
```

```
## [1] "First Quartile variation is  4873.5"
```

```
paste("Second Quartile variation is ",sm[3]-sm[2])
```

```
## [1] "Second Quartile variation is  1360.5"
```

```
paste("Third Quartile variation is ",sm[5]-sm[3])
```

```
## [1] "Third Quartile variation is  4035.5"
```

```
paste("Fourth Quartile variation is ",sm[6]-sm[5])
```

```
## [1] "Fourth Quartile variation is  689.5"
```

4b. Produce the output for the minumum of the Dow closing value in the datset as shown

```r
minDowClose <-min(dow$Close)
row_minDowClose <-which(dow$Close == min(dow$Close) )
date_minDowClose<- dow$Date[row_minDowClose]

paste("The minimum Dow value of ",minDowClose," is at row ",row_minDowClose," on ",date_minDowClose)
```

```
## [1] "The minimum Dow value of  18592  is at row  56  on  3/23/20"
```

4c. Suppose you have an index fund tied to the Dow closing value. If you have invested on the minimum date, what date from the dataset you would have sold to gain the maximum percentage gain. The output is as shown below. Note that the code should be generic so that it works on any such dataset.

```r
# Use the min values from above
# minDowClose
# date_minDowClose
# row_minDowClose

#end of the vector in use
e <- dim(dow)[1]

#slice the vector to only get a subset
w <- slice(dow, row_minDowClose: e)
maxDowClose <- max(w$Close)
# from the subset - select the max $Close value
hiRow <- which(w$Close==max(w$Close))
w[hiRow,1]
```

```
## # A tibble: 1 x 1
##   Date
##   <chr>
## 1 4/29/20
```

```r
gain <-  (maxDowClose / minDowClose) - 1

paste("I would sell on ",w[hiRow,1]," when DOW is at",max(w$Close)," for a gain of ",round(gain,digits =
```

```
## [1] "I would sell on  4/29/20  when DOW is at 24634  for a gain of  32.5 %"
```

4d. Use the diff function to calculate the differences between consecutive closing values in the dataset. Insert the value 0 at the beginning of these differences. Add this result as the DIFFS column of the data frame. The result is as shown below.

```r
# Create my new vector
cl<-diff(dow$Close)
# comparing lengths.. one is smaller than the other
length(cl)
```

```
## [1] 91
```

```r
length(dow$Close)
```

```
## [1] 92
```

```r
# Prepend a zero
cl<-prepend(cl,0)
length(cl)
```

```
## [1] 92
```

```r
# Add the diff column to the dow dataframe
dow$DIFFS<-cl
head(dow,n=6)
```

```
## # A tibble: 6 x 3
##   Date   Close DIFFS
##   <chr>  <dbl> <dbl>
## 1 1/2/20 28869     0
## 2 1/3/20 28635  -234
## 3 1/6/20 28703    68
## 4 1/7/20 28584  -119
## 5 1/8/20 28745   161
## 6 1/9/20 28957   212
```

4e. How many days did the Dow close higher than its previous day value? How many days did the Dow close lower than its previous day value?

```r
# Close higher

paste(length(dow$DIFFS[(dow$DIFFS>0)])," days DOW close higher than previous day")
```

```
## [1] "44  days DOW close higher than previous day"
```

```r
# Close lower



paste(length(dow$DIFFS[(dow$DIFFS<0)]),"  days DOW closed lower than previous day")
```

```
## [1] "47  days DOW closed lower than previous day"
```

4f. Show the subset of the data where there was a gain of at least 1000 points from its previous

day value.

```r
dow[(dow$DIFFS>1000),]
```

```
## # A tibble: 8 x 3
##   Date    Close DIFFS
##   <chr>   <dbl> <dbl>
## 1 3/2/20  26703  1294
## 2 3/4/20  27091  1174
## 3 3/10/20 25018  1167
## 4 3/13/20 23186  1985
## 5 3/17/20 21237  1048
## 6 3/24/20 20705  2113
## 7 3/26/20 22552  1351
## 8 4/6/20  22680  1627
```