Tarea final

Computación y programación en R

Juan Cantero Jimenez

13 de diciembre, 2021

Ejercicio 1

Se ha creado una función que toma como argumentos, el banco de datos, el nombre como caracter de una de las variables del dataset, el tipo de motor gráfico que se desea usar, ggplot2 o R base y el tipo de gráfico que se desea. Por defecto la función realiza un gráfico de correlación con R base, pero tambien puede ser indicado con le tipo "correlacion". Otra opción es indicar "VS" en el argumento tipo y una variable numérica, para lo cual la función crea un plot de cajas y bigotes con la variable numérica agrupada en función de todas las variables discretas del dataset. Por último se puede indicar "unico" y una variable para lo cual la función creará un histograma para las funciones continuas o un gráfico de barras para las discretas.

```
n = 100
data <- data.frame(list(A=rnorm(n),
                        B=as.factor(sample(c("enfermo","sano","nulo"),n,
                                           replace=TRUE,prob=c(0.5,0.25,0.25))),
                        C=as.factor(sample(c("bueno","feo","malo"),n,
                                           replace=TRUE,prob=c(0.5,0.25,0.25))),
                        D=rgamma(n,1,1),
                        E=rgamma(n,1,1),
                        H=as.factor(sample(c("enfermo","sano","nulo"),n,
                                             replace=TRUE,prob=c(0.5,0.25,0.25)))))
# Se define un dataframe con 100 observaciones,
# 3 variables numéricas continuas y 2 variables discretas.
graphics_wrap <- function(data,
                          tipo=c("correlacion","unico","VS"),
                          variable=NULL,
                          gplot=FALSE){#Se crea una función que genera plots
  #de correlación, plot de una sola variable, así como agrupaciones de una
  #variable numérica con todas la categóricas. Permitiendo además seleccionar
  #un motor gráfico.
  require(ggplot2)
  correlation <- function(data){ #Se define la función que creará un plot de
   #correlación con todas la variables numéricas presentes en data. Esta función
    #hace uso del paquete gráfico básico de R.
    tipos <- sapply(data, function(x){
      is.numeric(x)
  })#Se itera sobre las columnas del data.frame, comprobando si esta es númerica
    # el resultado es un vector de tipo lógico.

    numeric_names <- names(data)[tipos] # se seleccionan los nombres de las
    #  columnas que poseen datos de tipo numérico
    if (length(numeric_names )<= 5){ # se comprueba que el número de variables
    #sea menor que 5. En este caso se realizará un plot de correlación mediante
```

```
     #un panel en el que se representan, mediante scatter plot, todas las
      #convinaciones de variables numéricas de tamaño 2.
par(mfrow=c(length(numeric_names),length(numeric_names)))#se ajusta la distri_
      #bución de los plots.
      for (x in numeric_names){
        for (y in numeric_names){

      plot(data[[x]],data[[y]],xlab="",ylab="")#Se crean los plots que enfrentan
          # a las variables.
          title(main=paste(x,"VS",y),
                sub = paste("correlacion =",
                            cor(data[[x]],
                                data[[y]])),
                xlab=x, ylab=y)#se añade el título, el titulo de los ejes, así
          # como un subtitulo que indique el valor del indice de correlación.
        }
      }# Se iterá dos veces sobre el vector numeric_names.
    }else{
      #stop("Too much to plot")
    matt <- matrix(0, nrow=length(numeric_names),ncol=length(numeric_names))#se
      #crea la matriz que alvergará los coeficientes de correlación
      colnames(matt) <- numeric_names#Se da nombre a las columnas
      rownames(matt) <- numeric_names# Se da nombre a las filas
      for (x in numeric_names){
        for (y in numeric_names){
          matt[x, y] <- cor(data[[x]],data[[y]])
        }
      }#Se iterá sobre el vector numeric_names para calcular los coeficientes de
      #correlación así como para rellenar matt
      par(mfrow=c(1,1))# Se ajustan el número de plots por ventana
      #matt[nrow(matt):1,]
      colfunc <- colorRampPalette(c("blue","white","red"))#Se crea la funcíon
    #colfunc que devuelve un degradado de colores desde el azula al rojo pasando
    #por el blanco, en un número de pasos igual al valor de su único argumento.
      #Vease uso más adelante.
    matt[upper.tri(matt)] <- 0 #Se elimina la información repetida de la matriz
    layout(matrix(1:2,ncol=2), width = c(2,1),height = c(1,1))#Se crea un layout
      #que permita representar la matriz de puntos así como la leyenda.
      image( matt, xaxt= "n", yaxt= "n",col = colfunc(20) )#Se representa la
      #matriz matt con los valores de correlación, coloreados en función de
      #colfunc
      axis( 1,
            at=seq(0,1,length.out=ncol( matt ) ),
            labels= colnames( matt ), las= 2 )
      axis( 2,
            at=seq(0,1,length.out=nrow( matt ) ),
            labels= rownames( matt ), las= 2)#Se añaden ambos ejes a la imagen
      #crada por la función image

      legend_image <- as.raster(matrix(colfunc(20), ncol=1))#Se crea la imagen
      #que indica el valor numérico aproximado de la coloración escogida.
```

```
    plot(c(0,2),c(0,1),type = 'n',
         axes = F,xlab = '', ylab = '', main = 'Correlación')#se crea un plot
    #vacio en el que se situará la leyenda
  text(x=1.5, y = seq(0,1,l=5), labels = seq(-1,1,l=5)[5:1])#Se añade el texto
    #de esta.
    rasterImage(legend_image, 0, 0, 1,1)#Se generá la imagen en el plot vacio
    #crado anteriormente.
  }# En caso de que existan demasiadas columnas a representar, el plot de
  #correlación se realiza como una matriz, coloreada en función del indice
  # de correlación.
  #return(matt)
}

numeric_categoric <- function(data,variable){ #Se crea una función que
  #represente la variable numerica indicada en sus argumentos, en un boxplot
#con los datos agrupados en función de todas la variables categóricas presentes
  #en el dataset
  tipos <- sapply(data, function(x){
    is.factor(x)
  })#Se obtiene la posición de las columnas de tipo factor en el data.frame

  numeric_names <- names(data)[tipos]#Se seleccionan los nombres de las
  #variables de tipo factor.
  n_plots <- length(numeric_names)#Se obtiene la longitud del vector
  #numeric names
  if (n_plots <= 2){
    par(mfrow=c(1, n_plots))
  }else{
    par(mfrow=c(round(sqrt(n_plots)),round(sqrt(n_plots))))
  }#Se crea un layout en función del número de plots en la imagen, si este
#es inferior a 2, se representan en una única fila, mientras que si es mas que
  #2 se representa en una cuadrícula.
  for (grp in numeric_names){



    boxplot(data[[variable]] ~ data[[grp]],xlab="", ylab="")
    title(main = paste(variable, "vs", grp), ylab=variable, xlab=grp)#Se crean
    # los boxplot y se añade título a estos.
  }#Se iterá sobre numeric_names para generar un plot de la variable, agrupada
  #en función de las variables presentes en numeric_names.
  if (length(numeric_names) < prod(c(round(sqrt(n_plots)),round(sqrt(n_plots))))){
  n <-   prod(c(round(sqrt(n_plots)),round(sqrt(n_plots)))) -length(numeric_names)
  for (i in n){
    plot(c(1,1),c(1,1),type = "n")
  }
  }
}

var_description <- function(data, variable, ggplot){#Se crea una función que
  #genera un plot de la variable indicada del dataset aportado, permitiendo
```

