Anomalias_B1

Miguel Merelo Hernández

February 11, 2018

Crear datos

Utilizamos LetterRecognition del paquete mlbench. Solo utilizamos las primeras 200 líneas ya que la visualización de 20000 entradas hace imposible analizar el problema.

```
mydata.numeric.scaled<-scale(mydata.numeric,center=TRUE,scale=TRUE)
columna<-mydata.numeric[,indice.columna]
nombre.columna<-names(mydata.numeric[indice.columna])
columna.scaled<-scale(columna)
```

1. Cómputo de los outliers IQR

Calcular los outliers según la regla IQR. Directamente sin funciones propias.

```
cuartil.primero<-quantile(columna,0.25)
cuartil.tercero<-quantile(columna,0.75)
iqr<-IQR(columna)
iqr
```

```
## [1] 3
```

```
quantile(columna)
```

```
##   0%  25%  50%  75% 100%
##    1    3    4    6   12
```

Obtenemos que el 50% de los datos se encuentran entre los valores 3 y 6, que el valor máximo es 12 y el mínimo es 1. Con IQR=3 no podemos decir que la dispersión de los datos sea muy grande.

```
extremo.superior.outlier.normal  <- cuartil.tercero+1.5*iqr
extremo.inferior.outlier.normal  <- cuartil.primero-1.5*iqr
extremo.superior.outlier.extremo <- cuartil.tercero+3*iqr
extremo.inferior.outlier.extremo <- cuartil.primero-3*iqr

vector.es.outlier.normal <- columna>extremo.superior.outlier.normal |
  columna<extremo.inferior.outlier.normal
vector.es.outlier.extremo <- columna>extremo.superior.outlier.extremo |
  columna<extremo.inferior.outlier.extremo


print(paste("Extremo outlier normal.",
            "Inferior:",extremo.inferior.outlier.normal,
            "Superior:",extremo.superior.outlier.normal))
```

```
## [1] "Extremo outlier normal. Inferior: -1.5 Superior: 10.5"
```

```
print(paste("Extremo outlier extremo.",
            "Inferior:",extremo.inferior.outlier.extremo,
            "Superior:",extremo.superior.outlier.extremo))
```

```
## [1] "Extremo outlier extremo. Inferior: -6 Superior: 15"
```

```
print(paste("Indices de outliers normales:",
            paste(which(vector.es.outlier.normal),collapse = " ")))
```

```
## [1] "Indices de outliers normales: 10 28"
```

```
print(paste("Indices de outliers extremos:",
            paste(which(vector.es.outlier.extremo), collapse = " ")))
```

```
## [1] "Indices de outliers extremos: "
```

Consideramos como outlier normal a aquella entrada con valor menor a -1.5 o superior a 10.5 y como outlier extremo a aquellos con valores menores a -6 y superiores a 15. Para nuestros datos solo tenemos 2 valores de outliers normales y 0 extremos. Confirmamos que la dispersión de los datos es pequeña con solo un 1% de outliers.

Índices y valores de los outliers

```
claves.outliers.normales<-which(vector.es.outlier.normal)
data.frame.outliers.normales<-mydata.numeric[claves.outliers.normales,]
nombres.outliers.normales<-row.names(data.frame.outliers.normales)
valores.outliers.normales<-columna[claves.outliers.normales]
valores.outliers.normales
```

```
## [1] 11 12
```

Obtenemos que los dos únicos outliers de la columna que habíamos seleccionado tienen los valores 11 y 12.

Desviación de los outliers con respecto a la media de la columna.

