# Ryan Timbrook

## **Applied Data Science**

## **IST687 Intro to Data Science**, Spring 2019

## **Due Date:** 05/1/2019

## **Homework:** 4

### NetID: RTIMBROO

### SUID: 386792749

## #R Code - unexecuted

# Clear objects from Memory

rm(list=ls())

# Clear Console:

cat("\014")

# Homework Week 4: Exploration of Sampling

# Step 1: Write a summarizing function to understand the distribution of a vector

# ---------------------------------------------------------------------------------

## 1.1: The summarizing function: printVecInfo

printVecInfo <- function(vec){

if(!require("moments")){install.packages("moments")}

cat("MEAN: ",round(mean(vec),2), "\nMEDIAN: ",round(median(vec),2), "\nMAX: ",max(vec), "\nMIN: ",min(vec),

"\nSD: ",round(sd(vec),2), "\nSKEWNESS: ",round(skewness(vec),2), "\nQUANTILE .05: ",quantile(vec,0.05), "\nQUANTILE .95: ",quantile(vec,0.95),"\n", sep = "")

}

## 1.2: Test printVecInfo function

v <- c(1,2,3,4,5,6,7,8,9,10,50)

printVecInfo(v)

# Step 2: Creating Samples in a Jar

# ---------------------------------------------------------------------------------

## 2.1 Functions:

### Create jar Container Object

createJarMarbles <- function(){

if(!require("stringr")){install.packages("stringr")}

#creating a container objec that holds 50 'red' and 50 'blue' -- marbles

jarMarbles <- c(replicate(50,"red"),replicate(50,"blue"))

return(jarMarbles)

}

### Jar Sampling Function - Return mean value of samples

sampleJarOfMarbles <- function(X,n,marble.type='red',rep=TRUE){

samp <- sample(X,n,replace=rep)

m <- length(samp[samp[]==marble.type])/length(X)

#return mean of sample set

return(m)

}

### Jar Replicate Sampling Function - Return list of mean values of samples

replicateSampleJarOfMarbles <- function(X,n=1,k=1){

sampMeans <- replicate(k,sampleJarOfMarbles(X,n))

return(sampMeans)

}

## 2.1(4) Create jar container object

jar <- createJarMarbles()

## 2.2(5) Confirm there are 50 reds

red.count <- length(jar[jar[] == 'red'])

red.count

## 2.3(6) Sample 10 'marbles' from the jar.

samp10 <- sample(jar,10,replace=TRUE)

### How many are red?

length(samp10[samp10=='red'])

### What is the percentage of red marbles?

cat(length(samp10[samp10=='red'])/length(samp10)\*100,"%","\n", sep="")

## 2.4(7) Do sampling 20 times - Generate list of 20 numbers - Each number is the mean of how many reds there were in 10 samples

## printVecInfo to see information of the samples

## generate histogram of the samples

#set.seed(42)

samp10.rep20 <- replicateSampleJarOfMarbles(jar,n=10,k=20)

length(samp10.rep20)

hist(samp10.rep20)

printVecInfo(samp10.rep20)

## 2.5(8) Repeat 2.4(7), with sampling at 100 times - Get 20 numbers, each represents the mean of reds there were in the 100 samples

samp100.rep20 <- replicateSampleJarOfMarbles(jar,n=100,k=20)

length(samp100.rep20)

hist(samp100.rep20)

printVecInfo(samp100.rep20)

## 2.5(9) Repeat 2.5(8), with sampling at 100 times - Get 100 numbers, each represents the mean of reds there were in the 100 samples.

samp100.rep100 <- replicateSampleJarOfMarbles(jar,n=100,k=100)

length(samp100.rep100)

hist(samp100.rep100)

printVecInfo(samp100.rep100)

# Step 3: Explore the airquality dataset

# -------------------------------------------------------------

## 3.1(10) Store the 'airquality' dataset into a temp variable

airQ <- airquality

## 3.2(11) Clean the dataset (i.e. remove the NAs) - replace with column means

na.2.mean <- function(x){

replace(x, is.na(x), mean(x, na.rm = TRUE))