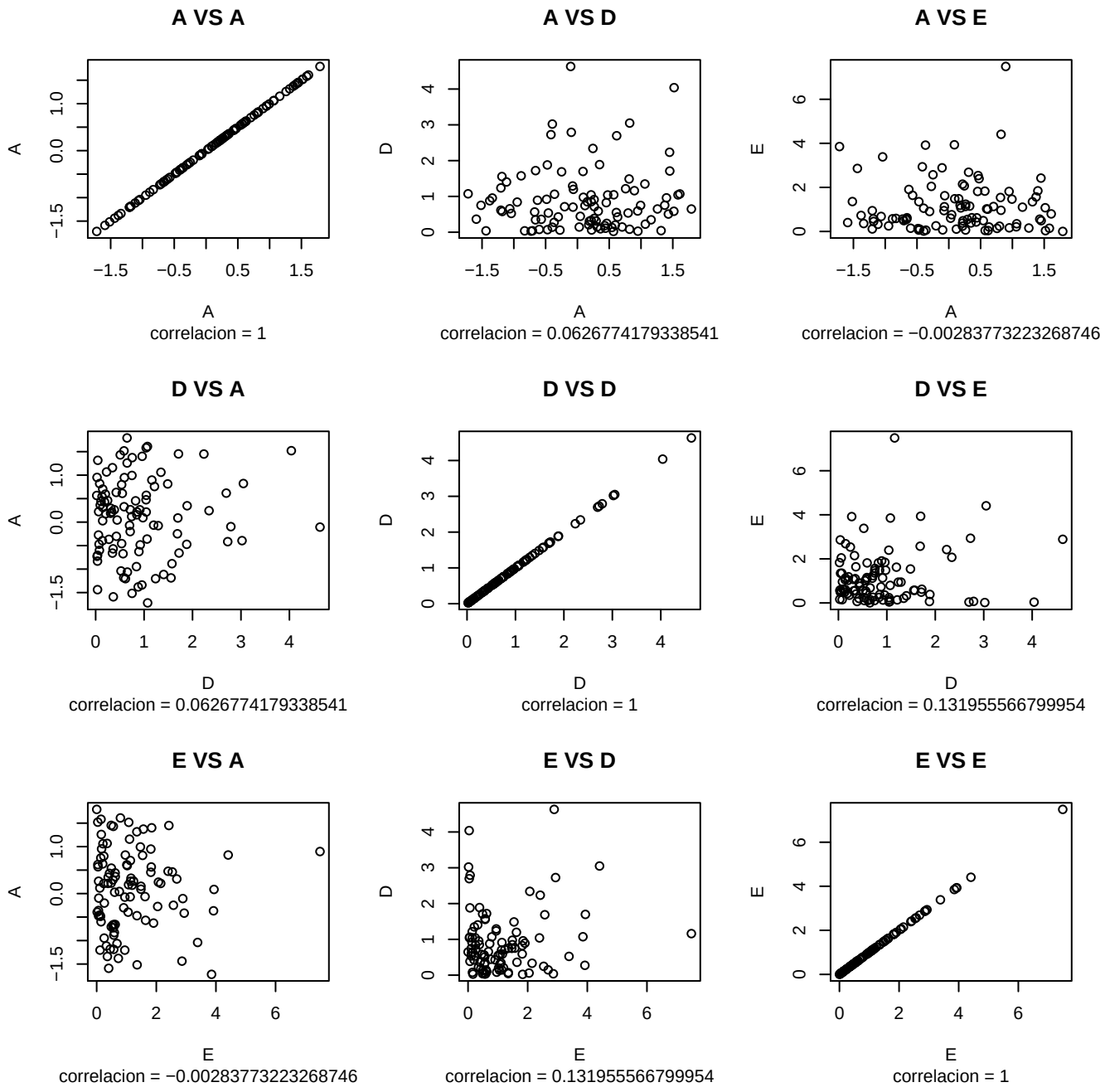
```
    #escoger el tipo de motor gráfico que se desea usar.
    if (ggplot){
      require(ggplot2)
    }#Se carga el paquete ggplot2 en el caso que sea necesario
    par(mfrow=c(1,1))#Se ajusta la disposición de los plots en la pantalla
    if(is.numeric(data[[variable]]) & ggplot == FALSE){
      hist(data[[variable]], xlab="",main="")
      title(main = paste("Histograma de",variable), xlab=variable)#Se crea un
      #histograma de la variables indicada con R base, se añade el título
      #principal y el de el eje x
      print(summary(data[[variable]]))#Se añade un sumario estadístico de la
      #variable

    }#Se define una subrutina para cuando la variables es número ica y se quiere
    #graficar con R base
    if(is.factor(data[[variable]]) & ggplot ==FALSE){
      barplot(table(data[[variable]]), xlab="")
      title(main = paste("Barplot de",variable), xlab=variable)#Se crea un
      #gráfico de barras de la variable discreta deseada, se añade el título
      #principal y el de la variable x.
      print(table(data[[variable]])/nrow(data))#Se muestrán las frecuencias
      #relativas de la variables a graficar.
    }#Se define una subrutina para cuando la variables es un factor y se quiere
    #graficar con R base.
    if(is.numeric(data[[variable]]) & ggplot == TRUE){
      j <-ggplot(data, aes_string(x=variable)) + geom_histogram() #Se crea un
      #histograma haciendo uso de ggplot2
      print(j)#Se imprime la figura generada
      print(summary(data[[variable]]))#Se añade un sumario estadístico de
      #la variable
    }#Se define una subrutina para cuando la variables es numerica y se quiere
    #graficar con ggplot.
    if(is.factor(data[[variable]]) & ggplot == TRUE){
      gather <- data.frame(categori = names(table(data[[variable]])),
                           dat = as.vector(table(data[[variable]])))#se crea un
      #dataframe adicional que facilitará la representación gráfica con ggplot2

      j <- ggplot(data=gather, aes(x=categori, y=dat)) +
      geom_bar(stat = "identity") # Se crea un grafico de barras de la variable
      # a graficar con el paquete ggplot2
      print(j)#Se imprime la figura creada
      print(table(data[[variable]])/nrow(data))#Se muestrán las frecuencias
      #relativas de la variables a graficar.
    }#Se define una subrutina para cuando la variables es de tipo factor y
    #se quiere graficar con ggplot.
  }

correlation_ggplot <- function(data){ # Se crea una función que genera gráficos
    #de correlación para la variables numéricas en el argumento data, usando
    #el motor gráfico ggplot2
    tipos <- sapply(data, function(x){
```

```
    is.numeric(x)
  })#Se obtiene la posición de las columnas de tipo numerico en el data.frame

  numeric_names <- sort(names(data)[tipos])#Se seleccionan los nombres de las
  #variables de tipo factor. Notese que se han ordenado los nombres, esto es
  #necesario debido a que ggplot2 ordenará posteriormente las variables a
  #representar.
  data_new <- data[,numeric_names] #Se crea un data.frame adicional que
  #facilitará la representación con ggplot2
  correlation_mat <- round(cor(data_new),2)#Se crea la matriz de correlación
  #en este caso con la función cor

  correlation_mat[upper.tri(correlation_mat)] <- NA #Se elimina la información
  # de la diagonal superior, pues esta repetida
  correlation_mat <- t(correlation_mat) # Por conveniencia se transpone la
  #matriz



  melt_corr <- data.frame(list(Var1=rep(numeric_names, length(numeric_names)),
                               Var2=rep(numeric_names,
                                        rep(length(numeric_names),
                                            length(numeric_names))),
                               value=rep(NA, length(numeric_names)^2)))
  #Se crea un data.frame auxiliar con toda la informacióna representar. Este
#se encuentra vacio y se rellenará con el bucle for que se encuentra más abajo.
  for (x in 1:nrow(melt_corr)){
    if(!is.na(correlation_mat[melt_corr[x, 2], melt_corr[x,1]])){
      melt_corr[x, 3] <- correlation_mat[melt_corr[x, 2], melt_corr[x,1]]
    }
  }#Se rellena el campo value del dataframe creado con la información presente
  #en correlation_mat gracias a la información aportada por las variables
  #Var1 y Var2

  melt_corr$Var1 <- as.factor(melt_corr$Var1)
  melt_corr$Var2 <- as.factor(melt_corr$Var2)
  melt_corr$value <- as.numeric(melt_corr$value)#Se convierten las variables
  #al tipo neceario
  melted_cormat <- melt_corr[!is.na(melt_corr$value),]#Se eliminan los NA

h <- ggplot(data = melted_cormat, aes(Var1, Var2, fill = value))+ #Se definen
    #el data.set de donde saldrán los datos, así como que hacer con cada
    #variable
    geom_tile() + #Se genera un geom de tipo raster, una imagen.
    scale_fill_gradient2(low = "blue", high = "red", mid = "white",
                         midpoint = 0, limit = c(-1,1), space = "Lab",
                         name="Pearson\nCorrelation") + #Se crea una escala
    #de color para usar en la imagen creada con geom_raster.
    theme_minimal()+
    theme(axis.text.x = element_text(angle = 45, vjust = 1,
                                     size = 12, hjust = 1))+#Se define como se
```

```
      #representará la etiqueta del eje x.
      coord_fixed()#Se fijan las coordenadas del plot creado.
    j <-  h +
      geom_text(aes(Var1, Var2, label = value), color = "black", size = 4) +
      theme(
        axis.title.x = element_blank(),
        axis.title.y = element_blank(),
        panel.grid.major = element_blank(),
        panel.border = element_blank(),
        panel.background = element_blank(),
        axis.ticks = element_blank(),
        legend.justification = c(1, 0),
        legend.position = c(0.6, 0.7),
        legend.direction = "horizontal")+
      guides(fill = guide_colorbar(barwidth = 7, barheight = 1,
                                   title.position = "top", title.hjust = 0.5))
  #Se añaden los valores del coeficiente de correlación en la posición adecuada.
    print(j)#Se imprime el gráfico creado.

}

numeric_categoric_ggplot <- function(data, variable){ #Se crea una función que
  #represente la variable numerica indicada en sus argumentos, en un boxplot
  #con los datos agrupados en función de todas la variables categóricas presentes
  #en el dataset
  require(ggplot2)#Se carga ggplot en caso de que no lo este
  tipos <- sapply(data, function(x){
    is.factor(x)
  })#Se obtiene la posición de las columnas de tipo factor en el data.frame

  factor_names <- names(data)[tipos]#Se seleccionan los nombres de las variables
  #de tipo factor a representar.
  data_variables <- rep(factor_names, rep(nrow(data), length(factor_names)))
  #Se genera una columna con el nombre de cada variable categórica repetida
  #tantas veces como observaciones tiene el dataset.
  data_numeric <- rep(data[[variable]], length(factor_names)) #Se crea una
  #columna con la variable numérica repetida tantas veces como variables
  #catefóricas tenga el dataset.
  data_factors <- NULL
  for ( x in factor_names){

    data_factors <- c(data_factors, as.character(unlist(data[x])))
  }#Se rellena data_factors con el contenido de las columnas categóricas en el,
  #mismo orden que data_variables.

  gather <- data.frame(list(variables=as.factor(data_variables),
                            response=as.numeric(data_numeric),
                            grupos=as.factor(data_factors)))#Se agrupan las
  #columnas creadas antes, imponiendoles además el tipo de dato que deben ser.
  p <- ggplot(gather)+
    aes(x=grupos, y=response)+
```

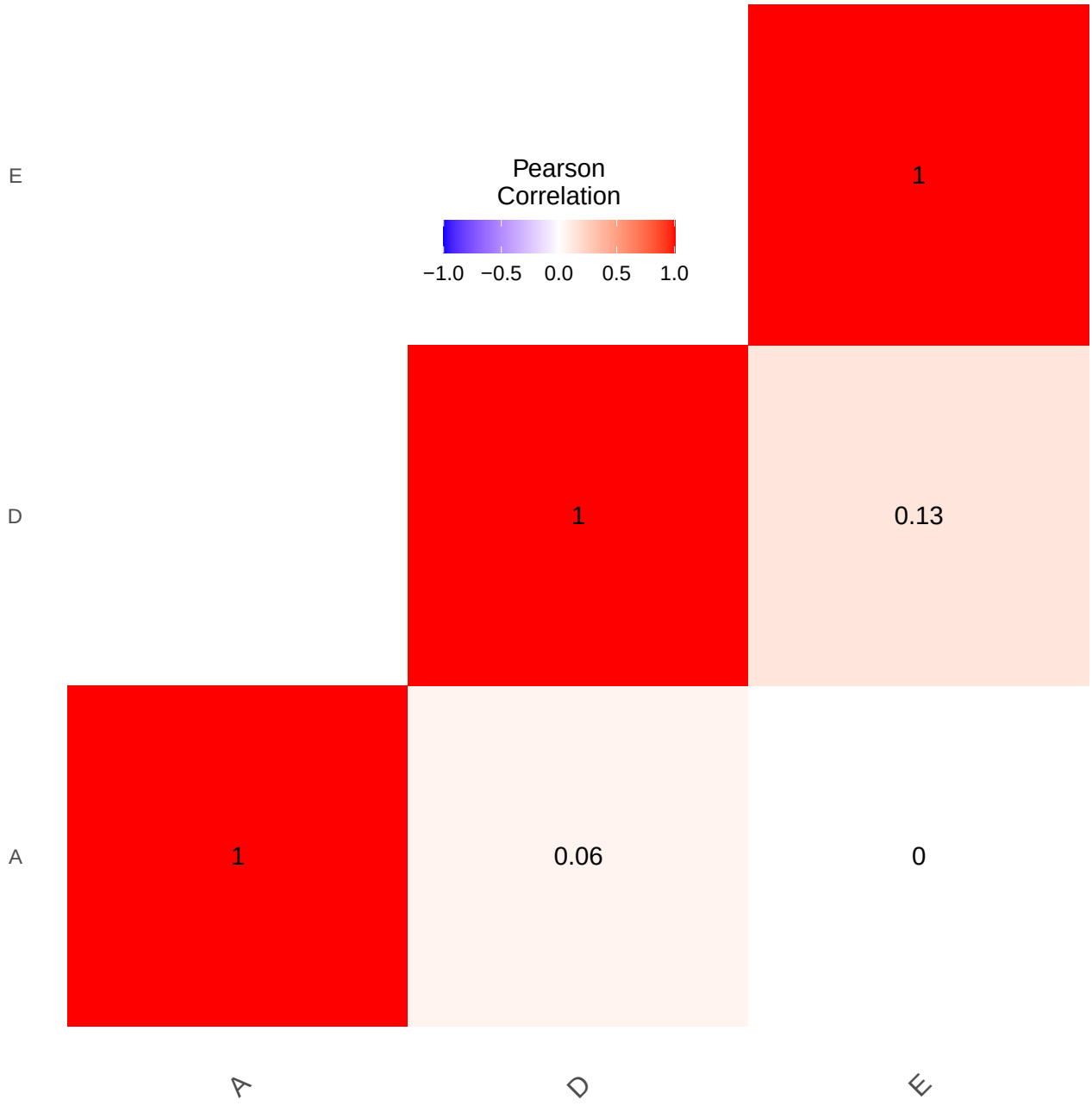
```
    geom_boxplot() +
    facet_grid(. ~ variables,space="free",scale="free") #Se crea un plot de tipo
  #boxplot en función de los grupos de las variables. Estas variables además
  #serán usados para definir el layout del plot.
  print(p)

}

if (length(tipo) > 1){tipo ="correlacion"}#Si no se indica un valor, por defecto
#realizará plot de tipo correlación.

if (tipo == "correlacion"){
  if (gplot == TRUE){
    correlation_ggplot(data)
    tipos <- sapply(data, function(x){
      is.numeric(x)
    })#Se itera sobre las columnas del data.frame, comprobando si esta es númerica
    # el resultado es un vector de tipo lógico.

    numeric_names <- names(data)[tipos] #Se obtienen los nombres de las variables
    #numéricas
    for (x in numeric_names){
      print(x)
      print(c(summary(data[[x]]), varianza=var(data[[x]])))
    }

  }else{
    correlation(data)
    tipos <- sapply(data, function(x){
      is.numeric(x)
    })#Se itera sobre las columnas del data.frame, comprobando si esta es númerica
    # el resultado es un vector de tipo lógico.

    numeric_names <- names(data)[tipos] #Se obtienen los nombres de las variables
    #numéricas
    for (x in numeric_names){
      print(x)
      print(c(summary(data[[x]]), varianza=var(data[[x]])))
    }

  }
}#Se crea una condición que ejecuta el programa en caso de que el tipo sea
#correlación
if(tipo == "unico"){
  var_description(data, variable,ggplot = gplot)
}#Se crea una condición que ejecuta el programa en caso de que el tipo sea
#unico
if(tipo == "VS"){
  if (gplot == TRUE){
    numeric_categoric_ggplot(data, variable)
    tipos <- sapply(data, function(x){
```

```
      is.factor(x)
    })#Se obtiene la posición de las columnas de tipo factor en el data.frame

    factor_names <- names(data)[tipos]#Se seleccionan los nombres de las variables
    #de tipo factor a representar.
    print(variable)
    print(c(summary(data[[variable]]),varianza=var(data[[variable]])))
    for (x in factor_names){
      print(x)
      print(table(data[[x]])/length(data[[x]]))
    }
  }else{
    numeric_categoric(data, variable)
    tipos <- sapply(data, function(x){
      is.factor(x)
    })#Se obtiene la posición de las columnas de tipo factor en el data.frame

    factor_names <- names(data)[tipos]#Se seleccionan los nombres de las variables
    #de tipo factor a representar.
    print(variable)
    print(c(summary(data[[variable]]),varianza=var(data[[variable]])))
    for (x in factor_names){
      print(x)
      print(table(data[[x]])/length(data[[x]]))
    }
  }
}#Se crea una condición que ejecuta el programa en caso de que el tipo sea
# VS


}




graphics_wrap(data,gplot = FALSE)

## Loading required package:  ggplot2
```

