**MINI PROJECT 6**

**By- Raghav Mathur (rxm162130), Deepak Shanmugam (dxs161930)**

**Contribution to the Project:** Each member contributed equally to the project. We worked together to analyse the dataset and in estimating the variables using bootstrap (using boot() and manually created functions).

**EXERCISE 1:**

**Section 1:**

We estimated the natural logarithm of population mean using boot() method in R and 95% of confidence interval using boot.ci() method. Also, we estimated the natural logarithm of population mean using manual calculations (code) (Refer Section 2 ##Manual Calculation) which are depicted as functions. We observed that bootstrap with CI normal approximation is helpful compared to z- critical point which may not be accurate as n (30) is not very large. Bootstrap Bias values is very small signifying a good CI. Also the values obtained using boot() method and manual created functions are comparable and the slight difference in values arise due to resampling of data.

##Results for boot method:

ORDINARY NONPARAMETRIC BOOTSTRAP

Call:

boot(data = cpu, statistic = logmean.func, R = 1000, sim = "ordinary",

stype = "i")

Bootstrap Statistics :

original bias std. error

t1\* 3.87605 -0.0012747 0.09758515

#bootstrap.bias

[1] -0.0012747

#bootstrap.se

[1] 0.09758515

#2.5th percentile of Theta.hat

2.5%

3.683008

>

#97.5th percentile of Theta.hat

97.5%

4.065087

>

#2.5th percentile of Theta.hat-theta

2.5%

-0.1930423

>

#97.5th percentile of Theta.hat-theta

97.5%

0.189037

##Results for boot.ci method:

BOOTSTRAP CONFIDENCE INTERVAL CALCULATIONS

Based on 1000 bootstrap replicates

CALL :

boot.ci(boot.out = logmean.boot)

Intervals :

Level Normal Basic

95% ( 3.686, 4.069 ) ( 3.685, 4.070 )

Level Percentile BCa

95% ( 3.682, 4.067 ) ( 3.695, 4.088 )

Calculations and Intervals on Original Scale

##Results for Manual Calculation

#std.error

[1] 0.09912876

#bias

[1] 0.0007544702

#2.5th percentile of Theta.hat

2.5%

3.682169

>

#97.5th percentile of Theta.hat

97.5%

4.053537

>

#2.5th percentile of Theta.hat-theta

2.5%

-0.1938809

>

#97.5th percentile of Theta.hat-theta

97.5%

0.1774867

#Normal bootstrap CI

[1] 3.681007 4.069585

#Basic bootstrap CI

[1] 3.698578 4.070750

#Percentile bootstrap CI

[1] 3.681351 4.053523

**Section 2: (R Code)**

install.packages("boot")

library(boot)

#Imported the dataset “cputime.txt”

cpu <- scan(file="/Users/deepaks/Desktop/UTD/Semester 2/Statistical Methods for Data Science/Miniproject/Project 6/cputime.txt")

#Statistic function for boot method

logmean.func=function(x,indices){

result= log(mean(x[indices]))

return(result)

}

#boot method to calculate sample estimate, standard error and bias

logmean.boot=boot(cpu,logmean.func,R=1000,sim="ordinary", stype = "i")

#using boot method

bootstrap.mean=logmean.boot$t0

bootstrap.se=sd(logmean.boot$t)

bootstrap.bias=mean(logmean.boot$t)-bootstrap.mean

#2.5th percentile of Theta.hat

quantile(logmean.boot$t,0.025)

#97.5th percentile of Theta.hat

quantile(logmean.boot$t,0.975)

#2.5th percentile of Theta.hat-theta

quantile(logmean.boot$t-bootstrap.mean,0.025)

#97.5th percentile of Theta.hat-theta

quantile(logmean.boot$t-bootstrap.mean,0.975)

#bootstrap CI

boot.ci(logmean.boot)

##Manual Calculation

n=1000

#Theta

theta = log(mean(cpu))

#Theta.hat which consists of 1000 samples calculated with replacement

theta.hat = replicate(n, log(mean(sample(cpu,length(cpu), replace=TRUE))))

#Bias Function

bias.func <- function(theta.hat , theta){

return (mean(theta.hat)- theta)

}

#Variance Function to calculate variance whose result is used for calculated standard error

variance.func <- function(theta.hat){

sum.val = sum((theta.hat-mean(theta.hat))\*(theta.hat-mean(theta.hat)))

return((sum.val)/(n-1))

}

#Standard Error

std.error = sqrt(variance.func(theta.hat))

#Bias

bias = bias.func(theta.hat,theta)

#2.5th percentile of Theta.hat

quantile(theta.hat,0.025)

#97.5th percentile of Theta.hat

quantile(theta.hat,0.975)

#2.5th percentile of Theta.hat-theta

quantile(theta.hat-theta,0.025)

#97.5th percentile of Theta.hat-theta

quantile(theta.hat-theta,0.975)

alpha = 1 - .95

#Sort the 1000 samples

sorted.theta.hat = sort(theta.hat)

#Normal Bootstrap CI function

normal.bootstrap.ci<-function(theta.hat,theta,alpha,bias){

c(theta - bias - qnorm(1-alpha/2)\*std.error,theta - bias - qnorm(alpha/2)\*std.error)

}

#Critical Point Calculation for basic and percentile bootstrap

critical.point <- function(data,n,alpha){

return((data[(n+1)\*alpha]))

}

#Basic Bootstrap CI function

basic.bootstrap.ci <- function(sorted.theta.hat,n,alpha, theta){

c(2\*theta - critical.point(sorted.theta.hat,n,(1-alpha/2)) ,2\*theta - critical.point(sorted.theta.hat,n,(alpha/2)))

}

#Percentile Bootstrap CI function

percentile.bootstrap.ci <- function(sorted.theta.hat,n,alpha){

c(critical.point(sorted.theta.hat,n,(alpha/2)), critical.point(sorted.theta.hat,n,(1-alpha/2)))

}

#Call the methods for normal, basic and percentile bootstrap CI

normal.bootstrap.ci(theta.hat,theta,alpha,bias)

basic.bootstrap.ci(sorted.theta.hat,b,alpha,theta)

percentile.bootstrap.ci(sorted.theta.hat,b,alpha)