Escalamiento no metrico

2024-04-24

Librerias necesarias

```
library(car)
```

```
## Loading required package: carData
```

```
library(smacof)
```

```
## Loading required package: plotrix
```

```
## Loading required package: colorspace
```

```
## Loading required package: e1071
```

```
##
## Attaching package: 'smacof'
```

```
## The following object is masked from 'package:base':
##
##     transform
```

```
library(cluster)
library(lubridate)
```

```
##
## Attaching package: 'lubridate'
```

```
## The following objects are masked from 'package:base':
##
##     date, intersect, setdiff, union
```

```
source("utilerias/funciones.R")
```

Indicaciones:

Utiliza el siguiente analisis para despues hacer tu propio analisis, seleccionando la mejor dimension con los ratings(columnas 5:18) de la db RockHard del paquete smacof.

Sinapsis

El escalamiento multidimensional no métrico tiene por objetivo preservar las disimilaridades mientras se posicionan los objetos en una menor dimension. Se aplica principalmente sobre datos ordinales o ratings.

Pasos escenciales:

0.- Preprocesamiento(Datos, atípicos, escalamiento, transformaciones)

1.- Matriz de datos con escala ordinales(Definir rangos)

2.- Cálculo de disimilaridades

4.- Escalamiento no métrico (Regresión monótona)

5.- Mejoras

Carga de la informacion

Práctica sobre el Indice de Rezago Social en Mexico

Variables: Indices

```
data <- read.csv("Indica.csv")
```

Formato Correcto

```
rownames(data) <- data$Entidad.Federativa # Estableciendo como indice las entidades
data$Entidad.Federativa <- NULL # Estableciendo como indice las entidades

# Establecimiendo de escalas ordinales
data$Grado.de.rezago.social <- factor(data$Grado.de.rezago.social, levels= c("Muy bajo", "Bajo","Medio"
```

Seleccion de las columnas auxiliares y de analisis.

La informacion debe ser suministrada por el dueño de los datos. Por ejemplo definir el comportamiento del índice de rezago

```
auxiliares <- colnames(data[, c(2,15)])
analisis <- colnames(data[,3:12]) # Seleccion de columnas
columnas <- c(auxiliares, analisis)
datos <- data[, columnas] # Extraccion
```

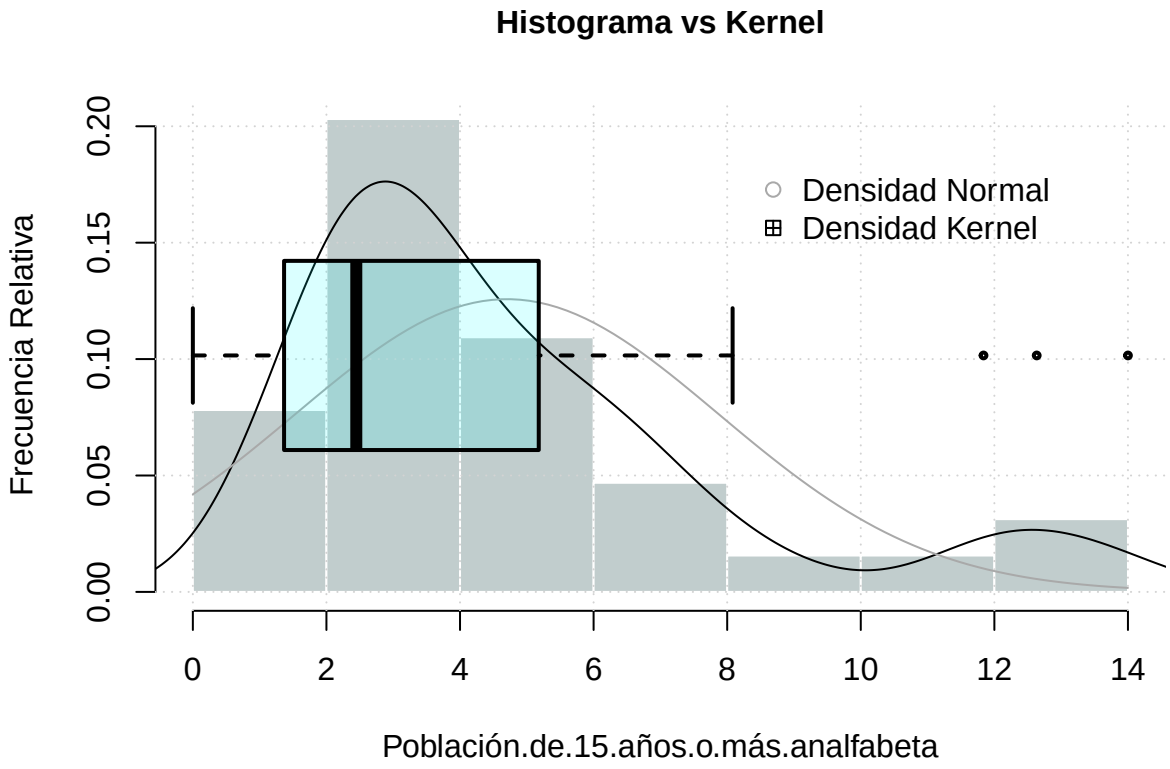
Escalas Iniciales

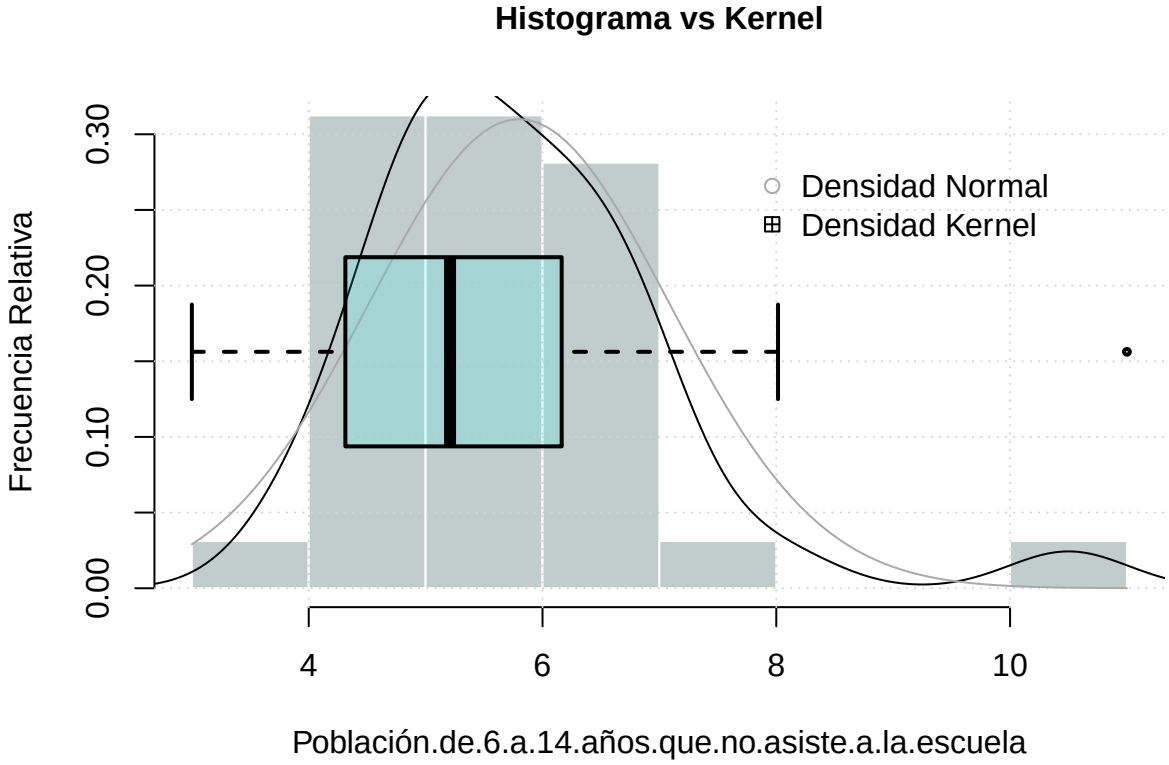
```
# Escalas
tipo <- sapply(datos, class)
continuas <-  which(tipo == "numeric") # continuas
enteras <- which(tipo == "integer") # enteras
numericas <- names(c(continuas,enteras))

# Variables Categóricas
nominales <- which( tipo == "factor") # categóricas
ordinales <- which( sapply(datos, is.ordered) )  # ordinales
fecha <- which(tipo == "Date") # Fecha
categoricas <- names(c(nominales, ordinales, fecha))
```

