dviz-practice

These exercises were adapted from Pearson, "Exploratory Data Analysis Using R", CRC Press (2019), chapter 2.8, p.69-70.

1. The `fgl` data frame from the `MASS` package characterizes 214 forensic glass samples in terms of their refractive index (RI), a type designation (type), and percentages by weight of eight different elemental oxides.

2. Load `fgl` and display its structure.

```
library(MASS)
data(fgl)
str(fgl)
```

```
'data.frame':    214 obs. of  10 variables:
 $ RI  : num  3.01 -0.39 -1.82 -0.34 -0.58 ...
 $ Na  : num  13.6 13.9 13.5 13.2 13.3 ...
 $ Mg  : num  4.49 3.6 3.55 3.69 3.62 3.61 3.6 3.61 3.58 3.6 ...
 $ Al  : num  1.1 1.36 1.54 1.29 1.24 1.62 1.14 1.05 1.37 1.36 ...
 $ Si  : num  71.8 72.7 73 72.6 73.1 ...
 $ K   : num  0.06 0.48 0.39 0.57 0.55 0.64 0.58 0.57 0.56 0.57 ...
 $ Ca  : num  8.75 7.83 7.78 8.22 8.07 8.07 8.17 8.24 8.3 8.4 ...
 $ Ba  : num  0 0 0 0 0 0 0 0 0 0 ...
 $ Fe  : num  0 0 0 0 0 0.26 0 0 0 0.11 ...
 $ type: Factor w/ 6 levels "WinF","WinNF",..: 1 1 1 1 1 1 1 1 1 1 ...
```

3. How many samples in this data frame come from vehicle headlamps?

   Tip: check out the documentation for the dataset! The answer is in there but you should still run the code to make sure!

```
sum(fgl$type == "Head")
```

```
[1] 29
```

4. Generate a plot of the magnesium oxide concentration (`Mg`) versus record number, with these features:

   - x-axis label "Record number"
   - y-axis label: "Mg concentration"
   - title: "Magnesium oxide concentration in the fgl dataset"
   - use the `las` parameter to make the labels horizontal for both axis

```
plot(
  fgl$Mg,
  xlab = "Record number",
  ylab = "Mg concentration",
  main = "Magnesium oxide concentration records in the fgl dataset",
  las = 1)
```

Magnesium oxide concentration records in the fgl dataset
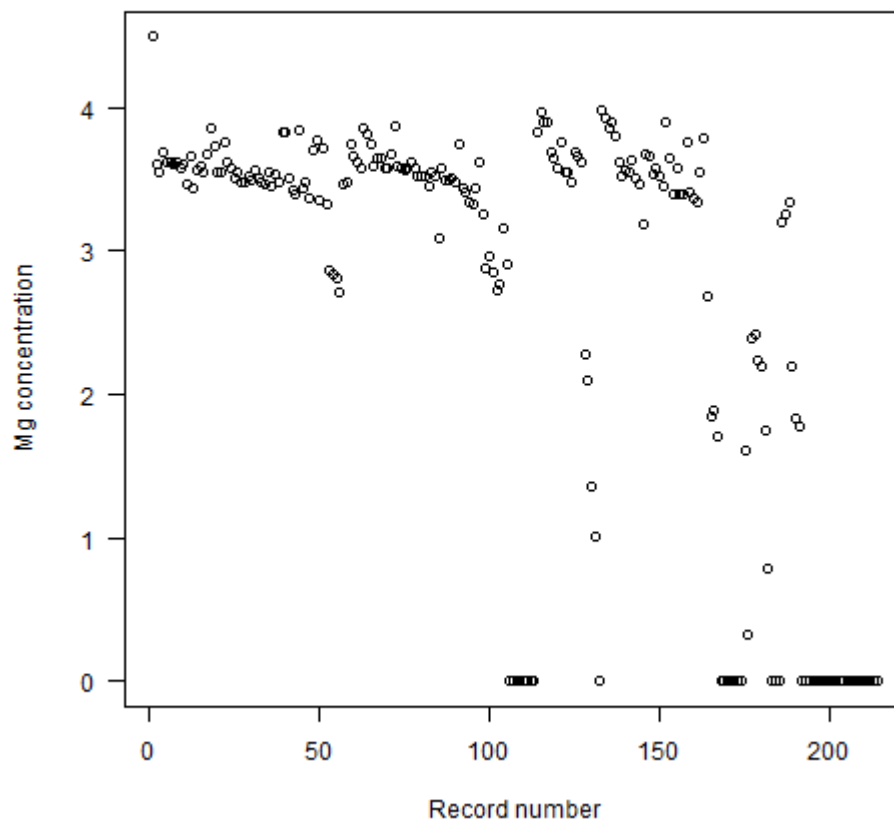
5. The generic function `plot(x,y)` generates a boxplot when `x` is a categorical variable and `y` is a numerical variable. Create a boxplot summary showing how the magnesium concentration in the `fgl` dataset varies with the different values of the categorical `type` variable. Customize as follows:

   - Specify the x-axis label as "Forensice glass type"
   - Specify the y-axis label as "Mg concentration",
   - Title the plot "Mg concentration for different glass types in the fgl data set". Use `\n` to break titles that are too long to fit
   - Make the labels horizontal for both axes

```
plot(
  x = fgl$type,
  y = fgl$Mg,
  xlab = "Forensic glass type",
  ylab = "Mg concentration",
  main = "Magnesium oxide concentration\n for different glass types in the fgl dataset",
  las = 1)
```

Magnesium oxide concentration
for different glass types in the fgl dataset
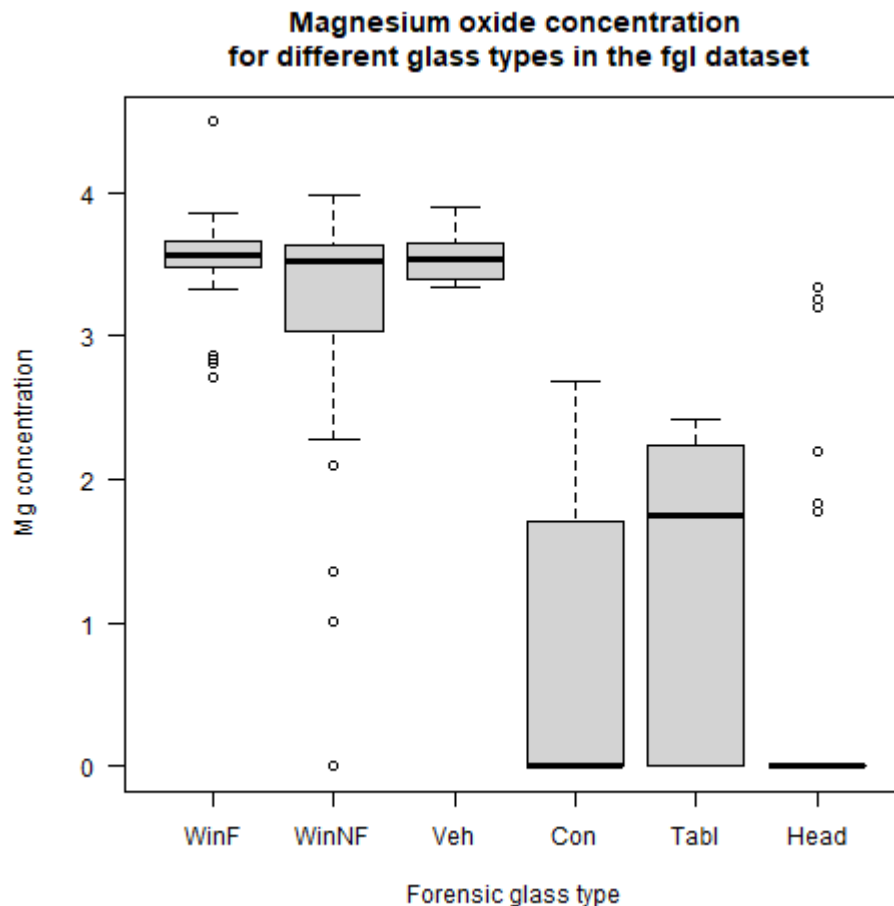
6. Using the `barplot` function, construct the following summary for the `type` variable from the `fgl` data frame in the `MASS` package:

   - Create a horizontal barplot of the `type` variable record frequencies (counts) using `table`
   - Display the frequencies sorted from largest (top) to smallest (bottom)
   - Label the x-axis "Record listing glass type"
   - Remove the y-axis label
   - Use the `las` parameter to make the `type` name labels horizontal
   - Use the `font.lab` parameter to make the x-label bold face
   - Use the `paste` function to create a string `tstring` for the title consisting of[1]:
     1. the text `"Horizontal barplot of the"`,
     2. the number of `levels` of the `type` variable,
     3. the text `"glass types"`
   - Add a title to the plot using `title` and the `tstring` variable, and change the font to *italics* with the `font.main` parameter

   Tip: plot the the simple `barplot(height=)` first before customizing!

```
types <- sort(table(fgl$type))  # sorted frequency table

barplot(
  height = types,
  xlab = "Record listing glass type",
```

```
    ylab = "",
    font.lab = 2,
    horiz = TRUE,
    las = 2)

tstring <- paste( "Horizontal barplot of the",
                  length(types),
                  "glass types")
title(tstring, font.main=3)
```

Horizontal barplot of the 6 glass types

Record listing glass type
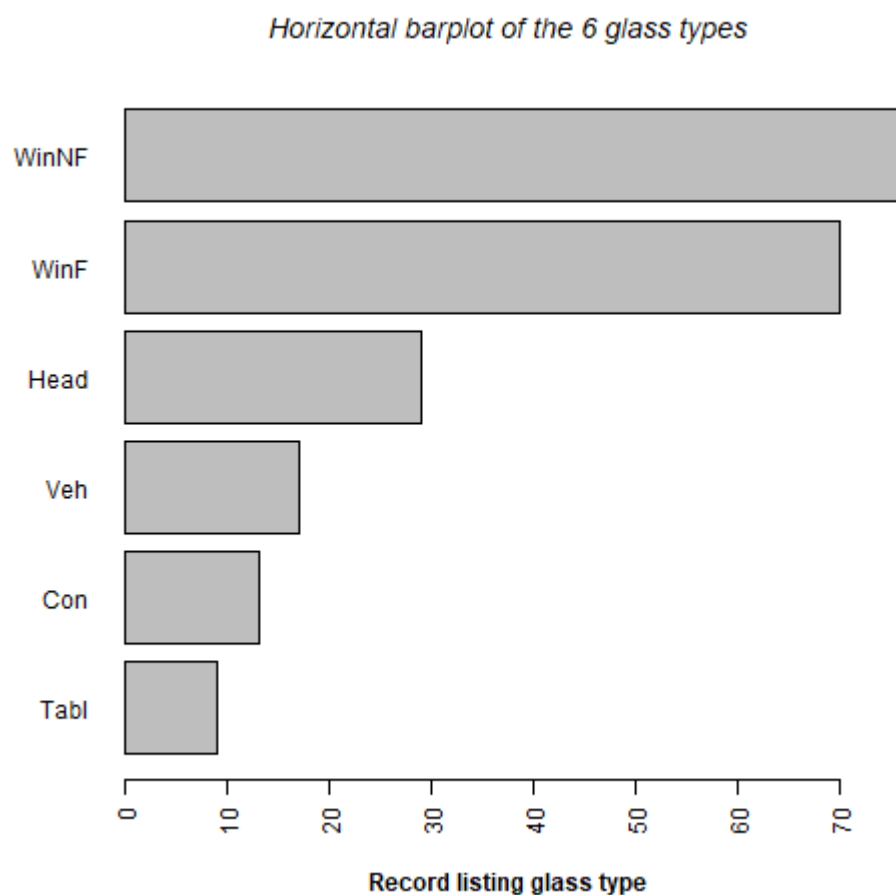
Footnotes:

[1] The function `paste` in action looks like this:

```
> n <- 50
> paste("I am", n, "years old")
[1] "I am 50 years old"
```

Created: 2022-10-28 Fri 16:29