}

airQ.clean <- replace(airQ, TRUE, lapply(airQ, na.2.mean))

## 3.3(12) Explore Ozone, Wind, and Temp by doing a 'printVecInfo' on each as well as generate historgram for each

## Ozone

printVecInfo(airQ.clean$Ozone)

hist(airQ.clean$Ozone)

## Wind

printVecInfo(airQ.clean$Wind)

hist(airQ.clean$Wind)

## Temp

printVecInfo(airQ.clean$Temp)

hist(airQ.clean$Temp)

## #R Code – executed

> # Homework Week 4: Exploration of Sampling

>

> # Step 1: Write a summarizing function to understand the distribution of a vector

> # ---------------------------------------------------------------------------------

> ## 1.1: The summarizing function: printVecInfo

> printVecInfo <- function(vec){

+ if(!require("moments")){install.packages("moments")}

+

+ cat("MEAN: ",round(mean(vec),2), "\nMEDIAN: ",round(median(vec),2), "\nMAX: ",max(vec), "\nMIN: ",min(vec),

+ "\nSD: ",round(sd(vec),2), "\nSKEWNESS: ",round(skewness(vec),2), "\nQUANTILE .05: ",quantile(vec,0.05), "\nQUANTILE .95: ",quantile(vec,0.95),"\n", sep = "")

+ }

>

> ## 1.2: Test printVecInfo function

> v <- c(1,2,3,4,5,6,7,8,9,10,50)

> printVecInfo(v)

MEAN: 9.55

MEDIAN: 6

MAX: 50

MIN: 1

SD: 13.72

SKEWNESS: 2.62

QUANTILE .05: 1.5

QUANTILE .95: 30

>

>

> # Step 2: Creating Samples in a Jar

> # ---------------------------------------------------------------------------------

> ## 2.1 Functions:

> ### Create jar Container Object

> createJarMarbles <- function(){

+ if(!require("stringr")){install.packages("stringr")}

+

+ #creating a container objec that holds 50 'red' and 50 'blue' -- marbles

+ jarMarbles <- c(replicate(50,"red"),replicate(50,"blue"))

+

+ return(jarMarbles)

+ }

>

> ### Jar Sampling Function - Return mean value of samples

> sampleJarOfMarbles <- function(X,n,marble.type='red',rep=TRUE){

+

+ samp <- sample(X,n,replace=rep)

+ m <- length(samp[samp[]==marble.type])/length(X)

+ #return mean of sample set

+ return(m)

+ }

> ### Jar Replicate Sampling Function - Return list of mean values of samples

> replicateSampleJarOfMarbles <- function(X,n=1,k=1){

+

+ sampMeans <- replicate(k,sampleJarOfMarbles(X,n))

+

+ return(sampMeans)

+ }

>

> ## 2.1(4) Create jar container object

> jar <- createJarMarbles()

>

> ## 2.2(5) Confirm there are 50 reds

> red.count <- length(jar[jar[] == 'red'])

> red.count

[1] 50

>

> ## 2.3(6) Sample 10 'marbles' from the jar.

> samp10 <- sample(jar,10,replace=TRUE)

>

> ### How many are red?

> length(samp10[samp10=='red'])

[1] 3

>

> ### What is the percentage of red marbles?

> cat(length(samp10[samp10=='red'])/length(samp10)\*100,"%","\n", sep="")

30%

>

> ## 2.4(7) Do sampling 20 times - Generate list of 20 numbers - Each number is the mean of how many reds there were in 10 samples