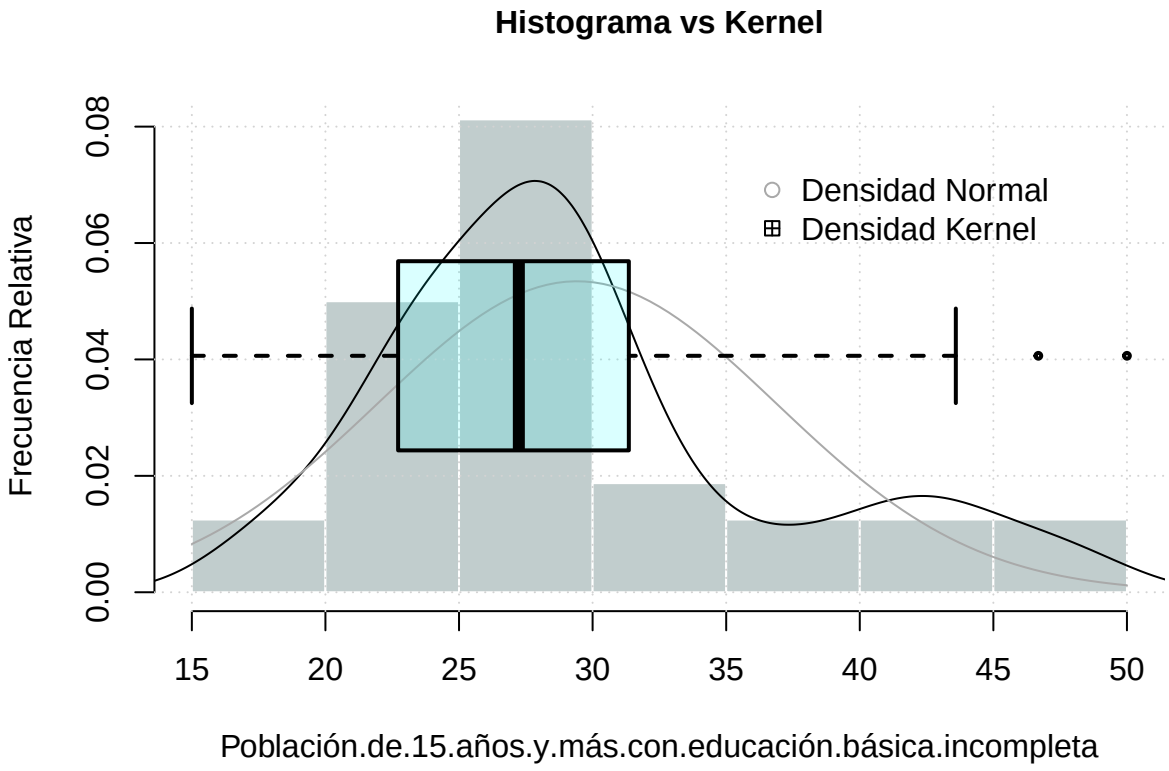
Descriptivos Multivariados

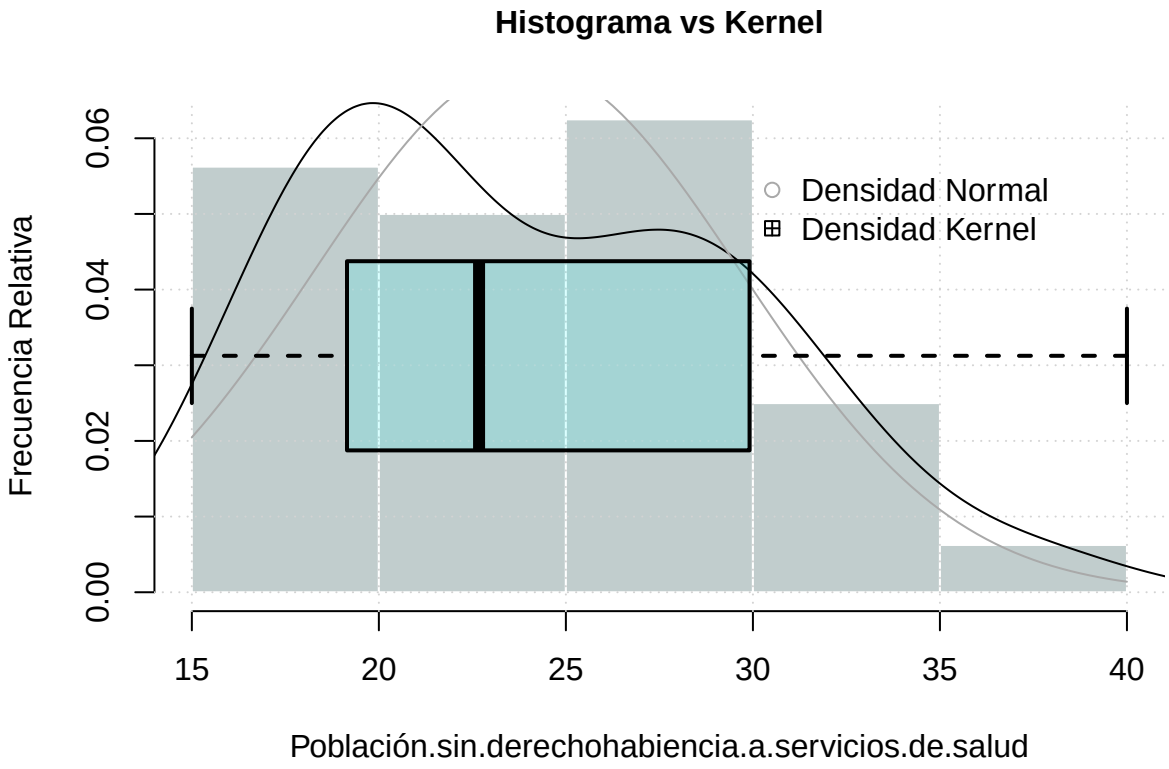
- Identificar Atipicos
- Problemas de escala
- Distribuciones