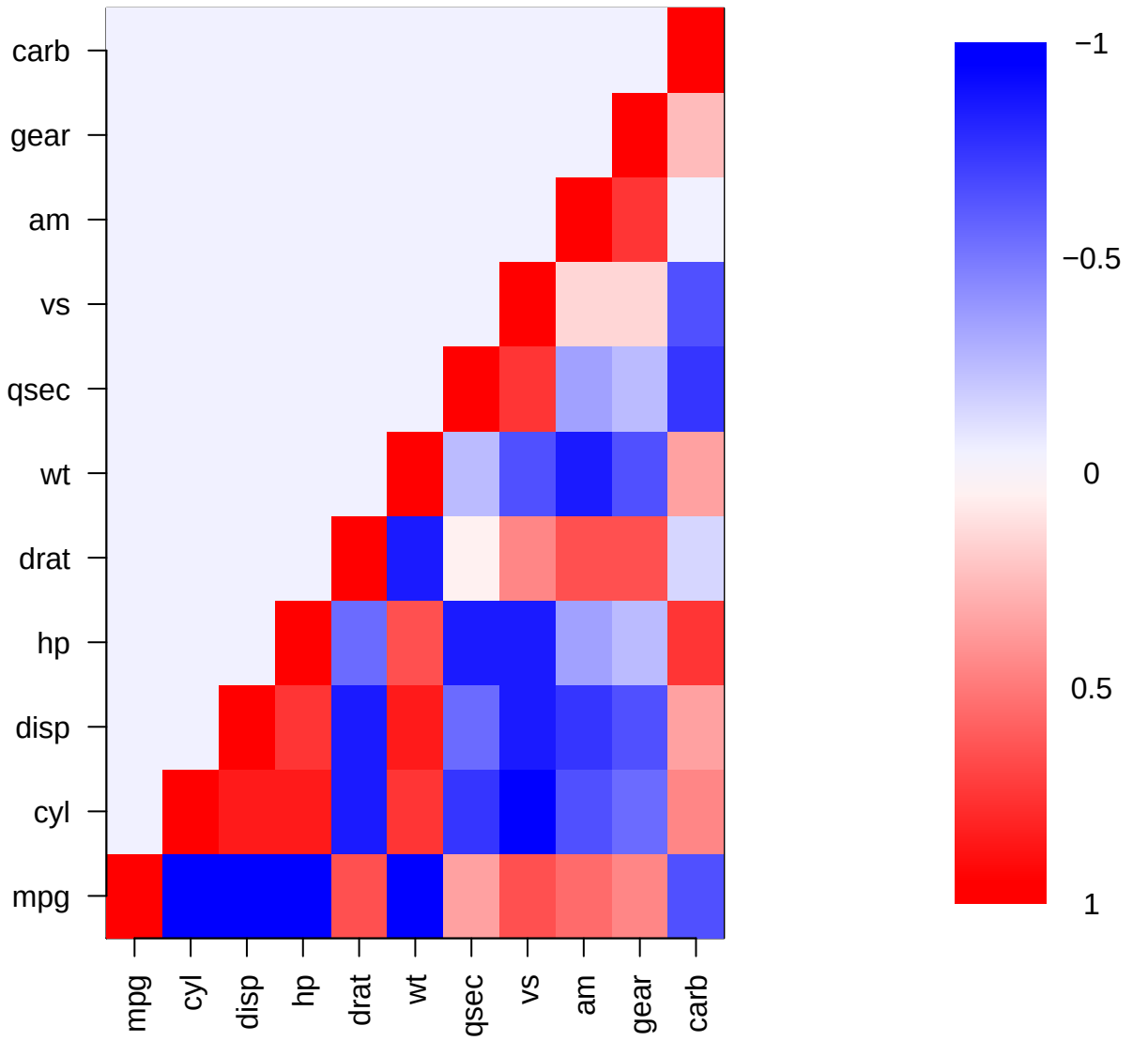



```
## [1] "A"
##       Min.    1st Qu.     Median       Mean    3rd Qu.       Max.   varianza
## -1.7192439 -0.4758483  0.2021251  0.1022933  0.6503286  1.7921763  0.7198792
## [1] "D"
##       Min.    1st Qu.     Median       Mean    3rd Qu.       Max.   varianza
## 0.02239897 0.30048623 0.67788113 0.88993645 1.09483711 4.63095774 0.74921441
## [1] "E"
##        Min.     1st Qu.      Median        Mean     3rd Qu.        Max.
## 0.002009988 0.343749416 0.743764105 1.136917479 1.547369583 7.489322058
##    varianza
## 1.414231954
```

```
graphics_wrap(data,gplot = TRUE)
```



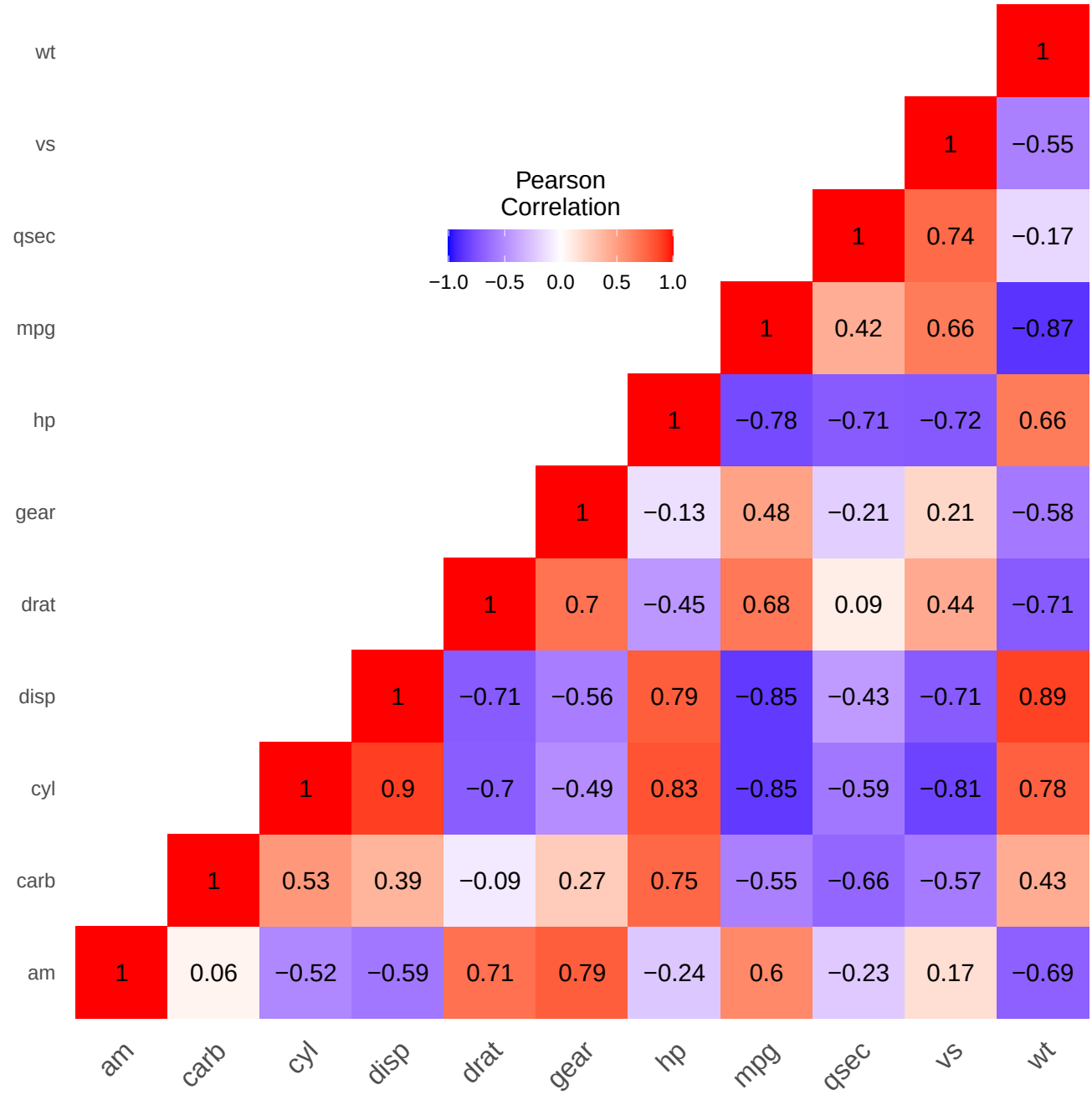

```
## [1] "A"
##       Min.    1st Qu.     Median       Mean    3rd Qu.       Max.   varianza
## -1.7192439 -0.4758483  0.2021251  0.1022933  0.6503286  1.7921763  0.7198792
## [1] "D"
##       Min.    1st Qu.     Median       Mean    3rd Qu.       Max.   varianza
## 0.02239897 0.30048623 0.67788113 0.88993645 1.09483711 4.63095774 0.74921441
## [1] "E"
##        Min.     1st Qu.      Median        Mean     3rd Qu.        Max.
## 0.002009988 0.343749416 0.743764105 1.136917479 1.547369583 7.489322058
##    varianza
## 1.414231954

graphics_wrap(mtcars, gplot=FALSE )
```




```
## [1] "mpg"
##     Min.  1st Qu.   Median     Mean  3rd Qu.     Max. varianza
## 10.40000 15.42500 19.20000 20.09062 22.80000 33.90000 36.32410
## [1] "cyl"
##     Min.  1st Qu.   Median     Mean  3rd Qu.     Max. varianza
## 4.000000 4.000000 6.000000 6.187500 8.000000 8.000000 3.189516
## [1] "disp"
##       Min.    1st Qu.     Median       Mean    3rd Qu.       Max.   varianza
##    71.1000   120.8250   196.3000   230.7219   326.0000   472.0000 15360.7998
## [1] "hp"
##      Min.   1st Qu.    Median      Mean   3rd Qu.      Max.  varianza
##   52.0000   96.5000  123.0000  146.6875  180.0000  335.0000 4700.8669
## [1] "drat"
```

```
##      Min.   1st Qu.    Median      Mean   3rd Qu.      Max.  varianza
## 2.7600000 3.0800000 3.6950000 3.5965625 3.9200000 4.9300000 0.2858814
## [1] "wt"
##     Min.  1st Qu.   Median     Mean  3rd Qu.     Max. varianza
## 1.513000 2.581250 3.325000 3.217250 3.610000 5.424000 0.957379
## [1] "qsec"
##      Min.   1st Qu.    Median      Mean   3rd Qu.      Max.  varianza
## 14.500000 16.892500 17.710000 17.848750 18.900000 22.900000  3.193166
## [1] "vs"
##      Min.   1st Qu.    Median      Mean   3rd Qu.      Max.  varianza
## 0.0000000 0.0000000 0.0000000 0.4375000 1.0000000 1.0000000 0.2540323
## [1] "am"
##      Min.   1st Qu.    Median      Mean   3rd Qu.      Max.  varianza
## 0.0000000 0.0000000 0.0000000 0.4062500 1.0000000 1.0000000 0.2489919
## [1] "gear"
##      Min.   1st Qu.    Median      Mean   3rd Qu.      Max.  varianza
## 3.0000000 3.0000000 4.0000000 3.6875000 4.0000000 5.0000000 0.5443548
## [1] "carb"
##     Min.  1st Qu.   Median     Mean  3rd Qu.     Max. varianza
## 1.000000 2.000000 2.000000 2.812500 4.000000 8.000000 2.608871
```

```
graphics_wrap(mtcars, gplot=TRUE)
```