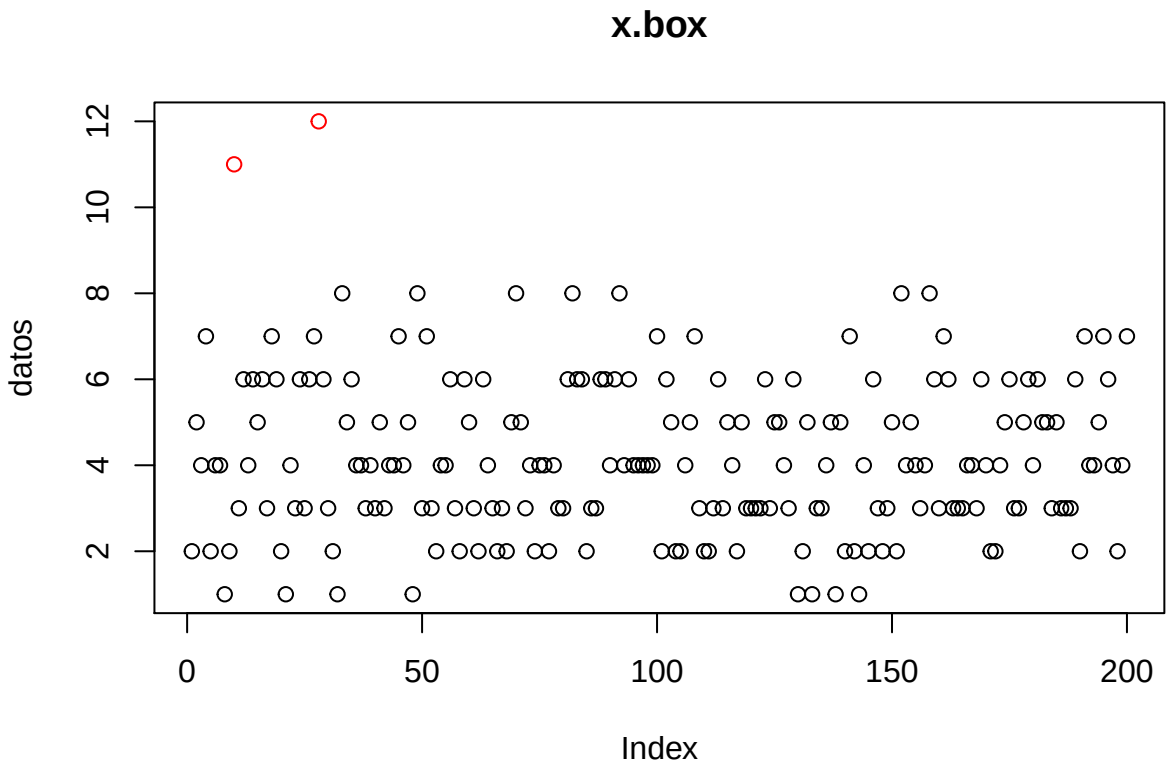
```
valores.normalizados.outliers.normales<-columna.scaled[vector.es.outlier.normal]
valores.normalizados.outliers.normales
```

```
## [1] 3.610215 4.141129
```

Valores de los outliers en el vector normalizado.

Plot

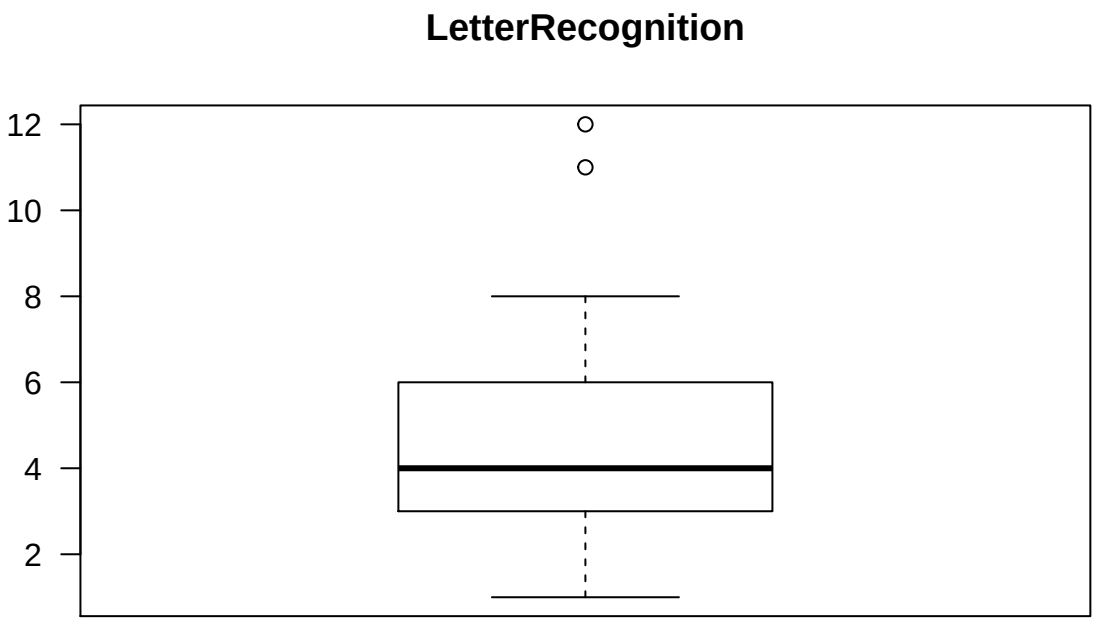
```
MiPlot_Univariate_Outliers(columna,claves.outliers.normales,nombre.columna)
```