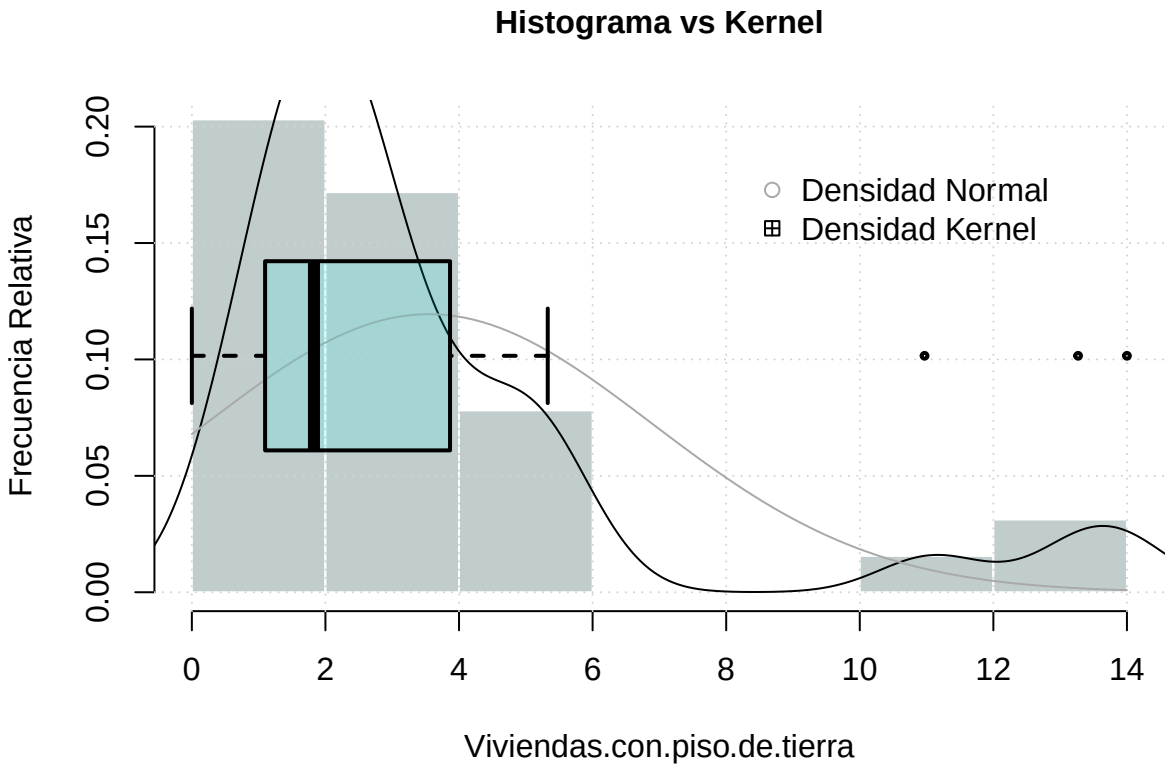
```r
# Histogramas
multi.hist(datos[, numericas])
```

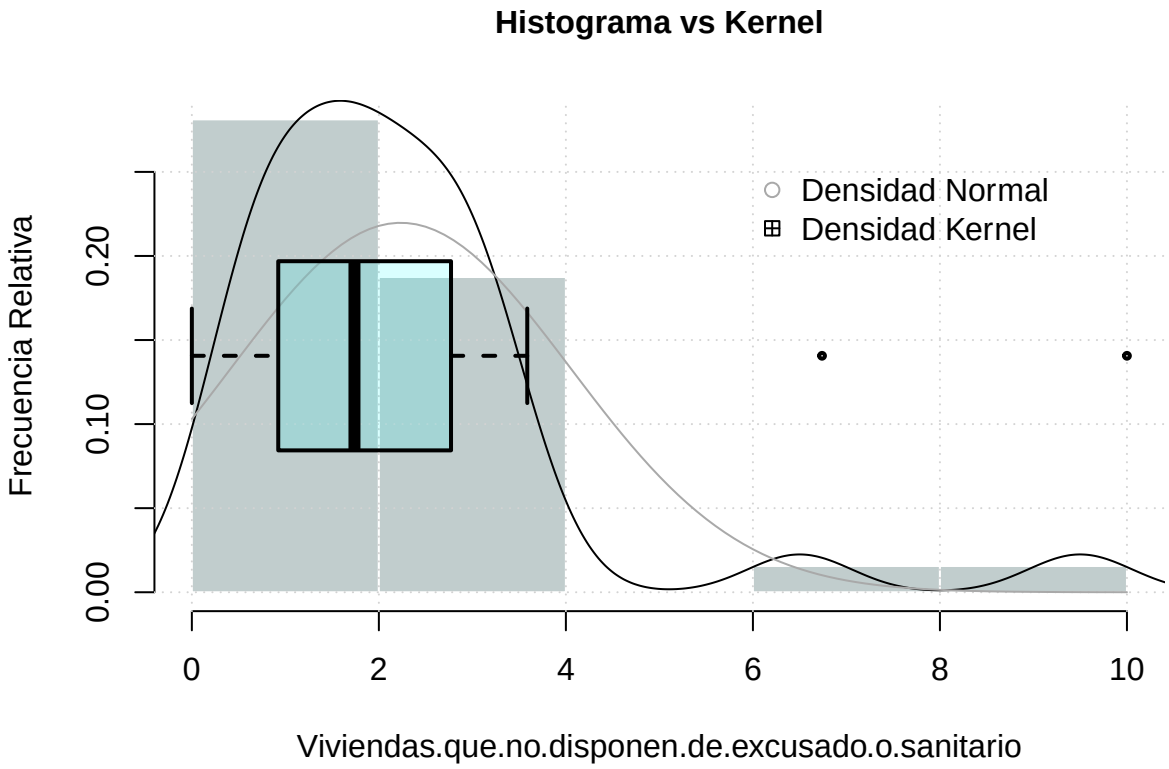


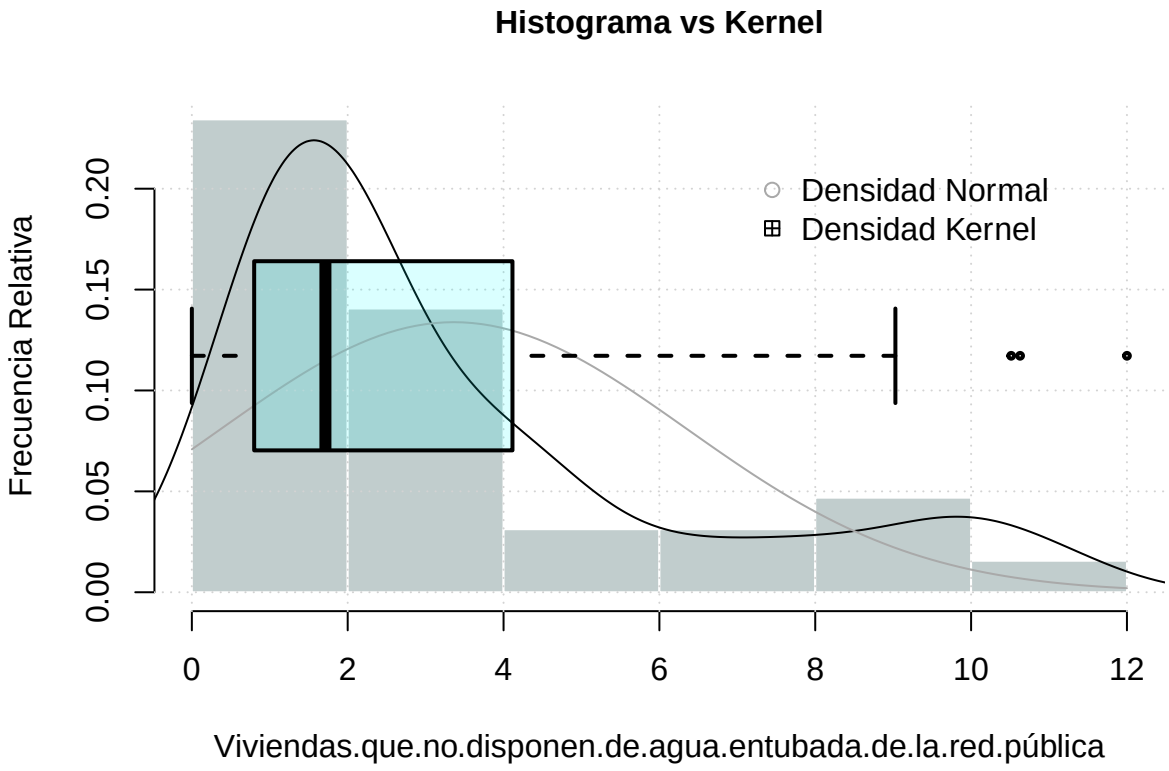

Histograma vs Kernel
Densidad Normal
Densidad Kernel
Frecuencia Relativa
0.00
0.10
0.20
0.30
4
6
8
10
Población.de.6.a.14.años.que.no.asiste.a.la.escuela