> ## printVecInfo to see information of the samples

> ## generate histogram of the samples

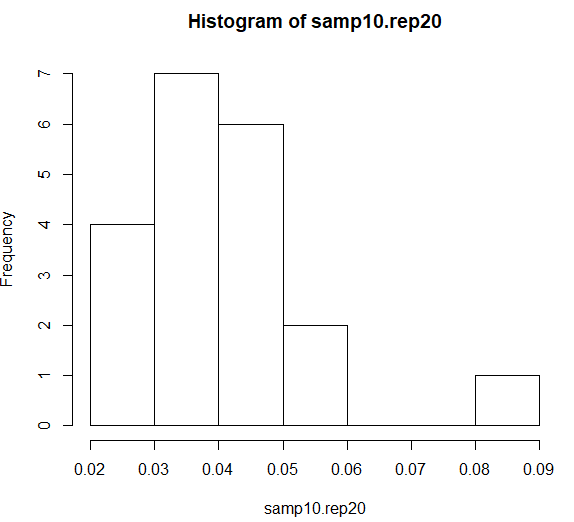
> #set.seed(42)

> samp10.rep20 <- replicateSampleJarOfMarbles(jar,n=10,k=20)

> length(samp10.rep20)

[1] 20

> hist(samp10.rep20)



> printVecInfo(samp10.rep20)

MEAN: 0.04

MEDIAN: 0.04

MAX: 0.09

MIN: 0.02

SD: 0.02

SKEWNESS: 0.99

QUANTILE .05: 0.02

QUANTILE .95: 0.0615

>

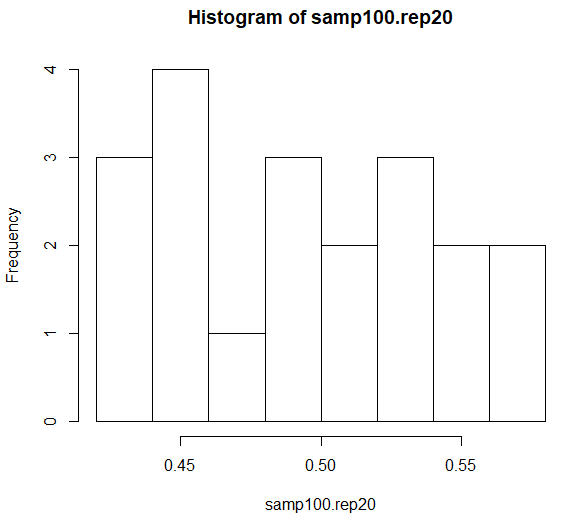
> ## 2.5(8) Repeat 2.4(7), with sampling at 100 times - Get 20 numbers, each represents the mean of reds there were in the 100 samples

> samp100.rep20 <- replicateSampleJarOfMarbles(jar,n=100,k=20)

> length(samp100.rep20)

[1] 20

> hist(samp100.rep20)



> printVecInfo(samp100.rep20)

MEAN: 0.5

MEDIAN: 0.5

MAX: 0.58

MIN: 0.42

SD: 0.05

SKEWNESS: 0.09

QUANTILE .05: 0.439

QUANTILE .95: 0.5705

>

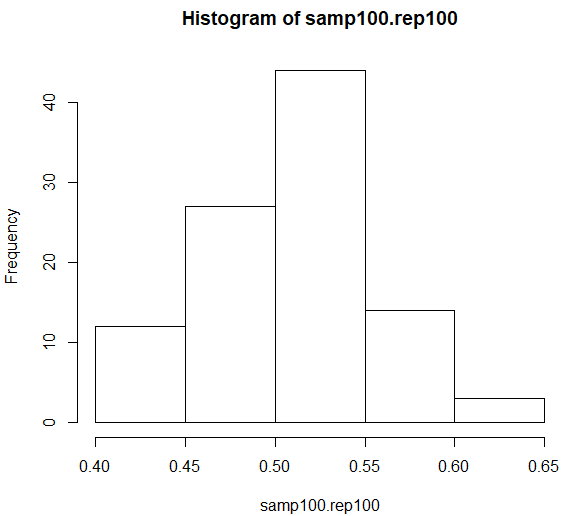
> ## 2.5(9) Repeat 2.5(8), with sampling at 100 times - Get 100 numbers, each represents the mean of reds there were in the 100 samples.

