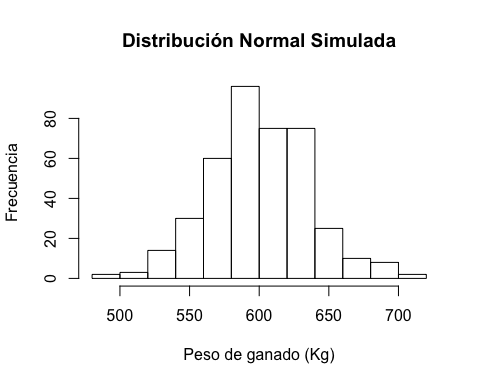
Estadistica Computacional : Trabajo 3

Jaime Gomez Marin

1.- Si el peso del ganado vacuno de una raza se distribuye según una Normal con media 600 kilo y desviación estándar de 35 simule los pesos de 400 de estos ganados, simule y grafique un histograma con los datos.

mu = 600  
sigma = 35  
n = 400  
  
# Primera forma para simular la distribucion normal   
dist\_normal\_1 <- function(n1, mu1, sigma1) {  
 x = rep(0,n1)  
 for(i in 1:n1) {  
 u1 = runif(12, min=0,max=1)  
 x[i] = sum(u1)-6  
 }  
 y = mu1 + sigma1\*x  
 return(y)  
}  
   
dnormal\_1 = dist\_normal\_1(n, mu, sigma)  
#dnormal\_1  
#hist(dnormal\_1 , xlab="Peso de ganado (Kg)", ylab ="Frecuencia",  
# main="Distribución Normal Simulada")

# Segunda forma para simular la distribucion normal   
dist\_normal\_2 <- function(n1, mu1, sigma1) {  
 x = rep(0,n1)  
 y = rep(0,n1)  
 for(i in 1:n1){  
 u1 = runif(1, min=0,max=1)  
 u2 = runif(1, min=0,max=1)  
 r = sqrt(-2\*log(u1))  
 theta = 2\*pi\*u2  
 x[i] = r\*cos(theta)  
 y[i] = r\*sin(theta)  
 X = mu1 + sigma1\*x  
 Y = mu1 + sigma1\*y  
 }  
   
 return(data.frame(X,Y))  
   
}  
   
dnormal\_2 = dist\_normal\_2(n, mu, sigma)  
hist(dnormal\_2$X , xlab="Peso de ganado (Kg)", ylab ="Frecuencia",  
 main="Distribución Normal Simulada")



2.- Al realizar un envío de obsequios por el día de la madre una tienda estimo que en el 6% de los casos los obsequios sufren accidentes, simule lo que pasa con 50 envíos.

Es una distribucion binomial

# Donde n es la cantidad de datos a simular  
# y theta es la probabilidad de exito  
bernoulli <- function(n,theta) {  
 uniformes=runif(n, min=0, max=1) # Mantenemos una linea constante   
# print(uniformes)  
 resultados=rep(0,n)  
 for (i in 1:n)   
 if (uniformes[i]>=1-theta)   
 resultados[i]=1  
 else   
 resultados[i]=0   
 return(resultados)   
}  
  
# Salida: 1 -> sufrio accidente , 0 -> no sufrio accidente  
bernoulli(1,0.06)

## [1] 1

# Funcion para generar numeros pseudoaleatorios   
# de una distribucion Binomial con parametros   
# r (nro. de repeticiones) y theta (segunda forma)  
binomial <-function(n,r,theta) {  
 x <- rep(0,n)  
   
 for (i in 1:r)  
 x[i]<-sum(bernoulli(n,theta))  
   
 #print(x)  
 prob = table(x)/r  
 return(prob)  
}  
  
n = 1 # tamanho de la muestra  
r = 50 # nro. de envios  
p = 0.06 # probabilidad que los regalos no llegen a su destino  
  
# Salida  
# x[0] : Numero de envios que llegan a su destino   
# x[1] : Numero de envios que NO llegan a su destino  
  
set.seed(100)  
r\*binomial(n, r, p)

## x  
## 0 1   
## 48 2

Se observa que hay 48 pedidos que llegaron a su destino y 2 pedidos que no llegaron a su destino

3.- Detalle un ejemplo de aplicación de la simulación con la distribución geométrica y geométrica negativa.