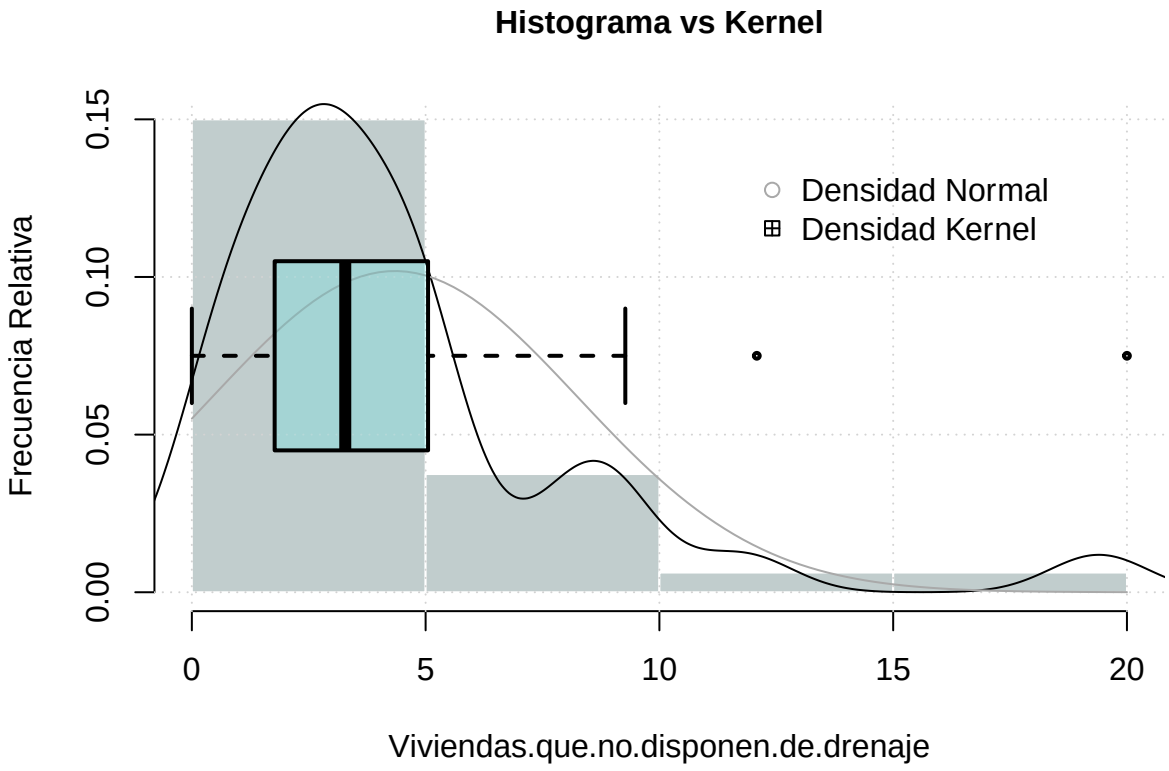

Histograma vs Kernel
Densidad Normal
Densidad Kernel
Frecuencia Relativa
0.00
0.02
0.04
0.06
0.08
15
20
25
30
35
40
45
50
Población.de.15.años.y.más.con.educación.básica.incompleta

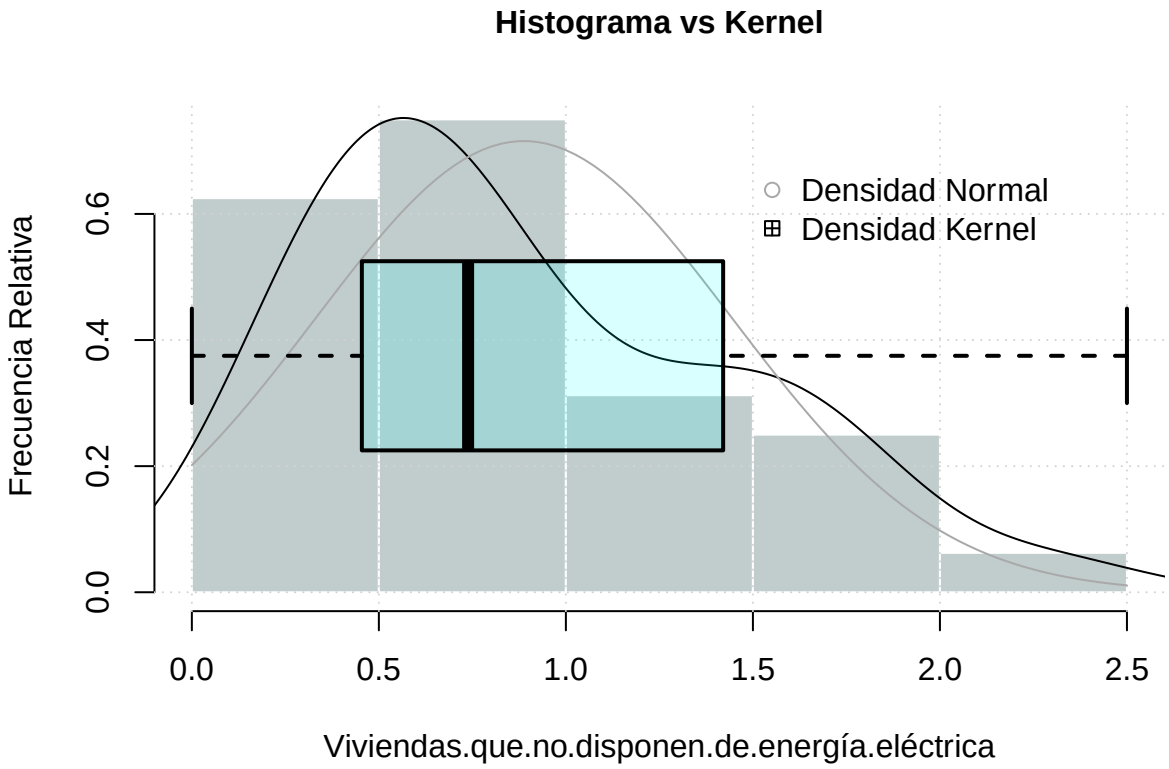

Histograma vs Kernel
Densidad Normal
Densidad Kernel
Frecuencia Relativa
0.00
0.02
0.04
0.06
15
20
25
30
35
40
Población.sin.derechohabiencia.a.servicios.de.salud