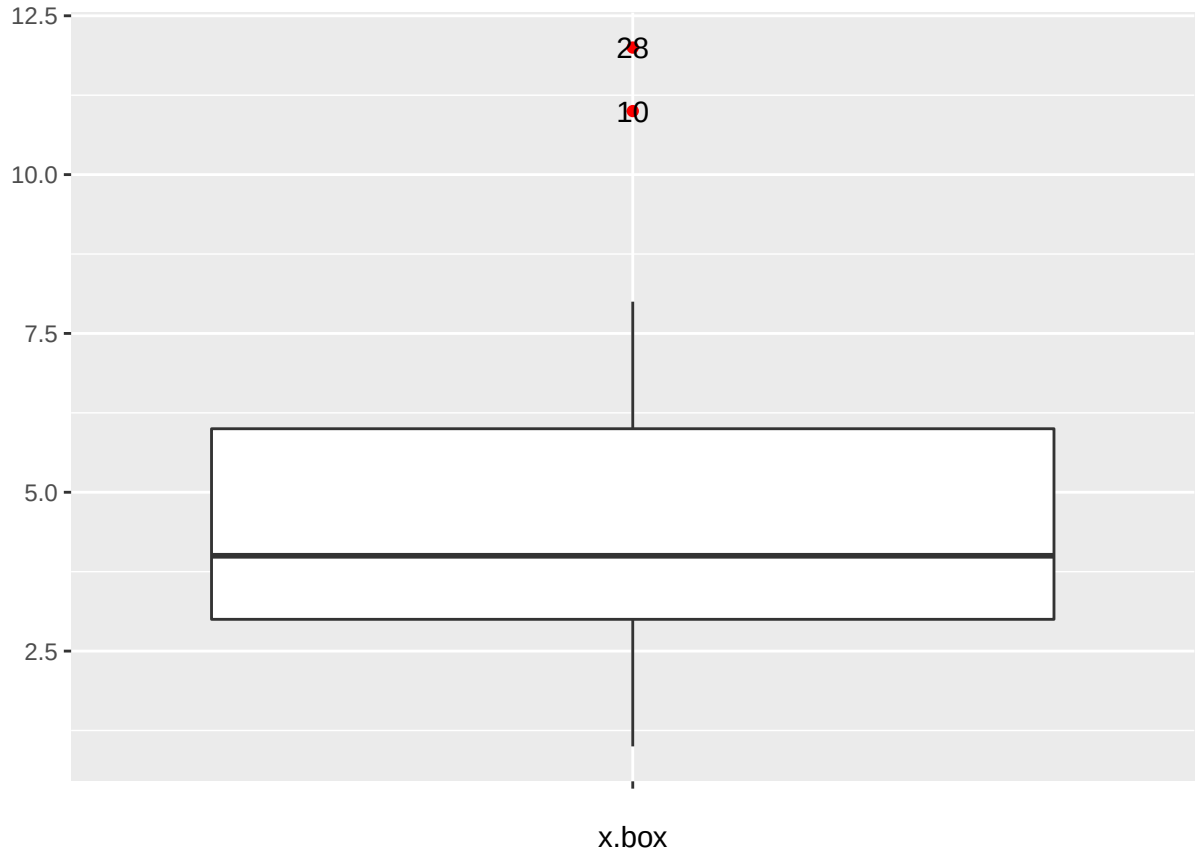
Vemos como para la 10ª y 28ª instancia tenemos los dos outliers con valor 11 y 12 respectivamente y que el resto de valores no supera 10.5 (valor a partir del cual tenemos outliers)