## Simulación con distribucion geométrica

Generar 12 numeros aleatorios que sigan una distribución geometrica con una probabilidad de 0.4%

#Funcion para generar numeros pseudoaleatorios de una distribucion geometrica con parametro theta  
geometrica <- function(n,theta) {  
 x=rep(0,n)  
 for (i in 1:n) {  
 U=runif(1)  
 x[i]=floor(log(U)/log(1-theta))  
 }  
 print(x)  
}  
  
geometrica(12,0.4)

## [1] 2 3 2 2 1 2 4 2 1 3 1 0

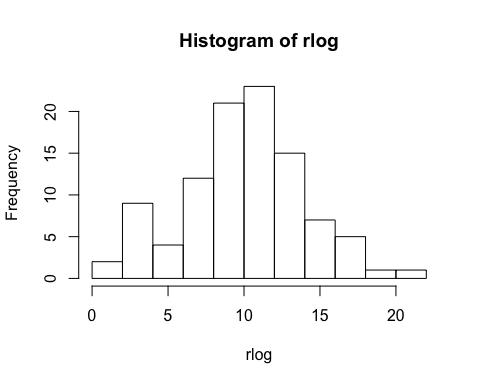
## Simulación con distribucion geométrica negativa

4.- Presente un contexto para simular 100 datos con las funciones rlogis y rweibull (defina los parámetros según el contexto que elija)

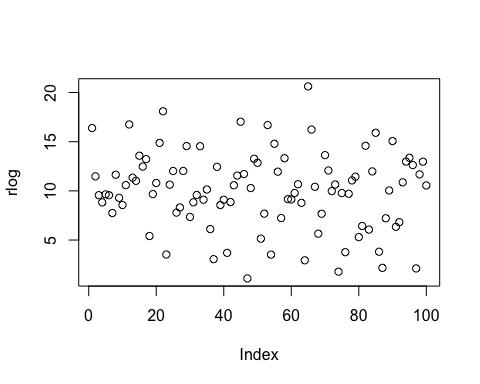
## Distribución Logistica

Ejemplo : Se tiene ganado vacuno con un parametro de posición de 10 y un parametor de scale =2, generar 100 numero pseudo aleatorios con este principio

n = 100  
rlog = rlogis(n,10,2)  
hist(rlog)



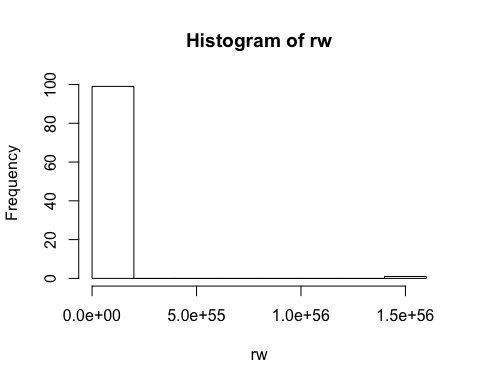
plot(rlog)



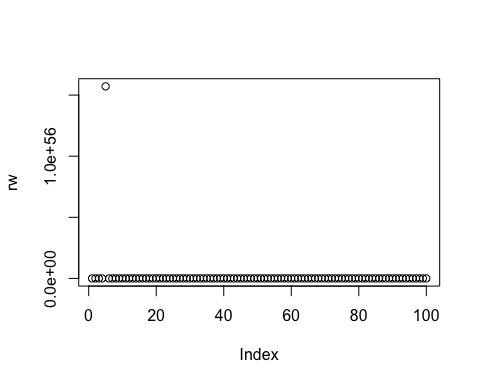
## Distribución Weibull

Ejemplo : El tiempo de via X en horas de un artefacto en el taller de mecanica tiene una distribución de Weibull con alpha igual a 0,01 y con beta igual a 2 . Se desea simular 100 observaciones

n = 100  
rw = rweibull(n, 0.01, 2)  
hist(rw)



plot(rw)



5.- Haga un programa que estime el coeficiente:

Pruebe la función para los siguientes datos : 12.150 12.000 10.875 11.400 12.525 12.000 10.050 10.650 11.700 11.550 10.800 9.600 9.450 11.850 10.950 11.400 12.150 12.600 12.525 11.850 10.800 12.000 13.80 13.125 12.075 12.00 13.725 12.600 12.300 11.700 12.450 12.300 11.700 11.700 12.750 13.50 13.050 12.30 2.90