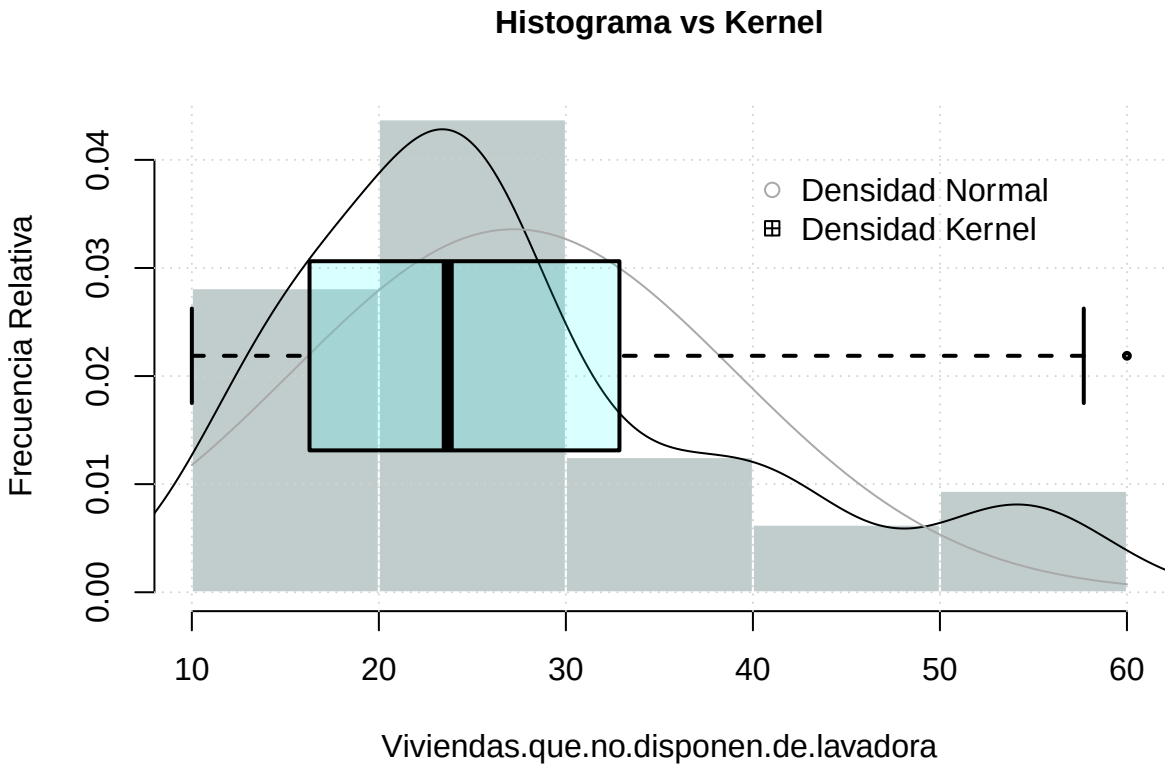

Histograma vs Kernel
Densidad Normal
Densidad Kernel
Frecuencia Relativa
0.00
0.05
0.10
0.15
0.20
0
2
4
6
8
10
12
14
Viviendas.con.piso.de.tierra

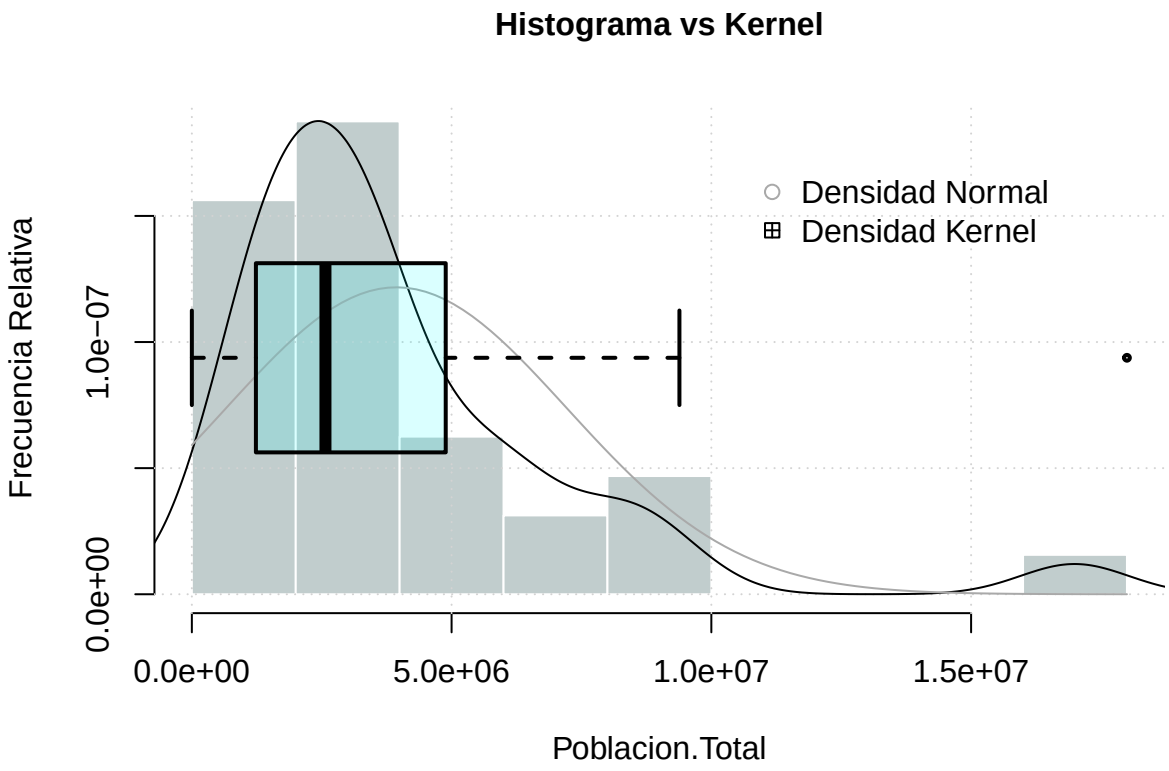

Histograma vs Kernel
Densidad Normal
Densidad Kernel
Frecuencia Relativa
0.00
0.10
0.20
0
2
4
6
8
10
Viviendas.que.no.disponen.de.excusado.o.sanitario


Histograma vs Kernel
Densidad Normal
Densidad Kernel
Frecuencia Relativa
0.00
0.05
0.10
0.15
0.20
0
2
4
6
8
10
12
Viviendas.que.no.disponen.de.agua.entubada.de.la.red.pública

