**metodo estandar.**

la siguiente función calcula la curtosis de un vector x

curtosis <- function(x) {

curtosis <- sum(((x-mean(x))/sd(x))^4)/(length(x)-1) - 3

curtosis

}

La siguiente función calcula el intervalo de confianza para

la curtosis a través del método **estandar.**

curtosisicb=function(datos,q,b,alpha)

{

#q es la columna de la data que se va a trabajar

#b es el numero de muestra bootstrap

#alpha es el nivel significacion de 0 a 1

n=dim(datos)[1]

curtosisbvec=rep(0,b)

for (i in 1:b)

{

data=1:n

mcr=sample(data,n,T)

datosb=datos[mcr,q]

curtosisbvec[i]=curtosis(datosb)

}

meanb=mean(curtosisbvec)

sdb=sd(curtosisbvec)

lis=meanb-qnorm(1-alpha/2)\*sdb

lss=meanb+qnorm(1-alpha/2)\*sdb

print(lis)

print(lss)

anchoi=lss-lis

print(anchoi)

}

ejemplo del cálculo del IC para la curtosis

en la base de datos IRIS,columna 1 ,

con 100 muestras bootstrap y con nivel de significancia de 0.05

curtosisicb(iris,1,100,0.05)

**Método de los Percentiles**

la siguiente función calcula la curtosis de un vector x

curtosis <- function(x) {

curtosis <- sum(((x-mean(x))/sd(x))^4)/(length(x)-1) - 3

curtosis

}

La siguiente funcion calcula el intervalo de confianza para

la curtosis a travez del metodo **percentiles.**

curtosisicbp=function(datos,q,b,alpha)

{

#q es la columna de la data que se va a trabajar

#b es el numero de muestra bootstrap

#alpha es el nivel significacion de 0 a 1

n=dim(datos)[1]

curtosisbvec=rep(0,b)

for (i in 1:b)

{

data=1:n

mcr=sample(data,n,T)

datosb=datos[mcr,q]

curtosisbvec[i]=curtosis(datosb)

}

hist(curtosisbvec)

curtosisvec=sort(curtosisbvec)

lis=quantile(curtosisvec,alpha/2)

lss=quantile(curtosisvec,1-alpha/2)

print(lis)

print(lss)

anchoi=as.numeric(lss-lis)

print(anchoi)

}

ejemplo del calculo del IC para la curtosis

en la base de datos IRIS,columna 1 ,

con 100 muestras bootstrap y con nivel de significancia de 0.05

curtosisicbp(iris,1,100,0.10)

**Método bootstrapping estudentizado**

la siguiente funcion calcula la curtosis de un vector x

curtosis <- function(x) {

curtosis <- sum(((x-mean(x))/sd(x))^4)/(length(x)-1) - 3

curtosis

}

La siguiente función calcula el intervalo de confianza para

la curtosis a través del método **estudentizado.**

curtosisicstud=function(datos,q,b,alpha)

{

#q es la columna de la data que se va a trabajar

#b es el numero de muestra bootstrap

#alpha es el nivel significacion de 0 a 1

n=dim(datos)[1]

curtosisbvec=rep(0,b)

for (i in 1:b)

{

data=1:n

mcr=sample(data,n,T)

datosb=datos[mcr,q]

curtosisbvec[i]=curtosis(datosb)

}

escurtosisb=sd(curtosisbvec)

testud=rep(0,b)

curtorig=curtosis(datos[,q])

for (j in 1:b)

{

testud[j]=(curtosisbvec[j]-curtorig)/escurtosisb

}

stestud=sort(testud)

lis=curtorig+quantile(stestud,alpha/2)\*escurtosisb

lss=curtorig+quantile(stestud,1-alpha/2)\*escurtosisb

print(lis)

print(lss)

anchoi=as.numeric(lss-lis)

print(anchoi)

}

ejemplo del calculo del IC para la curtosis

en la base de datos IRIS,columna 1 ,

con 100 muestras bootstrap y con nivel de significancia de 0.05

curtosisicstud(iris,1,100,0.10)

**Bootstraping Doble (anidado)**

siguiente funcion calcula la curtosis de un vector x

curtosis <- function(x) {

curtosis <- sum(((x-mean(x))/sd(x))^4)/(length(x)-1) - 3

curtosis

}

La siguiente funcion calcula el intervalo de confianza para

la curtosis a travez del metodo Bootstraping Doble (anidado)

curtosisicboble=function(datos,q,b1,b2,alpha)

{

#q es la columna de la data que se va a trabajar

#b1 es el numero de muestra bootstrap

#b2 es el numero de muestra bootstrap en la segunda etapa

#alpha es el nivel significacion de 0 a 1

n=dim(datos)[1]

curtosisbvecb1=rep(0,b1)

escurtosisb=rep(0,b1)

for (i in 1:b1)

{

data=1:n

mcr=sample(data,n,T)

datosb=datos[mcr,q]

curtosisbvecb1[i]=curtosis(datosb)

curtosisbvecb2=rep(0,b2)

for (k in 1:b2)

{

datosb2= sample(datosb,n,T)

curtosisbvecb2[k]=curtosis(datosb2)

}

escurtosisb[i]=sd(curtosisbvecb2)

}

print(curtosisbvecb1)

print(escurtosisb)

testud=rep(0,b1)

curtorig=curtosis(datos[,q])

print(curtorig)

for (i in 1:b1)

{

testud[i]=(curtosisbvecb1[i]-curtorig)/ escurtosisb[i]

}

escurtosisb=sd(curtosisbvecb1)

stestud=sort(testud)

lis=curtorig+quantile(testud,alpha/2)\*escurtosisb

lss=curtorig+quantile(testud,1-alpha/2)\*escurtosisb

print(lis)

print(lss)

}

curtosisicboble (iris,1,100,30,0.10)