Analisis biométrico de actividad y calidad del sueño en diferentes capitales europeas.

Jose Javier

7/10/2019

```
library(readr)
library(dplyr)
```

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

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

```
library(tidyverse)
```

```
## -- Attaching packages ---------------------------------------------- tidyverse 1.2.1 --
```

```
## v ggplot2 3.2.1     v purrr   0.3.2
## v tibble  2.1.3     v stringr 1.4.0
## v tidyr   1.0.0     v forcats 0.4.0
```

```
## -- Conflicts ------------------------------------------------- tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
library(ggplot2)
#1. Crear un nuevo proyecto denominado practica 4.

# 2. Mediante la libreria readr, o mediante los menus de RStudio, leer los datasets sleep.csv y activit
# ambos archivos deben estar previamente en la carpeta del proyecto creado
activities <- read_csv("C:/Users/Equipo/Desktop/CUNEF/Programacion con R/Proyecto/Practica 4/Practica 4
```

```
## Parsed with column specification:
## cols(
##   de = col_datetime(format = ""),
##   a = col_datetime(format = ""),
##   `from (manual)` = col_logical(),
##   `to (manual)` = col_logical(),
##   Timezone = col_character(),
##   `Activity type` = col_character(),
##   Data = col_character(),
##   GPS = col_character(),
##   Modified = col_datetime(format = "")
## )
```

```r
sleep <- read_csv("C:/Users/Equipo/Desktop/CUNEF/Programacion con R/Proyecto/Practica 4/Practica 4/slee
```

```
## Parsed with column specification:
## cols(
##   de = col_datetime(format = ""),
##   a = col_datetime(format = ""),
##   `ligero (s)` = col_double(),
##   `profundo (s)` = col_double(),
##   `Rem (seg)` = col_double(),
##   `despierto (s)` = col_double(),
##   despertar = col_double(),
##   `Duration to sleep (s)` = col_double(),
##   `Duration to wake up (s)` = col_double(),
##   `Snoring (s)` = col_double(),
##   `Snoring episodes` = col_double(),
##   `Average heart rate` = col_double(),
##   `Heart rate (min)` = col_double(),
##   `Heart rate (max)` = col_double()
## )
```

```r
# 3.Comprobar el contenido con View y contar cuantos NAs hay en la columna GPS del dataset activities
View(activities)
NAs<-sum(is.na(activities$GPS))


# 4. Crear un objeto R denominado act_new que contenga solo las variables
# siguientes: 1,2,5-6
act_new <- select(activities, de, a, Timezone, `Activity type`)
# 5. Renombrar la variable 'Activity type' con el nombre 'tipo' y la variable 'Time zone' como 'ciudad'
act_new <- rename(act_new, ciudad = Timezone, tipo = `Activity type`)

# 6. Realizar un recuento de tipo de actividad con summary. Para ello
# debes transformar previamente la variable tipo a factor con as.factor.
# Crea un grafico de barras con dicha variable par visualizar las frecuencias.
# Haz lo mismo para la variable ciudad
act_new$tipo <- as.factor(act_new$tipo)
str(act_new)
```