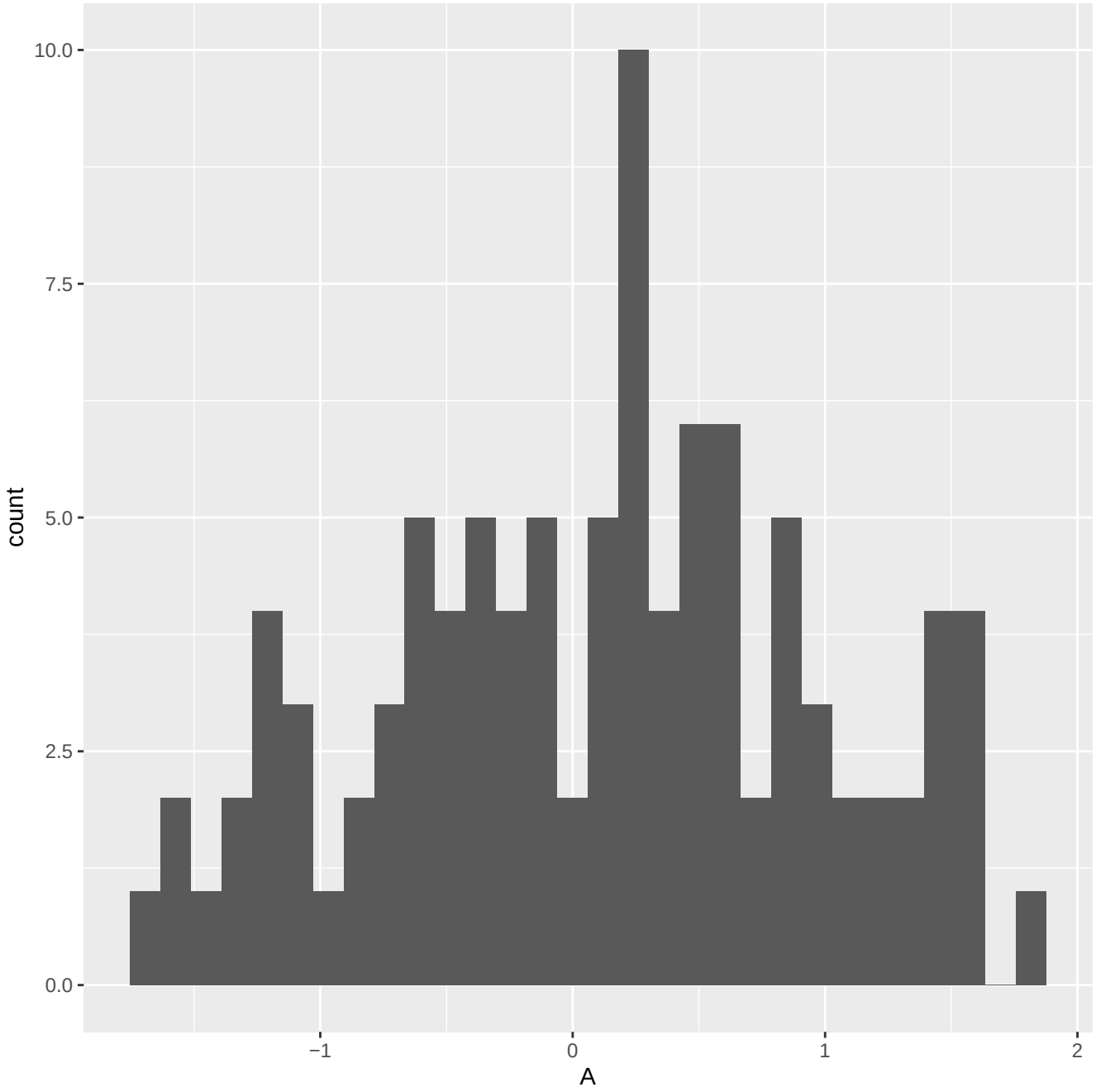



```
## [1] "mpg"
##     Min.  1st Qu.   Median     Mean  3rd Qu.     Max. varianza
## 10.40000 15.42500 19.20000 20.09062 22.80000 33.90000 36.32410
## [1] "cyl"
##     Min.  1st Qu.   Median     Mean  3rd Qu.     Max. varianza
## 4.000000 4.000000 6.000000 6.187500 8.000000 8.000000 3.189516
## [1] "disp"
##       Min.    1st Qu.     Median       Mean    3rd Qu.       Max.   varianza
##    71.1000   120.8250   196.3000   230.7219   326.0000   472.0000 15360.7998
## [1] "hp"
##      Min.   1st Qu.    Median      Mean   3rd Qu.      Max.  varianza
##   52.0000   96.5000  123.0000  146.6875  180.0000  335.0000 4700.8669
## [1] "drat"
```

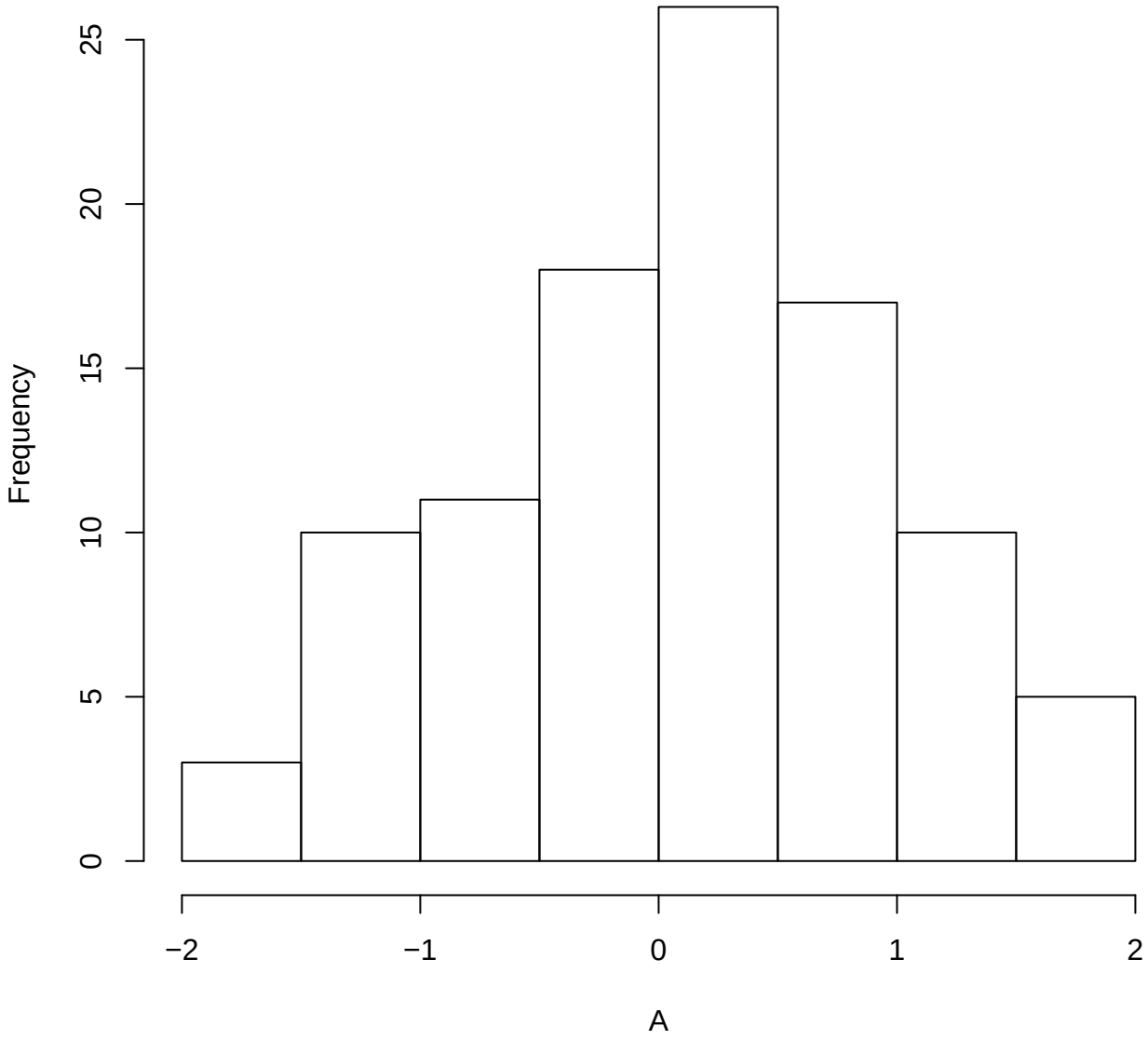
```
##      Min.   1st Qu.    Median      Mean   3rd Qu.      Max.  varianza
## 2.7600000 3.0800000 3.6950000 3.5965625 3.9200000 4.9300000 0.2858814
## [1] "wt"
##     Min.  1st Qu.   Median     Mean  3rd Qu.     Max. varianza
## 1.513000 2.581250 3.325000 3.217250 3.610000 5.424000 0.957379
## [1] "qsec"
##      Min.   1st Qu.    Median      Mean   3rd Qu.      Max.  varianza
## 14.500000 16.892500 17.710000 17.848750 18.900000 22.900000  3.193166
## [1] "vs"
##      Min.   1st Qu.    Median      Mean   3rd Qu.      Max.  varianza
## 0.0000000 0.0000000 0.0000000 0.4375000 1.0000000 1.0000000 0.2540323
## [1] "am"
##      Min.   1st Qu.    Median      Mean   3rd Qu.      Max.  varianza
## 0.0000000 0.0000000 0.0000000 0.4062500 1.0000000 1.0000000 0.2489919
## [1] "gear"
##      Min.   1st Qu.    Median      Mean   3rd Qu.      Max.  varianza
## 3.0000000 3.0000000 4.0000000 3.6875000 4.0000000 5.0000000 0.5443548
## [1] "carb"
##     Min.  1st Qu.   Median     Mean  3rd Qu.     Max. varianza
## 1.000000 2.000000 2.000000 2.812500 4.000000 8.000000 2.608871
```

```
graphics_wrap(data,tipo="unico",variable="A",gplot=TRUE)
```