BoxPlot

```
boxplot(columna, xlab=nombre.columna, main=nombre.mydata, las = 1)
```




```
MiBoxPlot_IQR_Univariate_Outliers(mydata.numeric, indice.columna, coef = 1.5)
```



Representación de los dos outliers con el resto de datos utilizando boxplot. Los valores normales no superan el valor 8.

Cómputo de los outliers IQR con funciones propias.

```
head(vector_es_outlier_IQR(columna.scaled,indice.columna),n=10)
```

```
##  [1] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE  TRUE
```

```
head(vector_claves_outliers_IQR(columna.scaled,indice.columna),n=10)
```

```
## [1] 10 28
```

Comprobación de que nuestras funciones dan los mismo resultados que las predefinidas.

```
indices.de.outliers.en.alguna.columna<-
  vector_claves_outliers_IQR_en_alguna_columna(mydata.numeric)
indices.de.outliers.en.alguna.columna<-
  indices.de.outliers.en.alguna.columna[
    !duplicated(indices.de.outliers.en.alguna.columna)]
length(indices.de.outliers.en.alguna.columna)
```

```
## [1] 92
```

```
head(mydata.numeric[indices.de.outliers.en.alguna.columna,],n=10)
```

```
##     x.box y.box width high onpix x.bar y.bar x2bar y2bar xybar x2ybr xy2br
## 10     11    15    13    9     7    13     2     6     2    12     1     9
```

```
## 28     12    14    12    8     5     9    10     4     3     5    10     7
## 27      7    11    11    8     9     3     8     4     5    10    11    10
## 141     7    11    10    8     6     6    11     3     6    13     6     3
## 195     7    10    10    7     7     9     6     2     5     9     5     7
## 34      5     9     7    7    10     9     8     4     4     6     8     6
## 88      6     9     9    8    10     8     7     5     5     7     6     8
## 146     6    11     9    8    11     7     7     3     6     6     7    11
## 31      2     3     3    4     1     0     1     5     6     0     0     6
## 52      3     6     3    4     1     1     0     6     6     0     1     5
##     x.ege xegvy y.ege yegvx
## 10      8     1     1     8
## 28     10    12     2     6
## 27     10     9     5     7
## 141     0    10     3     8
## 195     8     6     2     8
## 34      6    11     8     7
## 88      8     9     9     3
## 146     6    11     9     7
## 31      0     8     0     8
## 52      0     8     0     8
```

Calculamos las instancias con outliers en alguna columna. Tenemos 92 instancias con outlier. Mostramos solo las 10 primeras instancias.

Ampliación

Índices y valores de los outliers

```
frame.es.outlier<-as.matrix(sapply(1:ncol(mydata.numeric),
                                   function(x,y){vector_es_outlier_IQR(y,x)},
                                   mydata.numeric))
head(frame.es.outlier)
```

```
##       [,1]  [,2]  [,3]  [,4]  [,5]  [,6]  [,7]  [,8]  [,9] [,10] [,11]
## [1,] FALSE FALSE FALSE FALSE FALSE FALSE  TRUE FALSE FALSE FALSE FALSE
## [2,] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
## [3,] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
## [4,] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
## [5,] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
## [6,] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##      [,12] [,13] [,14] [,15] [,16]
## [1,] FALSE FALSE FALSE FALSE FALSE
## [2,] FALSE FALSE FALSE FALSE  TRUE
## [3,] FALSE FALSE FALSE FALSE FALSE
## [4,] FALSE FALSE FALSE FALSE FALSE
## [5,] FALSE FALSE FALSE FALSE  TRUE
## [6,] FALSE FALSE FALSE FALSE FALSE
```