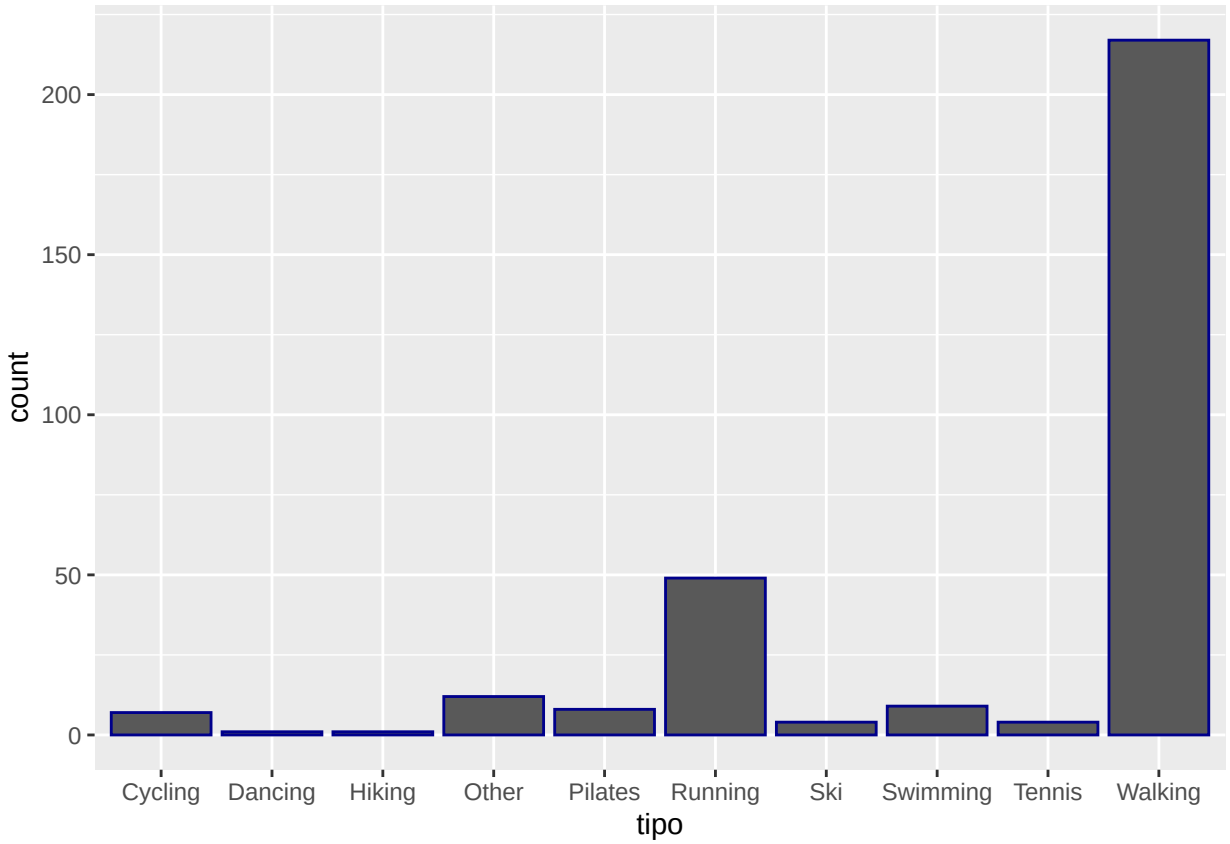
```
## Classes 'spec_tbl_df', 'tbl_df', 'tbl' and 'data.frame': 312 obs. of  4 variables:
##  $ de    : POSIXct, format: "2018-11-05 08:30:00" "2018-11-05 10:02:00" ...
##  $ a     : POSIXct, format: "2018-11-05 09:15:00" "2018-11-05 10:21:00" ...
##  $ ciudad: chr  "Europe/Madrid" "Europe/Madrid" "Europe/Madrid" "Europe/Madrid" ...
##  $ tipo  : Factor w/ 10 levels "Cycling","Dancing",..: 8 10 10 10 10 10 10 10 10 10 ...
##  - attr(*, "spec")=
##   .. cols(
##   ..   de = col_datetime(format = ""),
##   ..   a = col_datetime(format = ""),
##   ..   `from (manual)` = col_logical(),
##   ..   `to (manual)` = col_logical(),
##   ..   Timezone = col_character(),
##   ..   `Activity type` = col_character(),
##   ..   Data = col_character(),
##   ..   GPS = col_character(),
```

```
##   ..   Modified = col_datetime(format = "")
##   .. )
```