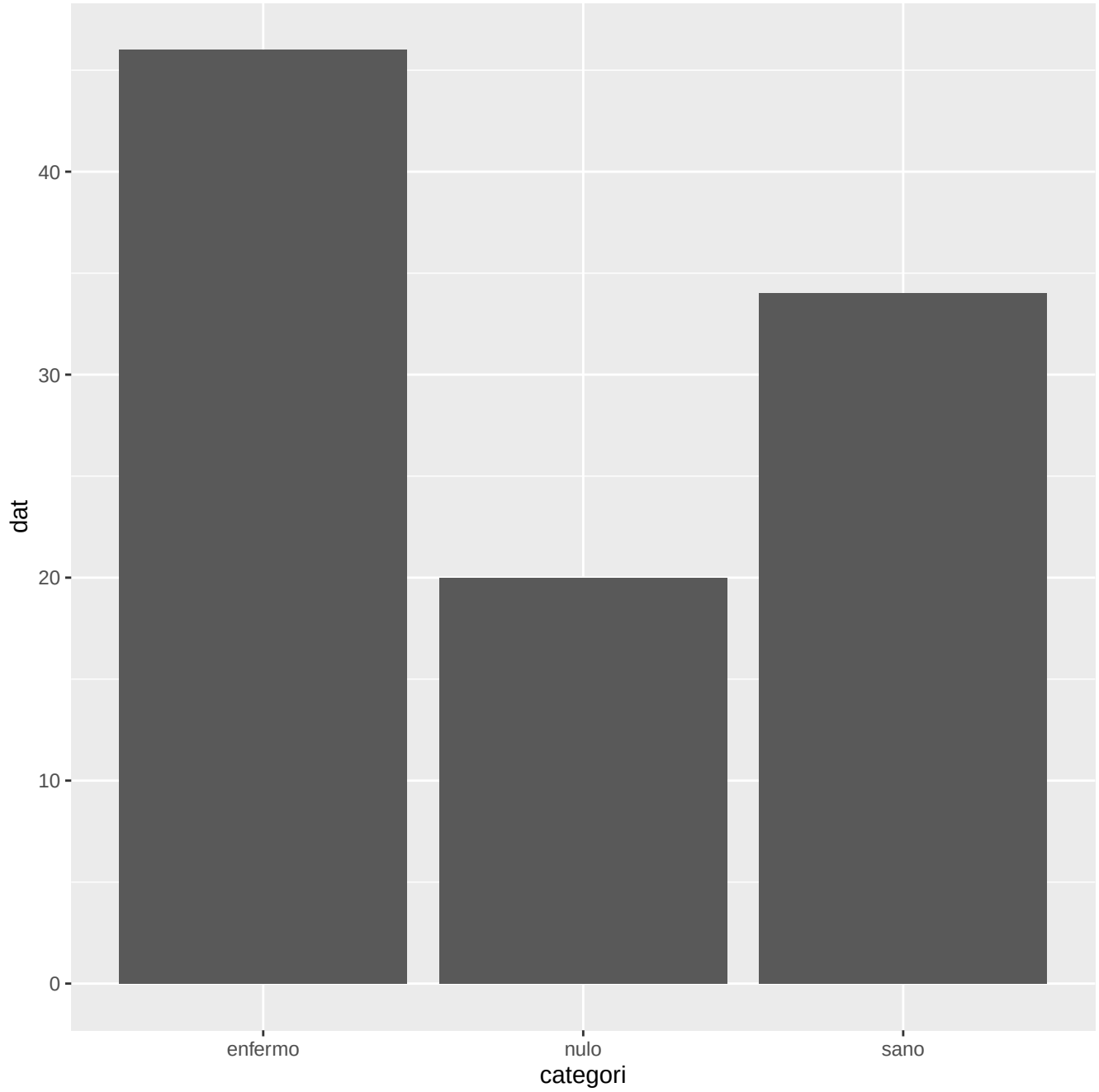
```
## `stat_bin()` using `bins = 30`.  Pick better value with `binwidth`.
```




```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
## -1.7192 -0.4758  0.2021  0.1023  0.6503  1.7922

graphics_wrap(data,tipo="unico",variable="A",gplot=FALSE)
```

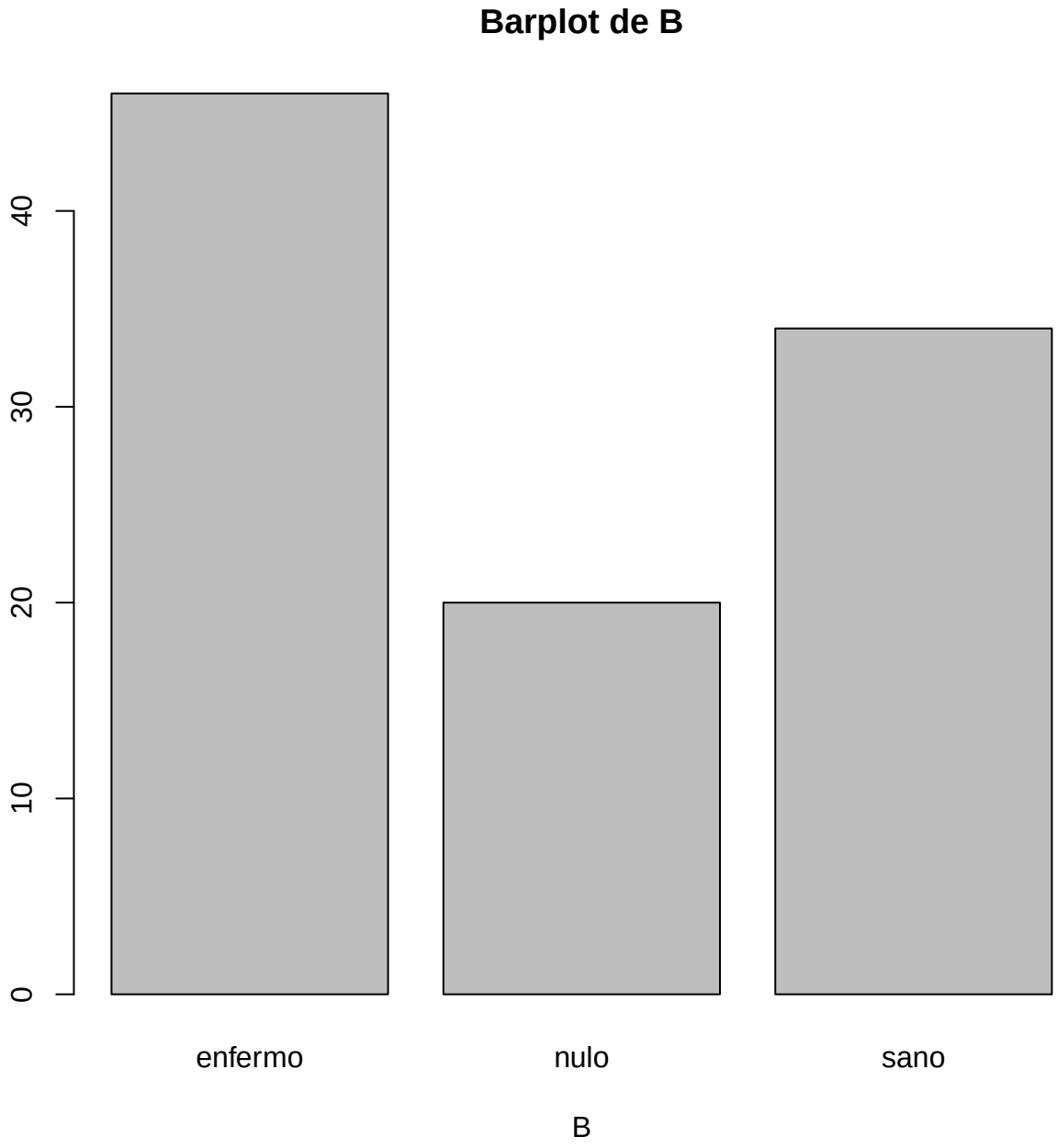



```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
## -1.7192 -0.4758  0.2021  0.1023  0.6503  1.7922

graphics_wrap(data,tipo="unico",variable="B",gplot=TRUE)
```



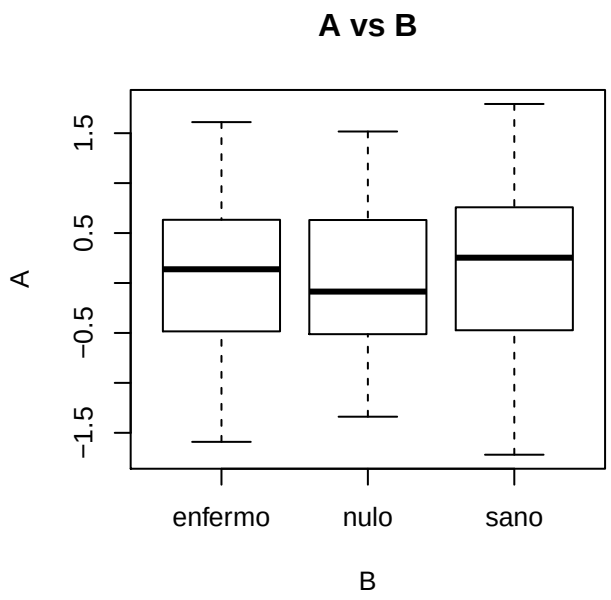

```
##
## enfermo    nulo    sano
##    0.46    0.20    0.34
```

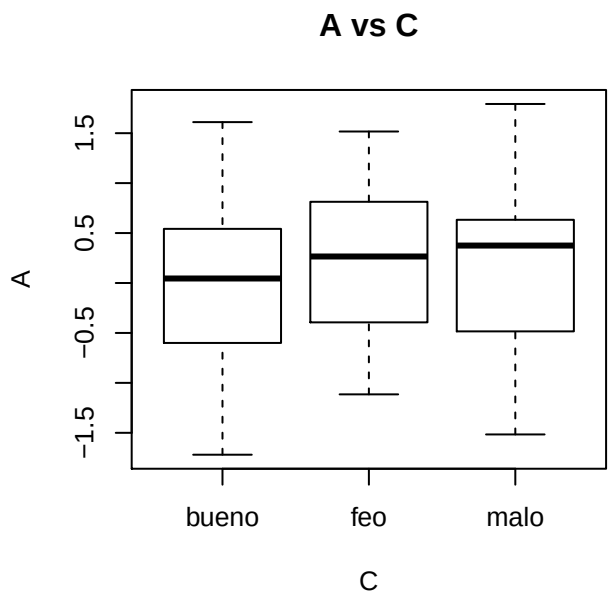
```
graphics_wrap(data,tipo="unico",variable="B",gplot=FALSE)
```

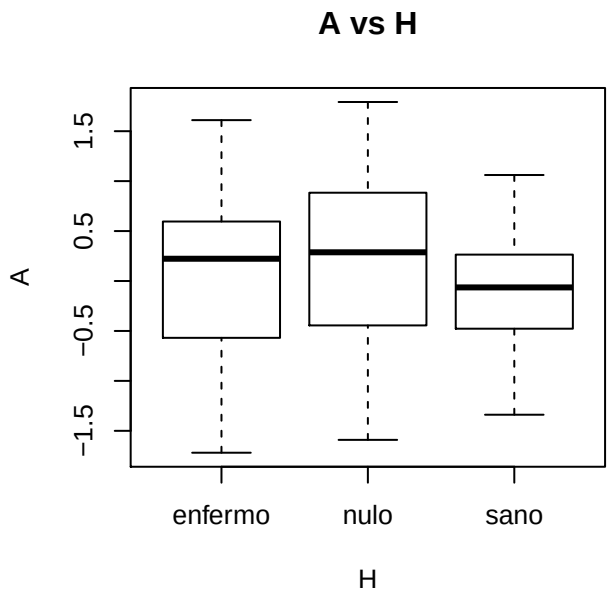


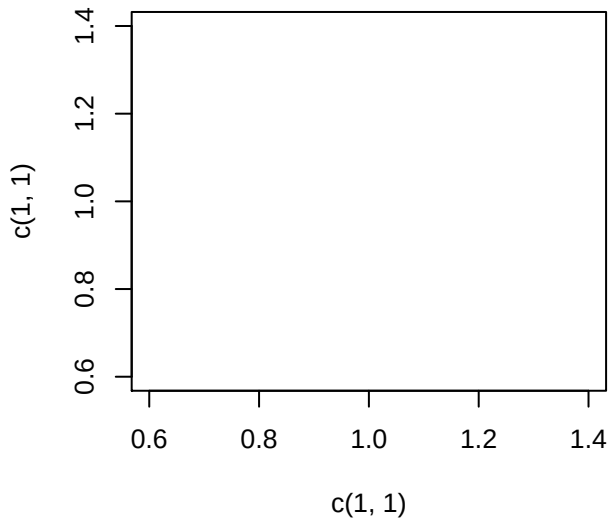

```
##
## enfermo    nulo    sano
##    0.46    0.20    0.34
```

```
graphics_wrap(data,tipo = "VS",variable="A",gplot=FALSE)
```