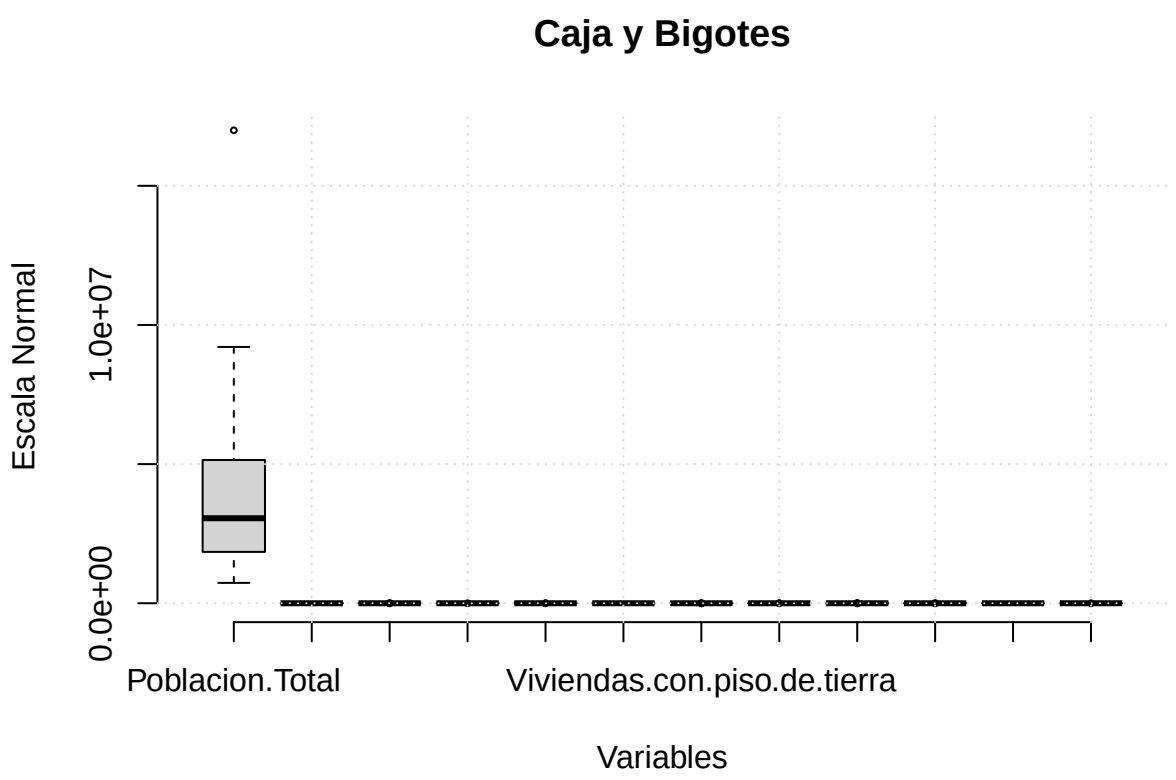

Histograma vs Kernel
Densidad Normal
Densidad Kernel
Frecuencia Relativa
0.00
0.05
0.10
0.15
0
5
10
15
20
Viviendas.que.no.disponen.de.drenaje


Histograma vs Kernel
Densidad Normal
Densidad Kernel
Frecuencia Relativa
0.0
0.2
0.4
0.6
0.0
0.5
1.0
1.5
2.0
2.5
Viviendas.que.no.disponen.de.energía.eléctrica


Histograma vs Kernel
Densidad Normal
Densidad Kernel
Frecuencia Relativa
0.00
0.01
0.02
0.03
0.04
10
20
30
40
50
60
Viviendas.que.no.disponen.de.lavadora




```
# Boxplot
boxplot(datos, main="Caja y Bigotes",
        frame = FALSE, xlab="Variables", ylab= "Escala Normal", cex=0.4);grid()
```

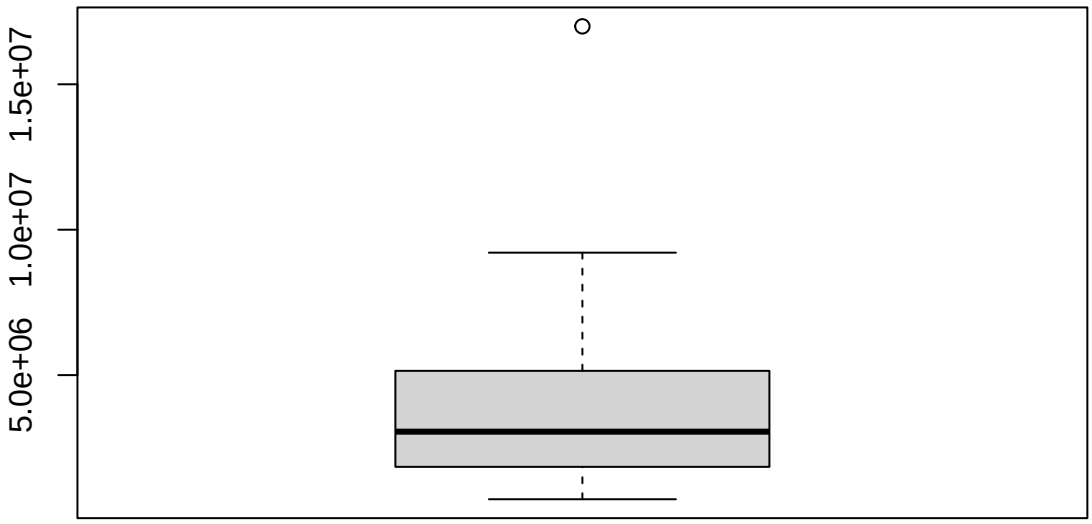



```
# Andrews ##CORREGIR!!!
#andrews(df = datos, type=2, bty = "n", ylab="f(t)", xlab="t",lwd=1, main="Grafico Andrews" ); grid()
```

Eliminacion de datos atipicos

- Importante ver que la variable auxiliar ayuda a identificar observaciones que afecten el análisis.

```
outliers <- boxplot(datos$Poblacion.Total)$out
```

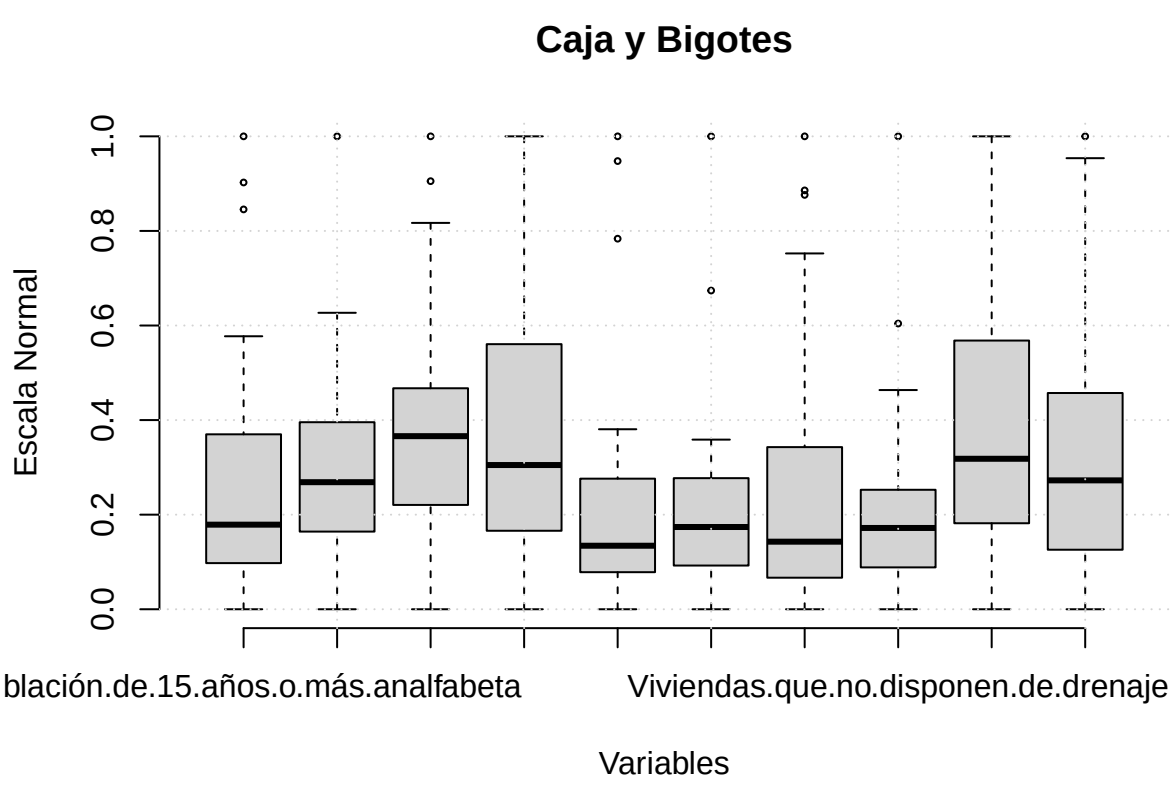



```
elementos <- which(datos$Poblacion.Total %in% outliers)

datos <- datos[-union(elementos,elementos), ]
```

Escalamiento

Iportante que los indices se recodifican a una escala ordinal, pero primero se normalizan ya que se trata de un indice.

```
# Normalizacion
datos[,analisis] <- sapply(datos[, analisis], function(data){
         (data - min(data)) / (max(data) - min(data))})
# Boxplot
boxplot(datos[, analisis], main="Caja y Bigotes",
        frame = FALSE, xlab="Variables", ylab= "Escala Normal", cex=0.4);grid()
```