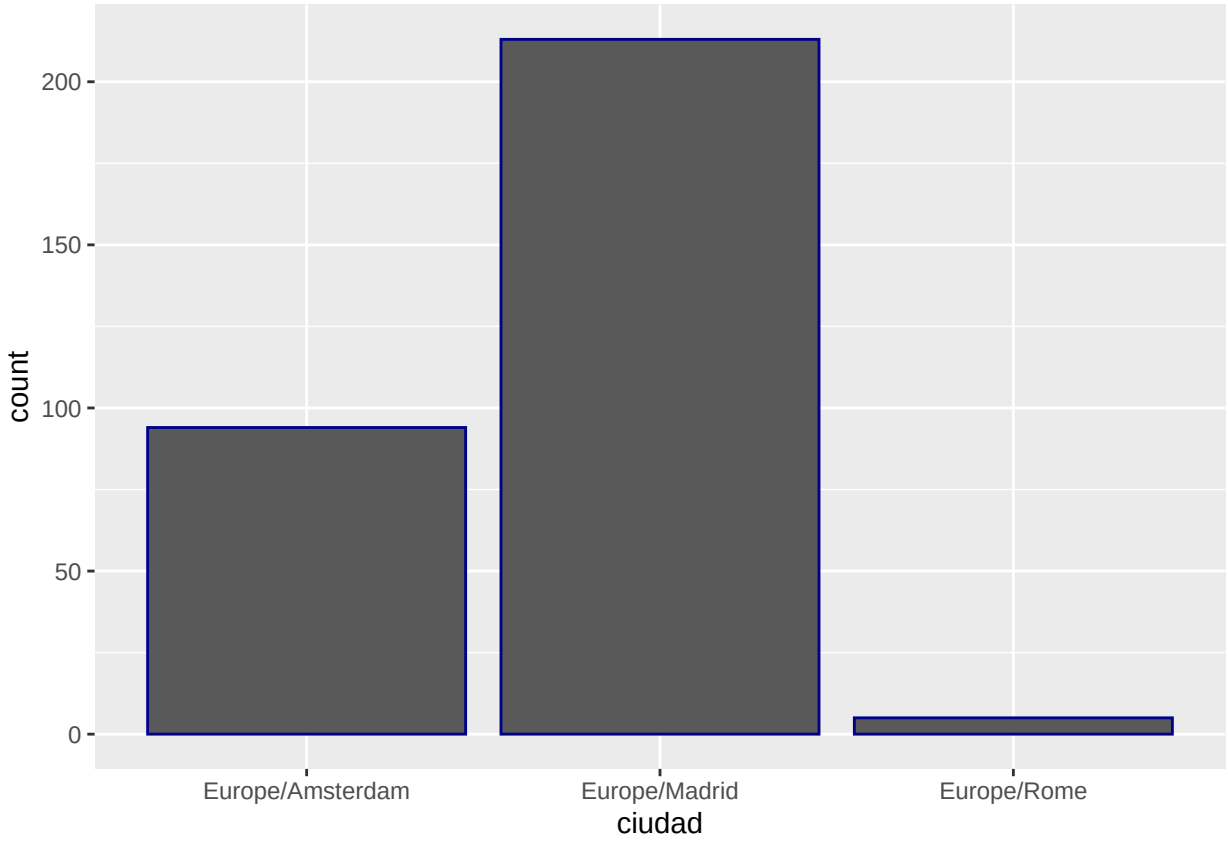
```
summary(act_new$tipo)
```

```
##  Cycling  Dancing   Hiking    Other  Pilates  Running      Ski Swimming
##        7        1        1       12        8       49        4        9
##   Tennis  Walking
##        4      217
```

```
ggplot(data = act_new)+
  geom_bar(mapping = aes(x = tipo), color = "darkblue")
```




```
#7. Filtrar los registros de act_new que correspondan con ciudad Amsterdam en otro objeto
# y lo mismo con Madrid. Con esos nuevos objetos determina los deportes que
# no se practican en Amsterdam y sí en Madrid y viceversa. Genera graficos para visualizar los resultad

Amsterdam <- filter(act_new, ciudad == "Europe/Amsterdam")

Madrid <- filter(act_new, ciudad == "Europe/Madrid")

ggplot(data = act_new)+
  geom_bar(mapping = aes(x = ciudad), color = "darkblue")
```