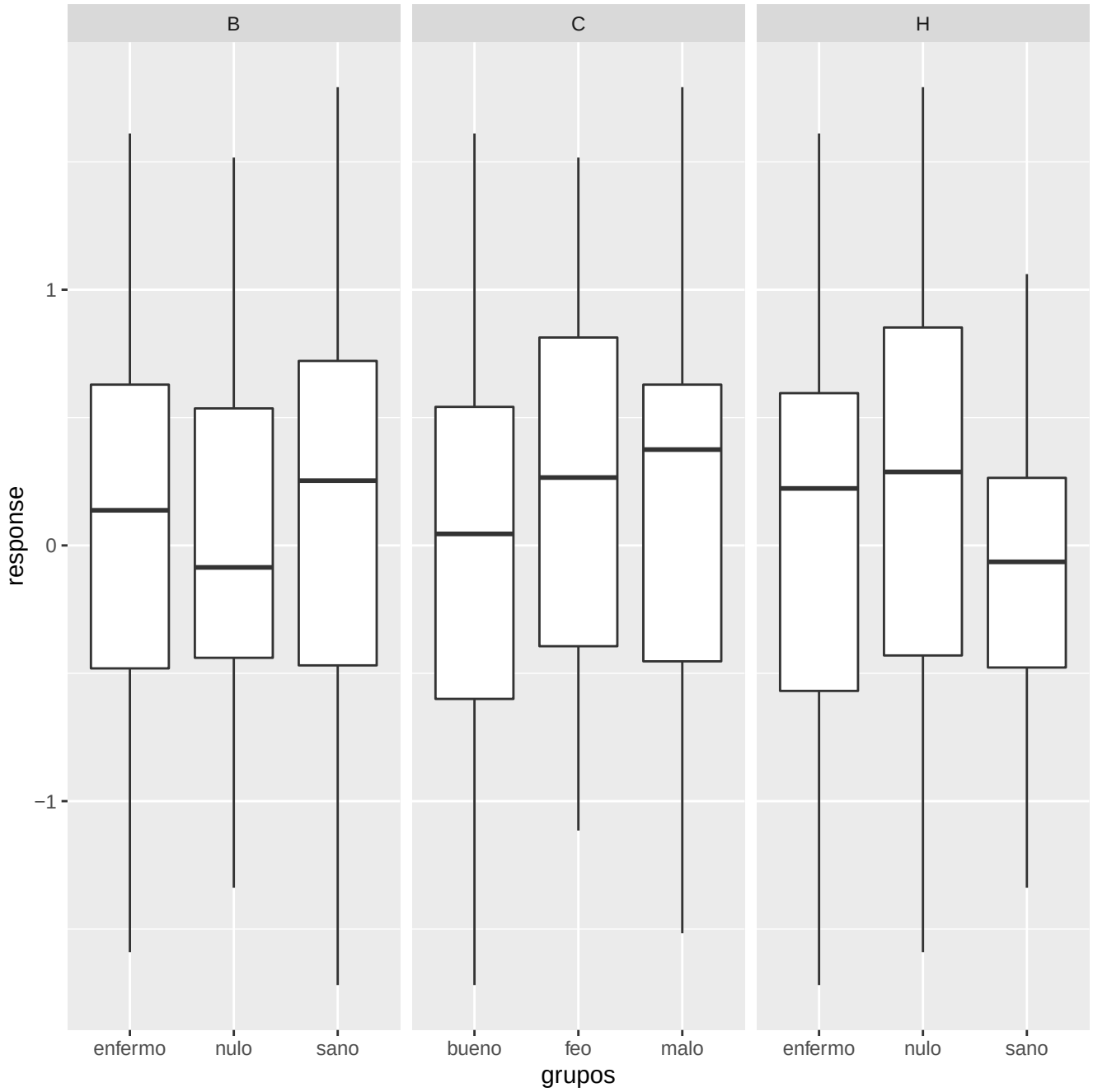





```
## [1] "A"
##       Min.    1st Qu.     Median       Mean    3rd Qu.       Max.   varianza
## -1.7192439 -0.4758483  0.2021251  0.1022933  0.6503286  1.7921763  0.7198792
## [1] "B"
##
## enfermo    nulo    sano
##    0.46    0.20    0.34
## [1] "C"
##
## bueno   feo  malo
##  0.57  0.25  0.18
## [1] "H"
##
```

```
## enfermo    nulo    sano
##    0.45    0.28    0.27
```

```
graphics_wrap(data,tipo = "VS",variable="A",gplot=TRUE)
```



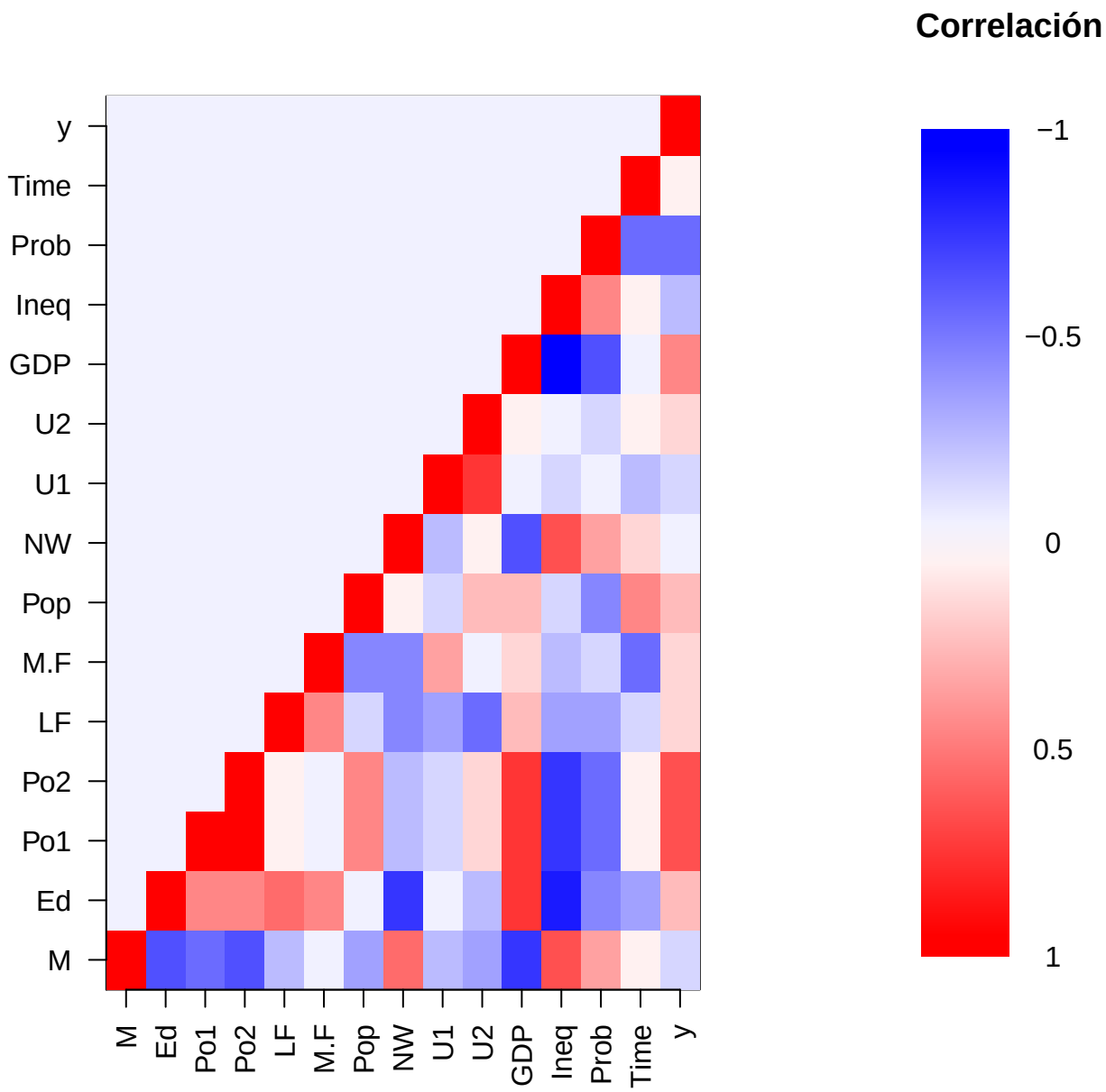

```
## [1] "A"
##       Min.    1st Qu.     Median       Mean    3rd Qu.       Max.   varianza
## -1.7192439 -0.4758483  0.2021251  0.1022933  0.6503286  1.7921763  0.7198792
## [1] "B"
##
## enfermo    nulo    sano
##    0.46    0.20    0.34
## [1] "C"
```

```
##
## bueno   feo  malo
##  0.57  0.25  0.18
## [1] "H"
##
## enfermo    nulo    sano
##    0.45    0.28    0.27
```

Ejercicio 2

```
options(max.print=100) #Se limita el Max.print para evitar que imprima grandes
#cantidades de datos.
library(MASS)
library(dplyr)

##
## Attaching package:  'dplyr'
## The following object is masked from 'package:MASS':
##
##     select
## The following objects are masked from 'package:stats':
##
##     filter, lag
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union

### En función de la información mostrada en la ayuda del paquete,
#se pasará la variable So
UScrime_ <- UScrime
UScrime_$So <- factor(UScrime$So,labels = c("Norte","Sur"))
#Se seleccionan las variables indicadas en el ejercicio así como se renombra
#las variables U1 y U2
task1 <- dplyr::select(UScrime_,c("M", "So", "Ed","Po1", "Pop","GDP", "U1", "U2")) %>% rename(Dese
task1

##      M    So  Ed Po1 Pop GDP Desempleo1 Desempleo2
## 1  151   Sur  91  58  33 394        108         41
## 2  143 Norte 113 103  13 557         96         36
## 3  142   Sur  89  45  18 318         94         33
## 4  136 Norte 121 149 157 673        102         39
## 5  141 Norte 121 109  18 578         91         20
## 6  121 Norte 110 118  25 689         84         29
## 7  127   Sur 111  82   4 620         97         38
## 8  131   Sur 109 115  50 472         79         35
## 9  157   Sur  90  65  39 421         81         28
## 10 140 Norte 118  71   7 526        100         24
## 11 124 Norte 105 121 101 657         77         35
```