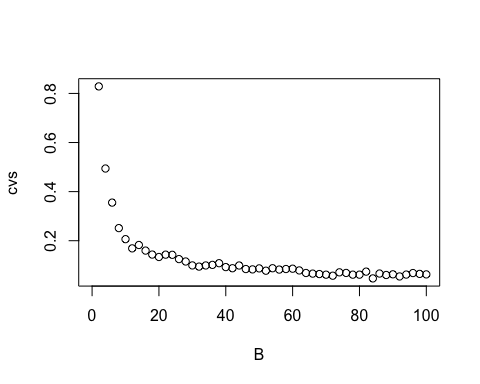
1. Cual es valor B de muestras bootstrap necesarios para la estimación del estadístico Ap2 ( considere 100 repeticiones)

# Calculo de estimados de ap2  
ap2 <- function(datos) {  
   
 media = mean(datos)  
 mediana = median(datos)   
 desviacion = sd(datos)  
 estimador = 3\*(3\*media - 2\*mediana)/(4\*desviacion)   
 return(estimador)  
  
}  
  
# Estimador bootstrap para RB  
calc\_bootstrap <- function(datos,B) {  
  
 n = length(datos)  
 estimac=rep(0,B)  
   
 for (i in 1:B){  
 muestra = sample(datos,n,T)  
 estimac[i] = ap2(muestra)  
 }  
   
 estboot=mean(estimac)  
 eeboot=sd(estimac)  
  
 return(list(estimador=estboot,eestandar=eeboot, vector=estimac))  
  
}  
  
# Repeticiones de muestras boostrap  
nro\_muestras\_bootstrap <- function(datos,B,r){  
   
 ee=rep(0,r)  
   
 for (i in 1:r)  
 ee[i] = calc\_bootstrap(datos,B)$eestandar  
  
 cv=sd(ee)/abs(mean(ee)) # Coeficiente de variabilidad  
  
 return(cv)  
}

set.seed(666)  
  
datos = c(12.150, 12.000, 10.875, 11.400, 12.525, 12.000, 10.050, 10.650, 11.700, 11.550, 10.800, 9.600, 9.450, 11.850, 10.950, 11.400, 12.150, 12.600, 12.525, 11.850, 10.800, 12.000, 13.80, 13.125, 12.075, 12.00, 13.725, 12.600, 12.300, 11.700, 12.450, 12.300, 11.700, 11.700, 12.750, 13.50, 13.050, 12.30, 2.90)  
  
#B = 10 # Numero de muestras bootstrap  
B = seq(0,100,2)  
nro\_rep = 100  
  
cvs = rep(0,length(B))   
  
for ( i in 1:length(B))  
 cvs[i] = nro\_muestras\_bootstrap(datos,B[i],r)  
  
cvs

## [1] NA 0.82863319 0.49410932 0.35525557 0.25137633 0.20631009  
## [7] 0.16883157 0.18255050 0.15975126 0.14358209 0.13365738 0.14320031  
## [13] 0.14256196 0.12516135 0.11531811 0.09936715 0.09450062 0.09913774  
## [19] 0.10158091 0.10842480 0.09257216 0.08786001 0.09902481 0.08457664  
## [25] 0.08316667 0.08728379 0.07779267 0.08784020 0.08244003 0.08467131  
## [31] 0.08600642 0.07893935 0.06847748 0.06594754 0.06419131 0.06158280  
## [37] 0.05742935 0.07116374 0.06866402 0.06169592 0.06169993 0.07404938  
## [43] 0.04645661 0.06630233 0.06032614 0.06312400 0.05438222 0.06205029  
## [49] 0.06829243 0.06478157 0.06282140

plot(B,cvs)

 Un valor adecuado de CV es menor o igual la 10%, entonces se puede asumir que el valor adecuado de B esta alrededor del 20

1. ¿Calcule el Intervalo de confianza para el estadístico Ap2, con los métodos: Estandar, percentiles, estudentizado y doblemente estudentizado.