Como TRUE tenemos los valores que son outliers en esa columna.

```
numero.total.outliers.por.columna<-apply(frame.es.outlier,2,sum)
numero.total.outliers.por.columna
```

```
##  [1]  2  0  5  0  3 10 21  9  1  3  7 10 13  1  0 41
```

Número de outliers por columna. Podemos observar como la última columna es la que más outliers tiene con 41.

```
indices.de.outliers.en.alguna.columna<-unlist(
  sapply(1:ncol(mydata.numeric),
         function(x,y){vector_claves_outliers_IQR(y,x)},
         frame.es.outlier))
indices.de.outliers.en.alguna.columna<-
  indices.de.outliers.en.alguna.columna[
    !duplicated(indices.de.outliers.en.alguna.columna)]

head(indices.de.outliers.en.alguna.columna)
```

```
## [1]  10  28  27 141 195  34
```

Vector de indices de las instancias con algún outlier en alguna columna.

Desviación de los outliers con respecto a la media de la columna

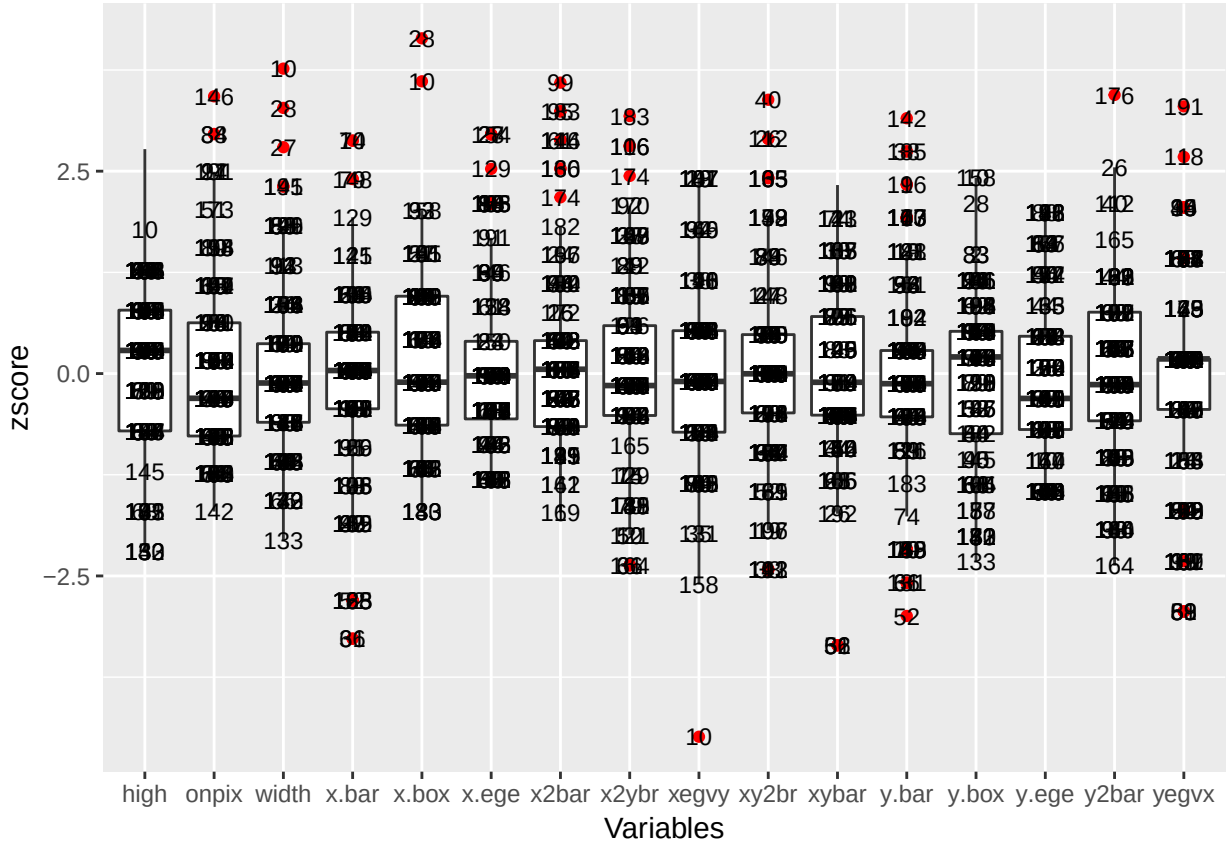
```
head(mydata.numeric.scaled[indices.de.outliers.en.alguna.columna,],n=10)
```

```
##          x.box      y.box      width       high      onpix       x.bar
## 10   3.6102153  2.4182312  3.7669823  1.7776177  1.5597890  2.87705153
## 28   4.1411293  2.1021225  3.2815464  1.2803820  0.6271797  0.98425447
## 27   1.4865592  1.1537966  2.7961105  1.2803820  2.4923982 -1.85494112
## 141  1.4865592  1.1537966  2.3106747  1.2803820  1.0934843 -0.43534332
## 195  1.4865592  0.8376879  2.3106747  0.7831463  1.5597890  0.98425447
## 34   0.4247312  0.5215793  0.8543671  0.7831463  2.9587028  0.98425447
## 88   0.9556452  0.5215793  1.8252388  1.2803820  2.9587028  0.51105521
## 146  0.9556452  1.1537966  1.8252388  1.2803820  3.4250074  0.03785594
## 31  -1.1680108 -1.3750726 -1.0873763 -0.7085609 -1.2380388 -3.27453891
## 52  -0.6370968 -0.4267467 -1.0873763 -0.7085609 -1.2380388 -2.80133965
##          y.bar       x2bar      y2bar      xybar      x2ybr        xy2br
## 10  -2.1745212  0.40716226 -1.4784905  1.5166019 -1.9977360  0.481156231
## 28   1.1046814 -0.30094601 -1.0311409 -1.3257578  1.3318240 -0.485991973
## 27   0.2848807 -0.30094601 -0.1364416  0.7044992  1.7017751  0.964730333
## 141  1.5145817 -0.65500015  0.3109080  1.9226533 -0.1479804 -2.420288381
## 195 -0.5349199 -1.00905429 -0.1364416  0.2984478 -0.5179316 -0.485991973
## 34   0.2848807 -0.30094601 -0.5837913 -0.9197064  0.5919218 -0.969566075
## 88  -0.1250196  0.05310812 -0.1364416 -0.5136550 -0.1479804 -0.002417871
## 146 -0.1250196 -0.65500015  0.3109080 -0.9197064  0.2219707  1.448304435
## 31  -2.5844215  0.05310812  0.3109080 -3.3560147 -2.3676871 -0.969566075
## 52  -2.9943218  0.40716226  0.3109080 -3.3560147 -1.9977360 -1.453140177
##         x.ege       xegvy      y.ege      yegvx
## 10   2.102702 -4.48618574 -1.0750228  0.1809413
## 28   2.954861  2.41081540 -0.6910861 -1.0669295
## 27   2.954861  0.52981509  0.4607241 -0.4429941
## 141 -1.305933  1.15681519 -0.3071494  0.1809413
## 195  2.102702 -1.35118522 -0.6910861  0.1809413
## 34   1.250543  1.78381530  1.6125342 -0.4429941
## 88   2.102702  0.52981509  1.9964710 -2.9387356
## 146  1.250543  1.78381530  1.9964710 -0.4429941
## 31  -1.305933 -0.09718502 -1.4589595  0.1809413
## 52  -1.305933 -0.09718502 -1.4589595  0.1809413
```

Mostramos solo las 10 primeras instancias normalizadas de las instancias con algún outlier.

Boxplot

```
MiBoxPlot_juntos_con_etiquetas(mydata.numeric)
```



Mostramos todos los outliers de las variables. Podemos comprobar como se corresponden con los datos antes calculados viendo que en la última columna tenemos mayor cantidad de outliers.