1 Matriz de datos con escala ordinales

Definicion de rangos para la escala de lickert

0-20 -> 1 21:40 -> 2 40:60 -> 3 61:80 -> 4 81:100 -> 5

```
# Transformacion a escala ordinal
datos[, analisis] <- datos[, analisis]*100
datos[, analisis] <- round(datos[, analisis])

for(indice in analisis){
  for(n in 1:nrow(datos)){
    datos[n,indice] = recode(datos[n,indice], "0:20=1; 21:40=2; 41:60=3; 61:80=4; 81:100=5")
  }
}

# Formato Correcto
for(indice in analisis){
  datos[, indice] <- factor(datos[, indice], order = TRUE)
}

# Redefinicion de Escalas
tipo <- sapply(datos, class)
continuas <-  which(tipo == "numeric") # continuas
enteras <- which(tipo == "integer") # enteras
numericas <- names(c(continuas,enteras))
```

```r
# Variables Categoricas
nominales <- which( tipo == "factor") # categoricas
ordinales <- which( sapply(datos, is.ordered) )  # ordinales
fecha <- which(tipo == "Date") # Fecha
categoricas <- names(c(nominales, ordinales, fecha))
```

Calculo de la matriz de Disimilaridad

- Como las variables son en escala ordinal, ent se utiliza distancia gower(mixtas).

```r
gower_dist <- daisy(datos[, analisis], metric = "gower")
```

Escalamiento no métrico

- Métricas de ajuste: stress con valor entre [0,1] y entre mas pequeño mejor. Y rss; entre mas pequeño mejor. En este caso, a prueba y error se encontro que 7 es la mejor dimensión.

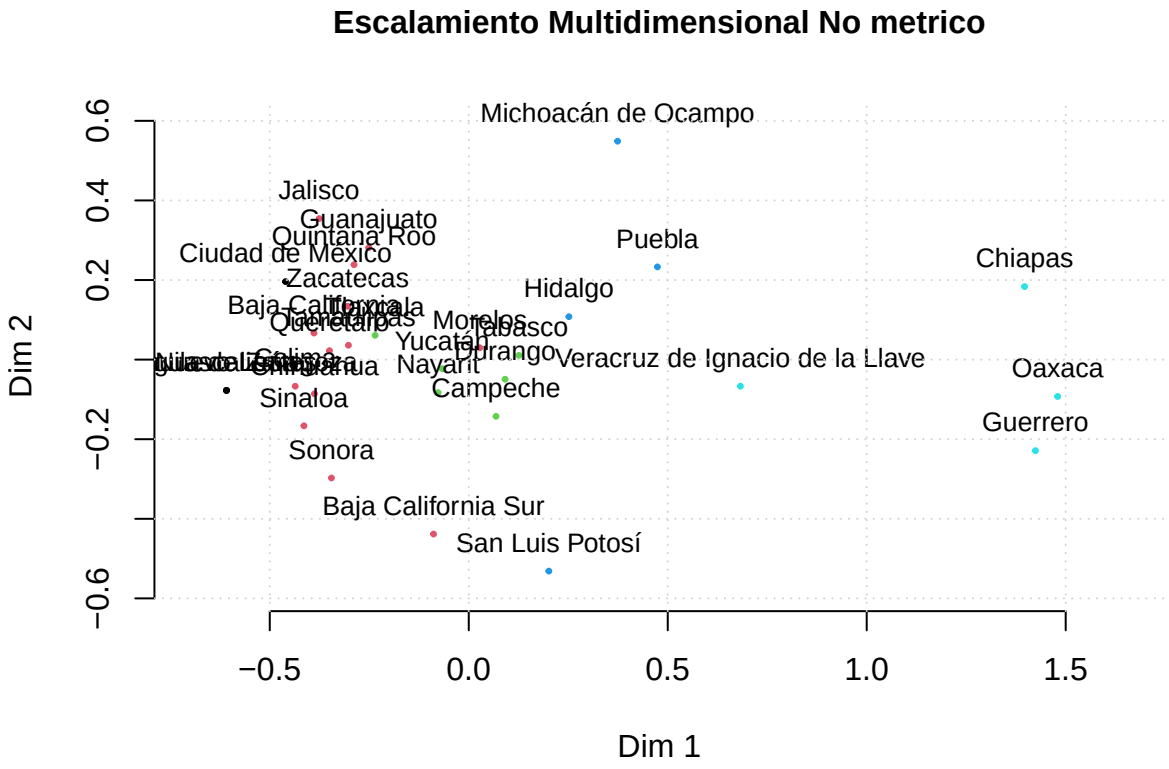
```r
fit.datos <- smacofSym(gower_dist, type = "ordinal", ndim = 7)
fit.datos$stress
```

```
## [1] 0.019318
```

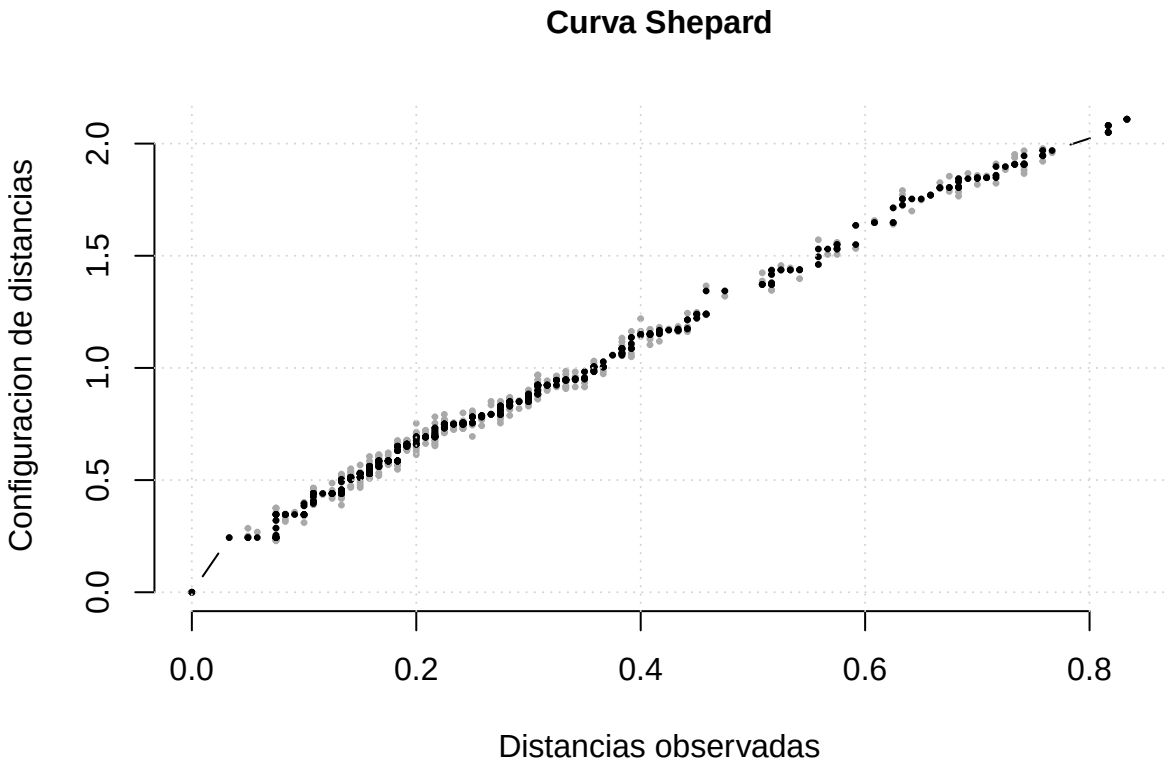
```r
fit.datos$rss
```

```
## [1] 0.1735311
```

```r
# Dispersion
plot(fit.datos, plot.dim = c(1,2), main = "Escalamiento Multidimensional No metrico",
     xlab="Dim 1", ylab="Dim 2", cex=0.5, cex.main=1,
      bty = "n",  col = datos$Grado.de.rezago.social );grid()
```




```
# Curva Shape
plot(fit.datos, plot.type = "Shepard", main="Curva Shepard",
     xlab="Distancias observadas", ylab="Configuracion de distancias", cex=0.5, cex.main=1,
     col="skyblue", bty = "n");grid()
```