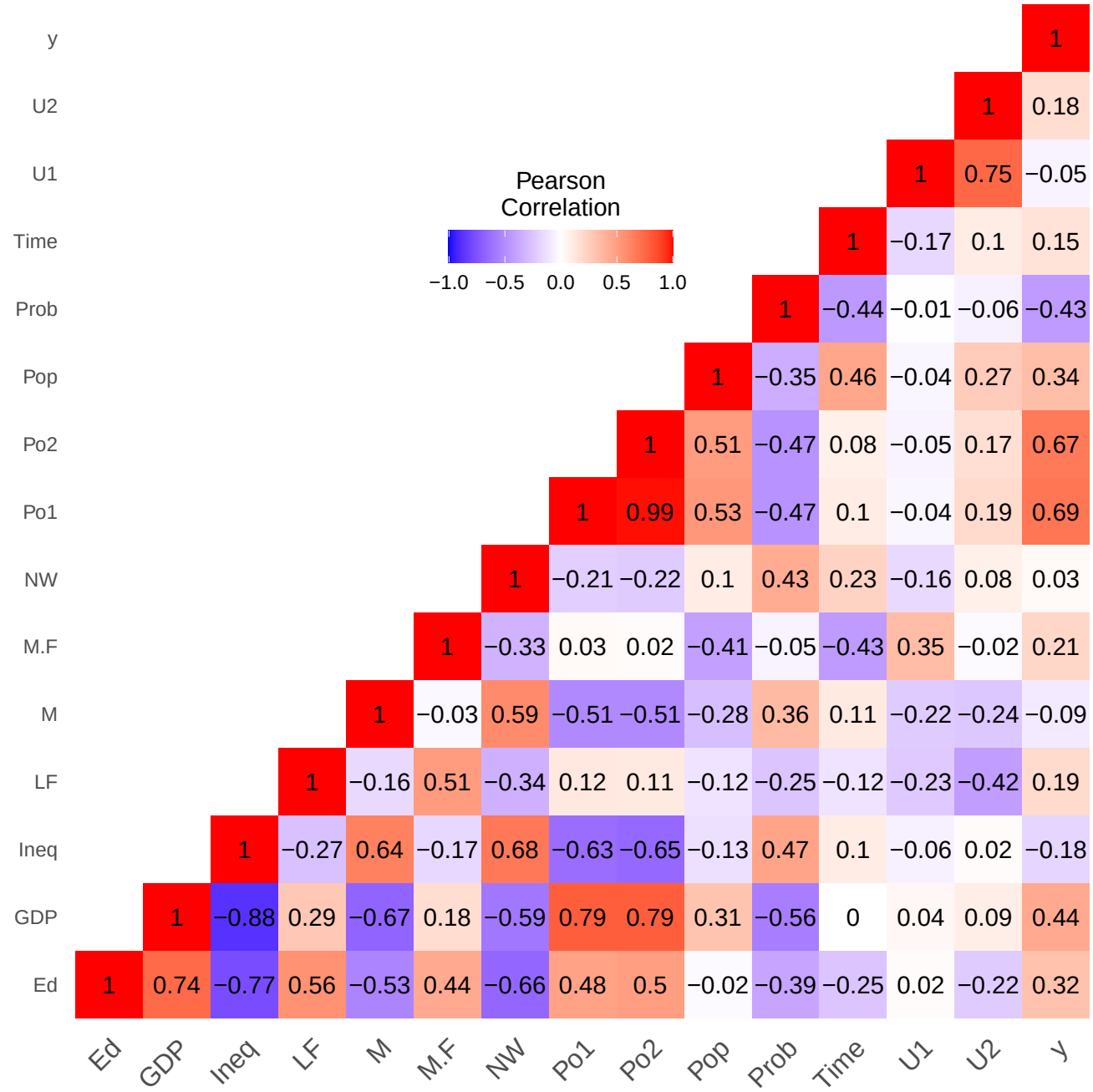
```
## 12 134 Norte 108  75  47 580         83         31
##  [ reached 'max' / getOption("max.print") -- omitted 35 rows ]

task2 <- dplyr::select(UScrime_,c("M", "So", "Ed","Po1", "Pop","GDP", "U1", "U2")) %>%
  rename(Desempleo1=U1) %>% rename(Desempleo2=U2)  %>% filter(So==0 & GDP > 530)
task2

## [1] M          So         Ed         Po1        Pop        GDP        Desempleo1
## [8] Desempleo2
## <0 rows> (or 0-length row.names)

UScrime_["Po1_discreto"] <- cut(UScrime_$Po1, 3) #se categoriza la variable Po1
#gracias a la función cut.
UScrime_

##     M    So  Ed Po1 Po2  LF  M.F Pop  NW  U1 U2 GDP Ineq     Prob    Time    y
## 1 151   Sur  91  58  56 510  950  33 301 108 41 394  261 0.084602 26.2011  791
## 2 143 Norte 113 103  95 583 1012  13 102  96 36 557  194 0.029599 25.2999 1635
## 3 142   Sur  89  45  44 533  969  18 219  94 33 318  250 0.083401 24.3006  578
## 4 136 Norte 121 149 141 577  994 157  80 102 39 673  167 0.015801 29.9012 1969
## 5 141 Norte 121 109 101 591  985  18  30  91 20 578  174 0.041399 21.2998 1234
##   Po1_discreto
## 1  (44.9,85.3]
## 2   (85.3,126]
## 3  (44.9,85.3]
## 4    (126,166]
## 5   (85.3,126]
##  [ reached 'max' / getOption("max.print") -- omitted 42 rows ]

graphics_wrap(UScrime_)
```




```
## [1] "M"
##     Min.  1st Qu.   Median     Mean  3rd Qu.     Max. varianza
## 119.0000 130.0000 136.0000 138.5745 146.0000 177.0000 157.9454
## [1] "Ed"
##     Min.  1st Qu.   Median     Mean  3rd Qu.     Max. varianza
##  87.0000  97.5000 108.0000 105.6383 114.5000 122.0000 125.1489
## [1] "Po1"
##     Min.  1st Qu.   Median     Mean  3rd Qu.     Max. varianza
##  45.0000  62.5000  78.0000  85.0000 104.5000 166.0000 883.2174
## [1] "Po2"
##      Min.   1st Qu.    Median      Mean   3rd Qu.      Max.  varianza
##  41.00000  58.50000  73.00000  80.23404  97.00000 157.00000 781.83534
## [1] "LF"
```

```
##      Min.   1st Qu.    Median      Mean   3rd Qu.      Max.  varianza
##  480.0000  530.5000  560.0000  561.1915  593.0000  641.0000 1633.1147
## [1] "M.F"
##      Min.   1st Qu.    Median      Mean   3rd Qu.      Max.  varianza
##  934.0000  964.5000  977.0000  983.0213  992.0000 1071.0000  868.3256
## [1] "Pop"
##       Min.    1st Qu.     Median       Mean    3rd Qu.       Max.   varianza
##    3.00000   10.00000   25.00000   36.61702   41.50000  168.00000 1449.41536
## [1] "NW"
##       Min.    1st Qu.     Median       Mean    3rd Qu.       Max.   varianza
##     2.0000    24.0000    76.0000   101.1277   132.5000   423.0000 10573.7660
## [1] "U1"
##      Min.   1st Qu.    Median      Mean   3rd Qu.      Max.  varianza
##  70.00000  80.50000  92.00000  95.46809 104.00000 142.00000 325.03700
## [1] "U2"
##     Min.  1st Qu.   Median     Mean  3rd Qu.     Max. varianza
## 20.00000 27.50000 34.00000 33.97872 38.50000 58.00000 71.32562
## [1] "GDP"
##     Min.  1st Qu.   Median     Mean  3rd Qu.     Max. varianza
##  288.000  459.500  537.000  525.383  591.500  689.000 9310.502
## [1] "Ineq"
##     Min.  1st Qu.   Median     Mean  3rd Qu.     Max. varianza
##  126.000  165.500  176.000  194.000  227.500  276.000 1591.696
## [1] "Prob"
##         Min.       1st Qu.       Median         Mean      3rd Qu.         Max.
## 0.0069000000 0.0327010000 0.0421000000 0.0470913830 0.0544500000 0.1198040000
##     varianza
## 0.0005169699
## [1] "Time"
##     Min.  1st Qu.   Median     Mean  3rd Qu.     Max. varianza
## 12.19960 21.60035 25.80060 26.59792 30.45075 44.00040 50.22408
## [1] "y"
##        Min.     1st Qu.      Median        Mean     3rd Qu.        Max.
##    342.0000    658.5000    831.0000    905.0851   1057.5000   1993.0000
##    varianza
## 149585.3839
```

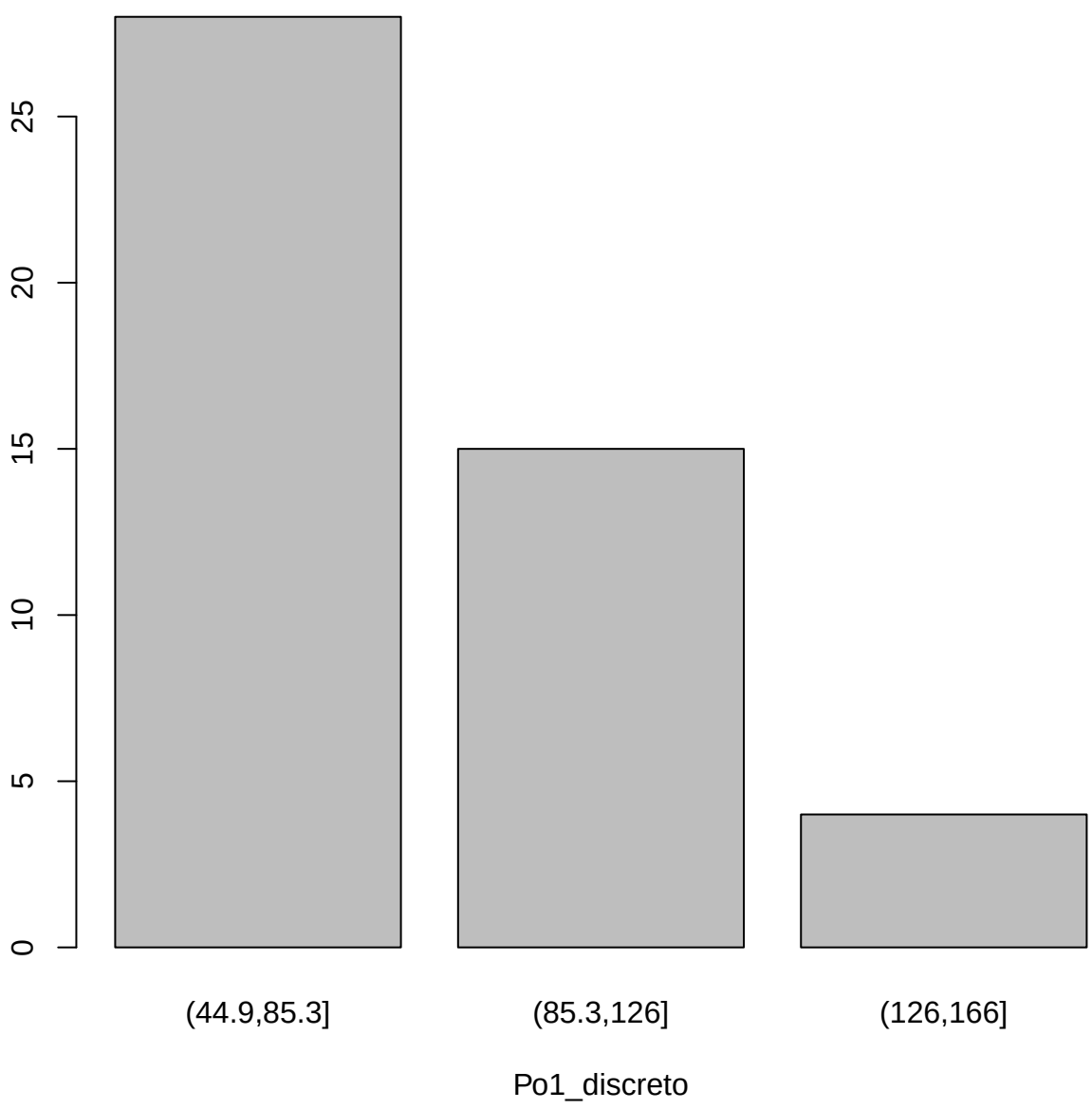
```
graphics_wrap(UScrime_,gplot=TRUE)
```




```
## [1] "M"
##     Min.  1st Qu.   Median     Mean  3rd Qu.     Max. varianza
## 119.0000 130.0000 136.0000 138.5745 146.0000 177.0000 157.9454
## [1] "Ed"
##     Min.  1st Qu.   Median     Mean  3rd Qu.     Max. varianza
##  87.0000  97.5000 108.0000 105.6383 114.5000 122.0000 125.1489
## [1] "Po1"
##     Min.  1st Qu.   Median     Mean  3rd Qu.     Max. varianza
##  45.0000  62.5000  78.0000  85.0000 104.5000 166.0000 883.2174
## [1] "Po2"
##      Min.   1st Qu.    Median      Mean   3rd Qu.      Max.  varianza
##  41.00000  58.50000  73.00000  80.23404  97.00000 157.00000 781.83534
## [1] "LF"
```