> samp100.rep100 <- replicateSampleJarOfMarbles(jar,n=100,k=100)

> length(samp100.rep100)

[1] 100

> hist(samp100.rep100)



> printVecInfo(samp100.rep100)

MEAN: 0.51

MEDIAN: 0.51

MAX: 0.63

MIN: 0.4

SD: 0.05

SKEWNESS: -0.02

QUANTILE .05: 0.4295

QUANTILE .95: 0.5805

>

> # Step 3: Explore the airquality dataset

> # -------------------------------------------------------------

> ## 3.1(10) Store the 'airquality' dataset into a temp variable

> airQ <- airquality

>

> ## 3.2(11) Clean the dataset (i.e. remove the NAs) - replace with column means

> na.2.mean <- function(x){

+ replace(x, is.na(x), mean(x, na.rm = TRUE))

+ }

> airQ.clean <- replace(airQ, TRUE, lapply(airQ, na.2.mean))

>

> ## 3.3(12) Explore Ozone, Wind, and Temp by doing a 'printVecInfo' on each as well as generate historgram for each

> ## Ozone

> printVecInfo(airQ.clean$Ozone)

MEAN: 42.13

MEDIAN: 42.13

MAX: 168

MIN: 1

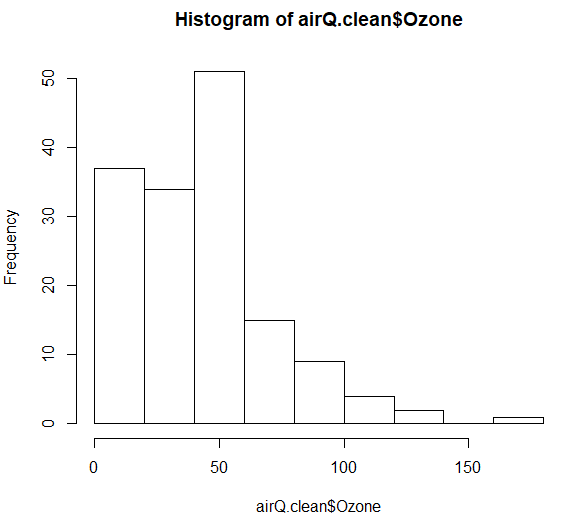
SD: 28.69

SKEWNESS: 1.41

QUANTILE .05: 9

QUANTILE .95: 97

> hist(airQ.clean$Ozone)

> 

> ## Wind

> printVecInfo(airQ.clean$Wind)

MEAN: 9.96

MEDIAN: 9.7

MAX: 20.7

MIN: 1.7

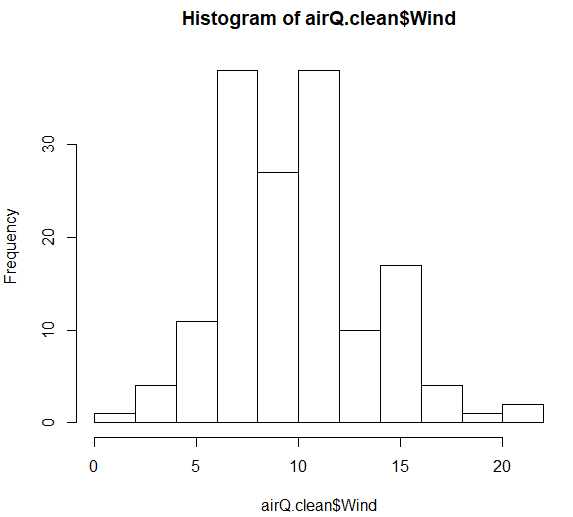
SD: 3.52

SKEWNESS: 0.34

QUANTILE .05: 4.6

QUANTILE .95: 15.5

> hist(airQ.clean$Wind)

> 

> ## Temp

> printVecInfo(airQ.clean$Temp)

MEAN: 77.88

MEDIAN: 79

MAX: 97

MIN: 56

SD: 9.47

SKEWNESS: -0.37

QUANTILE .05: 60.2

QUANTILE .95: 92

> hist(airQ.clean$Temp)