Mejoras

Para mejorar el ajuste, se puede intentar los siguiente:

1. Incrementar el numero de dimensiones(Capturar mayor variabilidad que implica menor rss)
2. Usar otra medida de disimilaridad
3. Usar otro algoritmo de optimizacion para el escalamiento
4. Problemas de preprocesamiento
5. Usar otro metodo como t-sne

Implementación de t-sne

```
from sklearn.manifold import TSNE
import seaborn as sns
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
```

Implementacion

```
# Datos
datos = r.datos
```

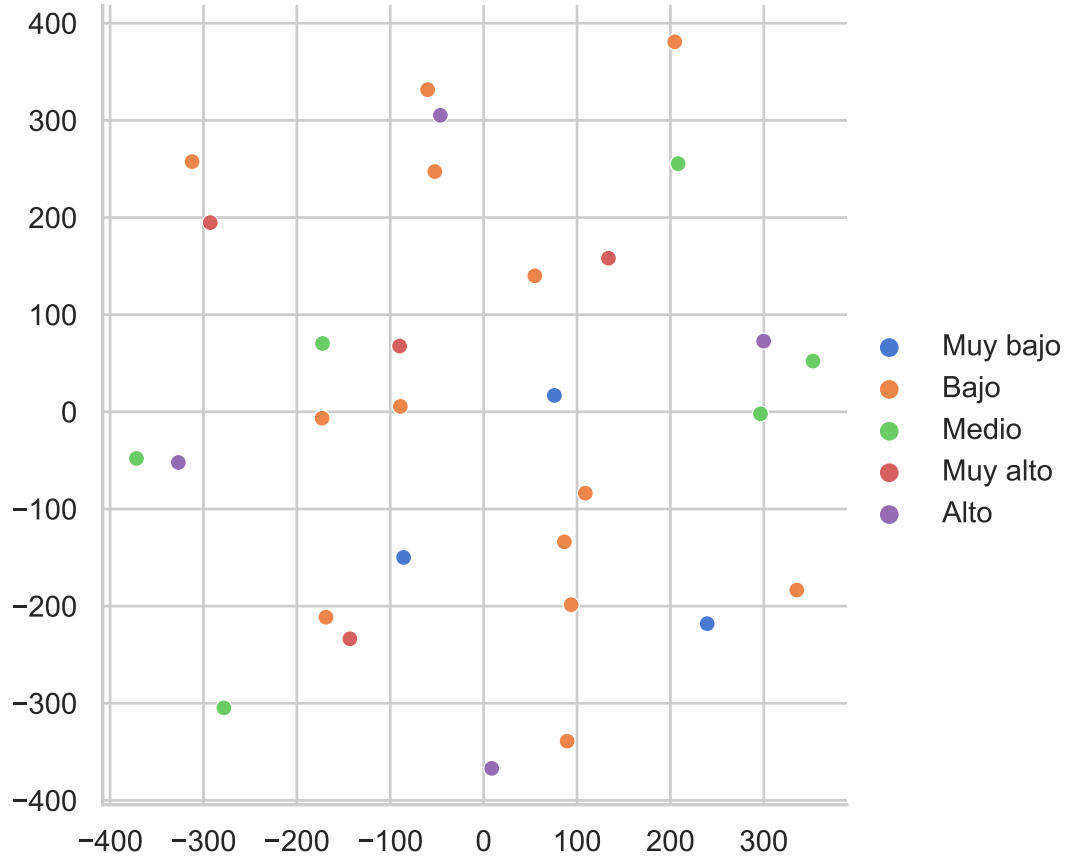
```
# Particion horizontal
x = np.array(datos[r.analisis])
y = np.array(datos[r.auxiliares[1]]) # Variable suplementaria
```

Ajuste

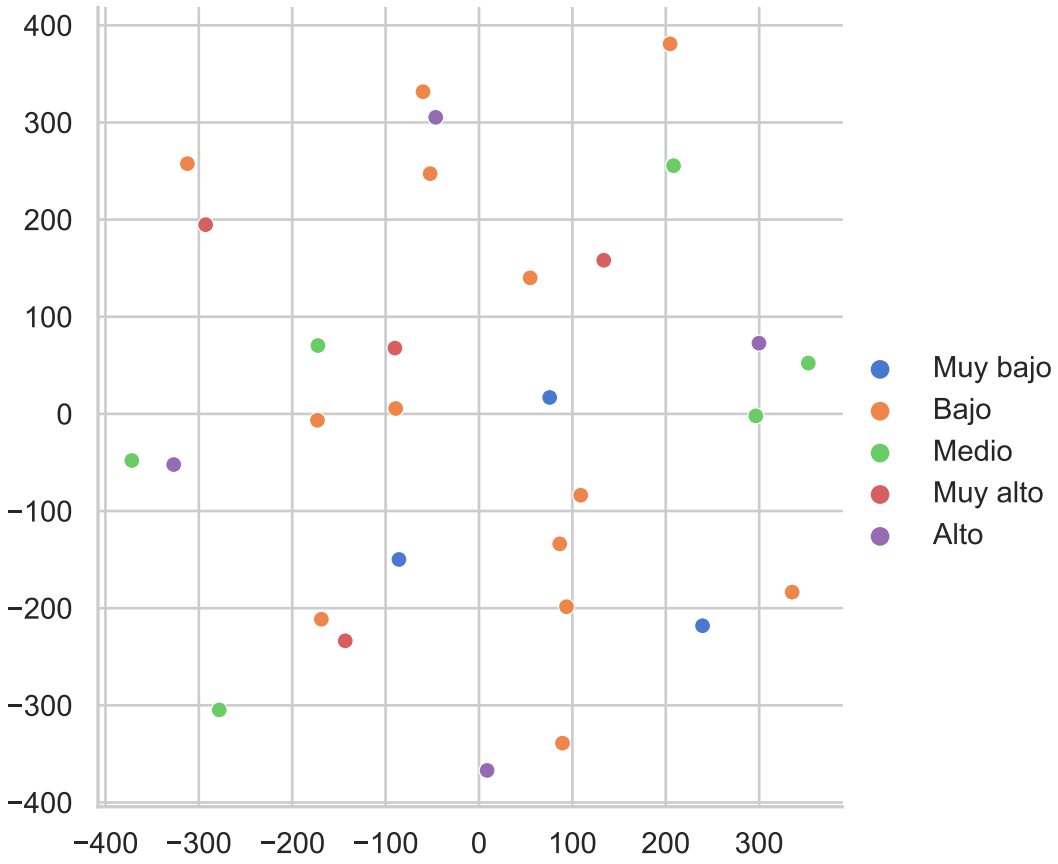
```
x_coord = TSNE(n_components = 3, perplexity = 30, n_iter = 4000).fit_transform(x)
```

Grafico

```
plt.clf()
sns.set(style="whitegrid")
sns.relplot(x=x_coord[:,0], y=x_coord[:,1], hue=y, palette="muted" )
```




```
plt.show()
```




```
exit
```

```
## Use exit() or Ctrl-Z plus Return to exit
```