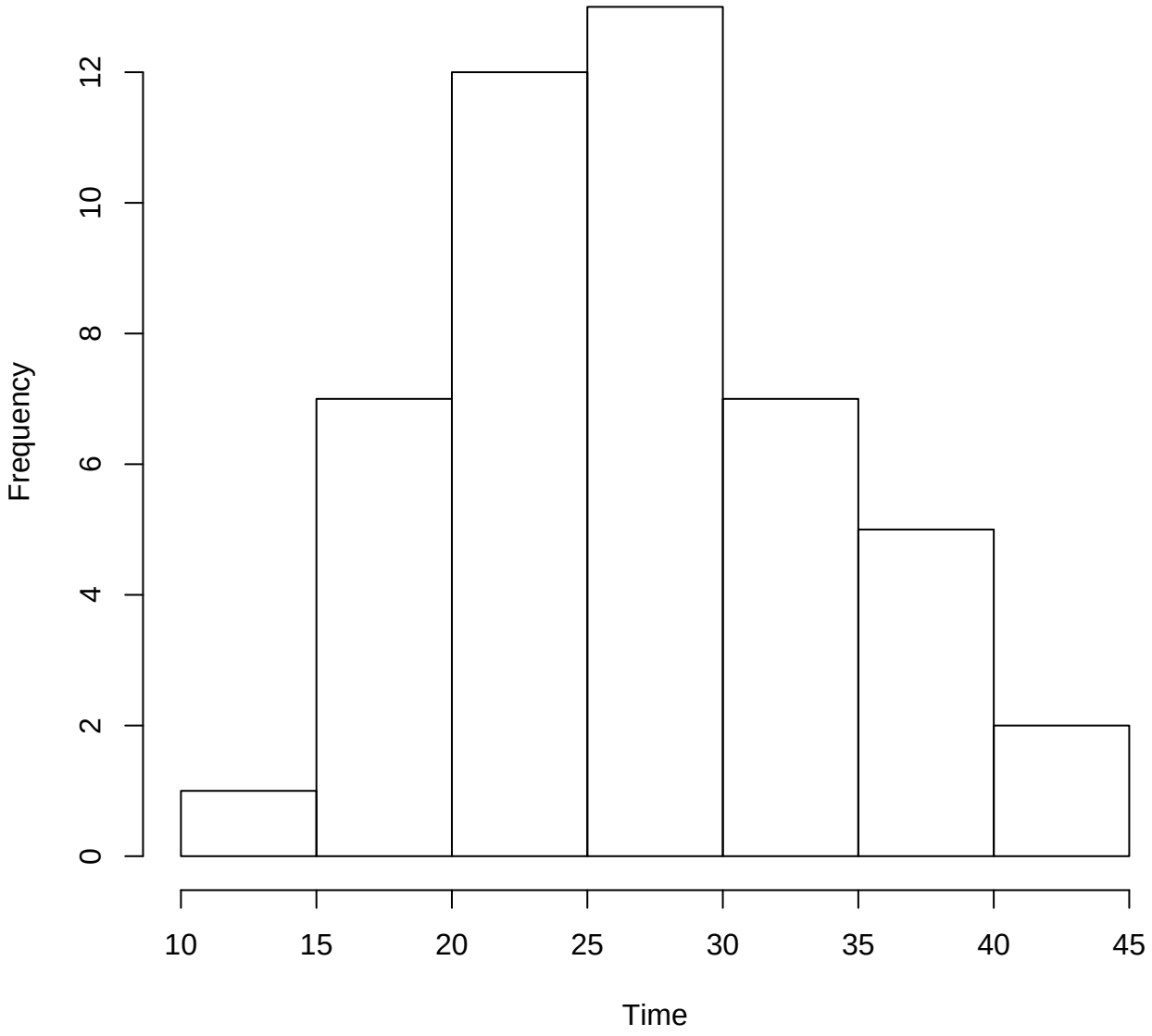
```
##      Min.   1st Qu.    Median      Mean   3rd Qu.      Max.  varianza
##  480.0000  530.5000  560.0000  561.1915  593.0000  641.0000 1633.1147
## [1] "M.F"
##      Min.   1st Qu.    Median      Mean   3rd Qu.      Max.  varianza
##  934.0000  964.5000  977.0000  983.0213  992.0000 1071.0000  868.3256
## [1] "Pop"
##       Min.    1st Qu.     Median       Mean    3rd Qu.       Max.   varianza
##    3.00000   10.00000   25.00000   36.61702   41.50000  168.00000 1449.41536
## [1] "NW"
##       Min.    1st Qu.     Median       Mean    3rd Qu.       Max.   varianza
##     2.0000    24.0000    76.0000   101.1277   132.5000   423.0000 10573.7660
## [1] "U1"
##      Min.   1st Qu.    Median      Mean   3rd Qu.      Max.  varianza
##  70.00000  80.50000  92.00000  95.46809 104.00000 142.00000 325.03700
## [1] "U2"
##     Min.  1st Qu.   Median     Mean  3rd Qu.     Max. varianza
## 20.00000 27.50000 34.00000 33.97872 38.50000 58.00000 71.32562
## [1] "GDP"
##     Min.  1st Qu.   Median     Mean  3rd Qu.     Max. varianza
##  288.000  459.500  537.000  525.383  591.500  689.000 9310.502
## [1] "Ineq"
##     Min.  1st Qu.   Median     Mean  3rd Qu.     Max. varianza
##  126.000  165.500  176.000  194.000  227.500  276.000 1591.696
## [1] "Prob"
##         Min.       1st Qu.       Median         Mean      3rd Qu.         Max.
## 0.0069000000 0.0327010000 0.0421000000 0.0470913830 0.0544500000 0.1198040000
##     varianza
## 0.0005169699
## [1] "Time"
##     Min.  1st Qu.   Median     Mean  3rd Qu.     Max. varianza
## 12.19960 21.60035 25.80060 26.59792 30.45075 44.00040 50.22408
## [1] "y"
##        Min.     1st Qu.      Median        Mean     3rd Qu.        Max.
##    342.0000    658.5000    831.0000    905.0851   1057.5000   1993.0000
##    varianza
## 149585.3839
```

```
graphics_wrap(UScrime_,tipo="unico",variable = "Po1_discreto")
```

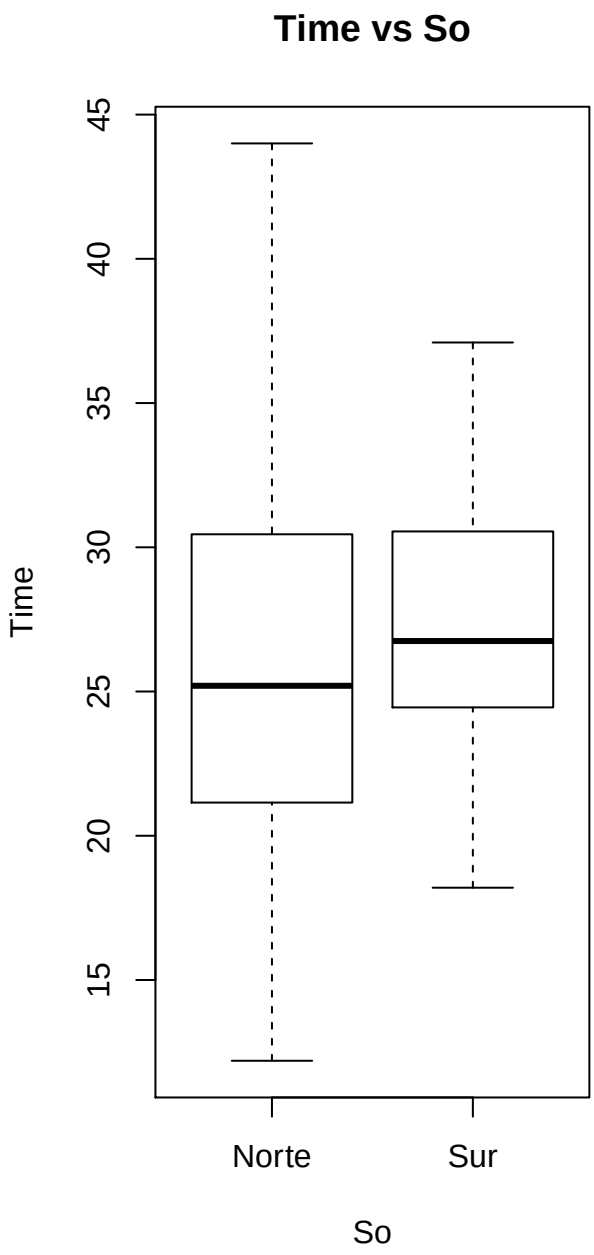


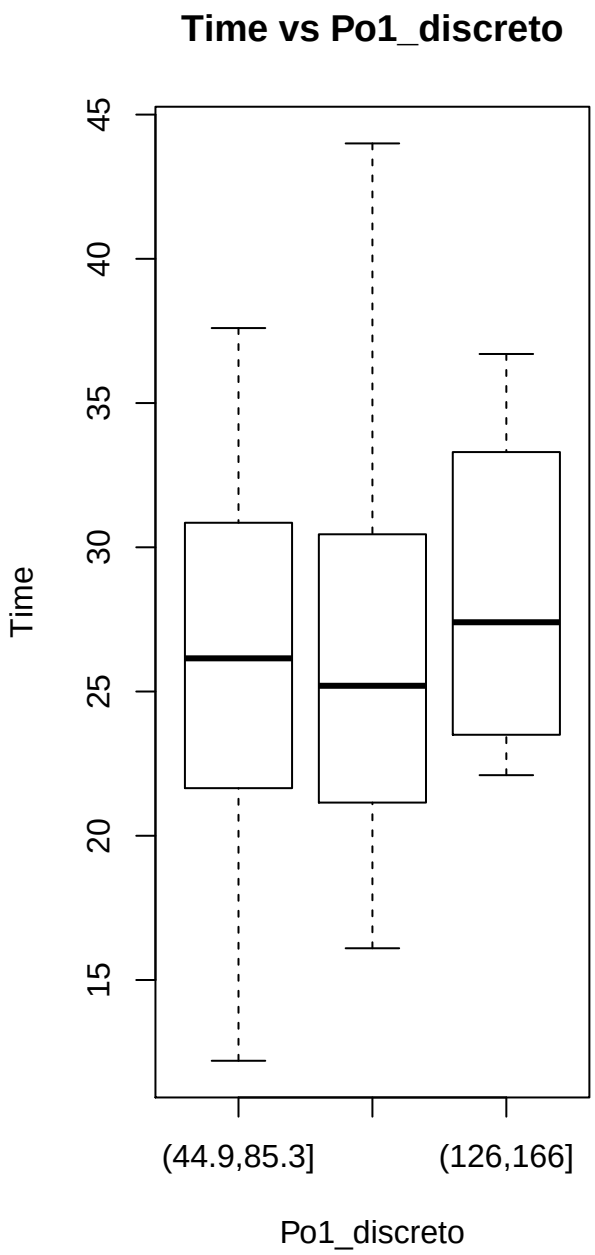

```
##
## (44.9,85.3]  (85.3,126]   (126,166]
##  0.59574468  0.31914894  0.08510638
```

```
graphics_wrap(UScrime_,tipo="unico",variable = "Time")
```




```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   12.20   21.60   25.80   26.60   30.45   44.00

graphics_wrap(UScrime_,tipo="VS",variable="Time")
```






```
## [1] "Time"
##     Min.  1st Qu.   Median     Mean  3rd Qu.     Max. varianza
## 12.19960 21.60035 25.80060 26.59792 30.45075 44.00040 50.22408
## [1] "So"
##
##     Norte       Sur
## 0.6595745 0.3404255
## [1] "Po1_discreto"
##
## (44.9,85.3]  (85.3,126]   (126,166]
##  0.59574468  0.31914894  0.08510638

#Se prueba la función y las distintas opciones con el dataframe generado.
```