# Metodo estandar

calc\_ic\_met\_estandar <-function(datos, B,alpha ) {  
 bstrap = calc\_bootstrap(datos,B)  
   
 media = bstrap$estimador  
 desvStd = bstrap$eestandar  
   
 lis = media - qnorm(1-alpha/2)\*desvStd  
 lss = media + qnorm(1-alpha/2)\*desvStd  
 rango = lss - lis  
   
 return(data.frame("metodo"="normal", lis, lss, rango))  
}  
  
#set.seed(666)  
#B = 20 # definido en la parte a) de la pregunta.  
#alpha = 0.05  
#calc\_ic\_met\_estandar(datos,B, alpha)

# Metodo percentiles

calc\_ic\_met\_percentiles <-function(datos,B,alpha ) {  
  
 bstrap = calc\_bootstrap(datos,B)  
   
 ap2vector = bstrap$vector  
   
 lis = quantile(ap2vector,alpha/2)  
 lss = quantile(ap2vector,1-alpha/2)  
  
 rango = lss - lis  
   
 return(data.frame("metodo"="percentiles", lis, lss, rango))  
}  
   
  
#set.seed(666)  
#  
#B = 20 # definido en la parte a) de la pregunta.  
#alpha = 0.05  
#calc\_ic\_met\_percentiles(datos,B,alpha)

# Metodo estudentizado

calc\_ic\_met\_estudentizado <-function(datos, B, alpha ) {  
  
 bstrap = calc\_bootstrap(datos,B)  
   
 ap2Std = bstrap$eestandar  
 ap2vector = bstrap$vector  
 ap2 = ap2(ap2vector)  
   
 testud=rep(0,B) # Inicializa el estudentizado  
   
 testud = (ap2vector-ap2)/ap2Std  
   
 lis = ap2 - quantile(testud,alpha/2)\*ap2Std   
 lss = ap2 + quantile(testud,1-alpha/2)\*ap2Std   
   
 rango = lss - lis  
   
 return(data.frame("metodo"="estudentizado", lis, lss, rango))  
}  
  
#set.seed(666)  
#alpha = 0.05 #  
#B = 20 # definido en la parte a) de la pregunta.  
# calc\_ic\_met\_estudentizado(datos, B,alpha)

# Metodo doblemente estudentizado

# Calcular IC usando el metodo de doble estudentizado  
calc\_ic\_met\_doble\_estudentizado <- function(datos,B1,B2,alpha) {  
  
 n = length(datos)  
 estimacB1 = rep(0,B1)  
 errorB1 = rep(0,B1)  
   
 for (i in 1:B1){  
 muestraB1 = sample(datos,n,T)  
 estimacB1[i] = ap2(muestraB1)  
   
 estimacB2 = rep(0,B2)  
 for (k in 1:B2) {  
 muestraB2 = sample(muestraB1,n,T)   
 estimacB2[k] = ap2(muestraB2)  
 }  
   
 errorB1[i] = sd(estimacB2)  
 }  
  
 testud = rep(0,B1) # t estudentizado , se hace para cada unos de la p muestras boostrap  
 estimacBASE = ap2(datos)  
   
 testud = (estimacB1-estimacBASE)/errorB1  
   
 ap2Std = sd(estimacB1)  
  
 lis = estimacBASE + quantile(testud,alpha/2)\*ap2Std   
 lss = estimacBASE + quantile(testud,1-alpha/2)\*ap2Std   
   
 rango = lss - lis  
   
 return(data.frame("metodo"="doble\_estudentizado", lis, lss, rango))  
}  
  
#set.seed(666)  
#  
#alpha = 0.05   
#B1 = 20 # definido en la parte a) de la pregunta.  
#B2 = 30 # definido en la parte a) de la pregunta.  
#  
#calc\_ic\_met\_doble\_estudentizado(datos,B1,B2,alpha)

set.seed(1000)  
  
alpha = 0.05  
B = 20 # definido en la parte a) de la pregunta.  
B1 = 20 # definido en la parte a) de la pregunta.  
B2 = 30 #   
  
df = rbind(calc\_ic\_met\_estandar(datos,B,alpha),   
calc\_ic\_met\_percentiles(datos,B,alpha),  
calc\_ic\_met\_estudentizado(datos,B,alpha),  
calc\_ic\_met\_doble\_estudentizado(datos,B1,B2,alpha))  
  
df

## metodo lis lss rango  
## 1 normal 1.834467 11.250908 9.416441  
## 2.5% percentiles 3.531009 9.184992 5.653983  
## 2.5%1 estudentizado 1.618970 9.592880 7.973910  
## 2.5%2 doble\_estudentizado 1.365026 15.115725 13.750699

1. Grafique los intervalos obtenidos en la parte b

boxplot(df$rango ~ df$metodo)

