Ciencia de datos, práctica 5

Juan Casado Ballesteros, Samuel García Gonzalez, Iván Anaya Martín

December 3, 2019

Abstract

Pendiente

¡¡functions, echo=False¿¿= frecuenciaAbsoluta ¡- function(data) uniquedata¡-
unique(data) uniquedata¡- sort(uniquedata) newdata¡- vector(mode="numeric",
length=0) for (value in uniquedata) ammount¡-length(data[data==value]) newdata¡-
c(newdata, ammount) setNames(newdata, uniquedata) frecuenciaRelativa ¡-
function(data) frecuencia ¡-frecuenciaAbsoluta(data) uniquedata¡-unique(data)
uniquedata¡- sort(uniquedata) newdata ¡- vector(mode="numeric", length=0)
for (value in 1:length(uniquedata)) newdata ¡- c(newdata, frecuencia[value]/length(data))
setNames(newdata, uniquedata) plotFrecuencyData ¡- function(data, xlabel="")
uniquedata¡-unique(data) frecuencia$_r elativa <- -as.table(frecuenciaRelativa(data))media_<-$
$mediaAritmetica(data)desviacion_t ipica_<-desviacionTipica(data)tchebychev_m in <$
$-media_-2*desviacion_t ipica_t chebychev_m ax <- -media_+2*desviacion_t ipica_m in_r ange =$
$min(tchebychev_m in, min(uniquedata))max_r ange = max(tchebychev_m in, max(uniquedata))plot(frecuencia$
$"h", xlab = xlabel, xlim = c(min_r ange, max_r ange))abline(v = c(media, tchebychev_m in, tchebychev_m ax), col$
$c("blue","gray","gray"), lty = c(2,2,2,3,3), lwd = c(2,2,2,1,1))legend("topright", legend =$
$c("media","2*desviacióntípica"), fill = c("blue","gray"))mediaAritmetica <$
$-function(data)acc <- 0for(valueindata)acc <- -acc + valueacc/length(data)varianza <$
$-function(data)v_m edia <- -mediaAritmetica(data)acc = 0for(valueindata)acc <- -acc + (value - v_m edia$
$-function(data)varianza(data)^(1/2)@$

1 K-vecinos sobre la muestra proporcionada para obtener outliers

Aplicaremos el algoritmo k-vecinos sobre la muestra que tenemos. Este algoritmo identificará de forma supervisada en la muestra datos anómalos, para poder obtener los outliers. En primer lugar deberemos cargar esta desde un archivo .txt. ¡¡cargar$_datos >>= datos1 < -read.table("datos1.txt")datos1@$

En segundo lugar calculamos las distancias euclídeas entre todos los puntos ¡¡distancias¿¿= distancias ¡- as.matrix(dist(datos1)) distancias @

Posteriormente, ordenamos las distancias de cada punto a todos los demás. ¡¡distancias$_ordenar >>= for(iin1 : length(distanciasordenadas[, 1]))distancias[, i] < -sort(distancias[, i])$
$-distancias@$

Como último paso, reordenamos la matriz para organizarla en función de la distancia de cada punto a su vecino número 1,2,3...etc. Tras haber organizado la matriz, buscamos en el tercer vecino, que es el valor k que hemos usado en nuestro análisis para poder identificar los outliers. ¡¡ordenacionfinalconoutlier¿¿= outliers$_kmeans = list()for(iin1 : length(distanciasordenadas[, 1]))if(distanciasordenadas[4, i] > 2.5)outlie$

2 Deteccion de datos anómalos sobre la resistencia

¡¡cargar$_datos >>= datos2 < -read.table("datos2.txt")datos2 < -data.frame(datos2)datos2@$

¡¡plot$_caja_bigotes, fig = TRUE >>=$ *boxplot*(*datos*2Resistencia, range=1.5, horizontal = TRUE) @

¡¡plot$_caja_bigotes, fig = TRUE >>=$ *boxplot*(*datos*2Densidad, range=1.5, horizontal = TRUE) @

3 Dispersión sobre la densidad, desviación típica

¡¡desviacion$_t ipica >>= datos2 < -read.table("datos2.txt")$@

$$¡¡\text{desviacion}_t ipica_d ensidad_p lot, fig = TRUE >>= plotFrecuencyData(datos2\text{Densidad})\ @$$

¡¡desviacion$_t ipica_d ensidad >>= int < -c(mediaAritmetica(datos2$Densidad)
- 2*desviacionTipica(datos2$Densidad), mediaAritmetica(datos2$Densidad) + 2*desviacionTipica(datos2$Den
$list()for(iin1 : length(datos2\text{Densidad}))$ if ((datos2$Densidad[i] < int[1])||(datos2$Densidad[i]¿int[2]))
outliers$_d esviacion[[length(outliers_d esviacion)+1]] < -t(matrix(c(i, datos2[i,]Densidad),
dimnames=list(c("Indice","Densidad")))) outliers$_d esviacion$@

$$¡¡\text{desviacion}_t ipica_r esistencia_p lot, fig = TRUE >>= plotFrecuencyData(datos2\text{Resistencia})\ @$$

¡¡desviacion$_t ipica_r esistencia >>= int < -c(mediaAritmetica(datos2Resistencia)
- 2*desviacionTipica(datos2$Resistencia), mediaAritmetica(datos2Resistencia)
+ 2*desviacionTipica(datos2$Resistencia))outliers$_d esviacion = list()for(iin1 :
length(datos2Resistencia)) if ((datos2$Resistencia[i] < int[1])||(datos2Resistencia[i]¿int[2]))
outliers$_d esviacion[[length(outliers_d esviacion)+1]] < -t(matrix(c(i, datos2[i,]Resistencia),
dimnames=list(c("Indice","Resistencia")))) outliers$_d esviacion$@