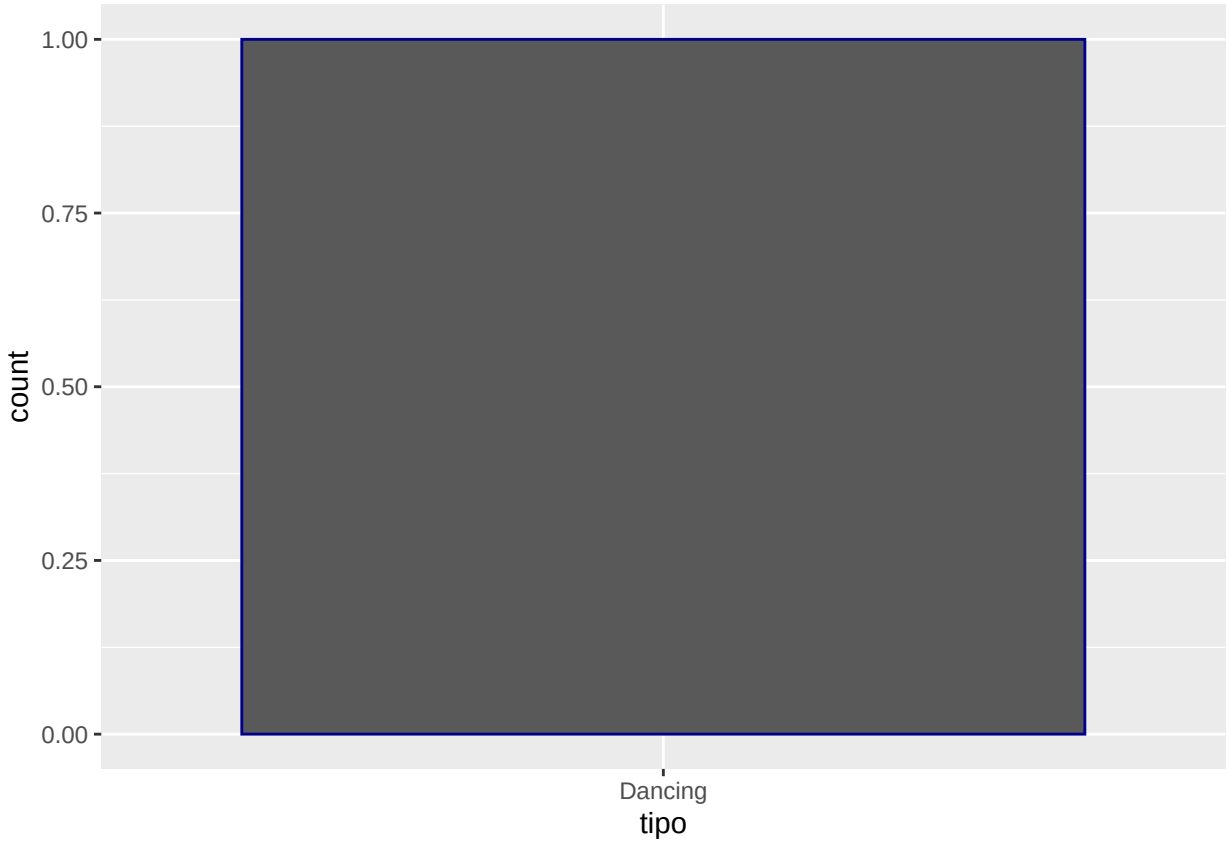

```
####Deportes que estan unicamente en Amsterdam
a <- data.frame(summary(Amsterdam$tipo), summary(Madrid$tipo))
a <- rename(a, Amsterdam = summary.Amsterdam.tipo. , Madrid = summary.Madrid.tipo.)

#Al hacer summary he visto que la actividad que solo realiza en Amsterdam es Dancing
#entonces he creado un objeto nuevo llamado Ams_dep para tener unicamente lo que solo
#pertenece a Amsterdam.
summary(Amsterdam)
```

```
##        de                          a
##  Min.   :2019-03-19 07:15:00   Min.   :2019-03-19 08:15:00
##  1st Qu.:2019-04-19 11:38:15   1st Qu.:2019-04-19 11:47:00
##  Median :2019-07-17 19:36:30   Median :2019-07-17 19:54:00
##  Mean   :2019-06-26 18:00:35   Mean   :2019-06-26 18:16:45
##  3rd Qu.:2019-08-26 12:16:45   3rd Qu.:2019-08-26 12:36:45
##  Max.   :2019-09-28 08:33:00   Max.   :2019-09-28 08:42:00
##
##     ciudad              tipo
##  Length:94          Walking:59
##  Class :character   Running:25
##  Mode  :character   Cycling: 6
##                     Pilates: 3
##                     Dancing: 1
##                     Hiking : 0
##                     (Other): 0
```

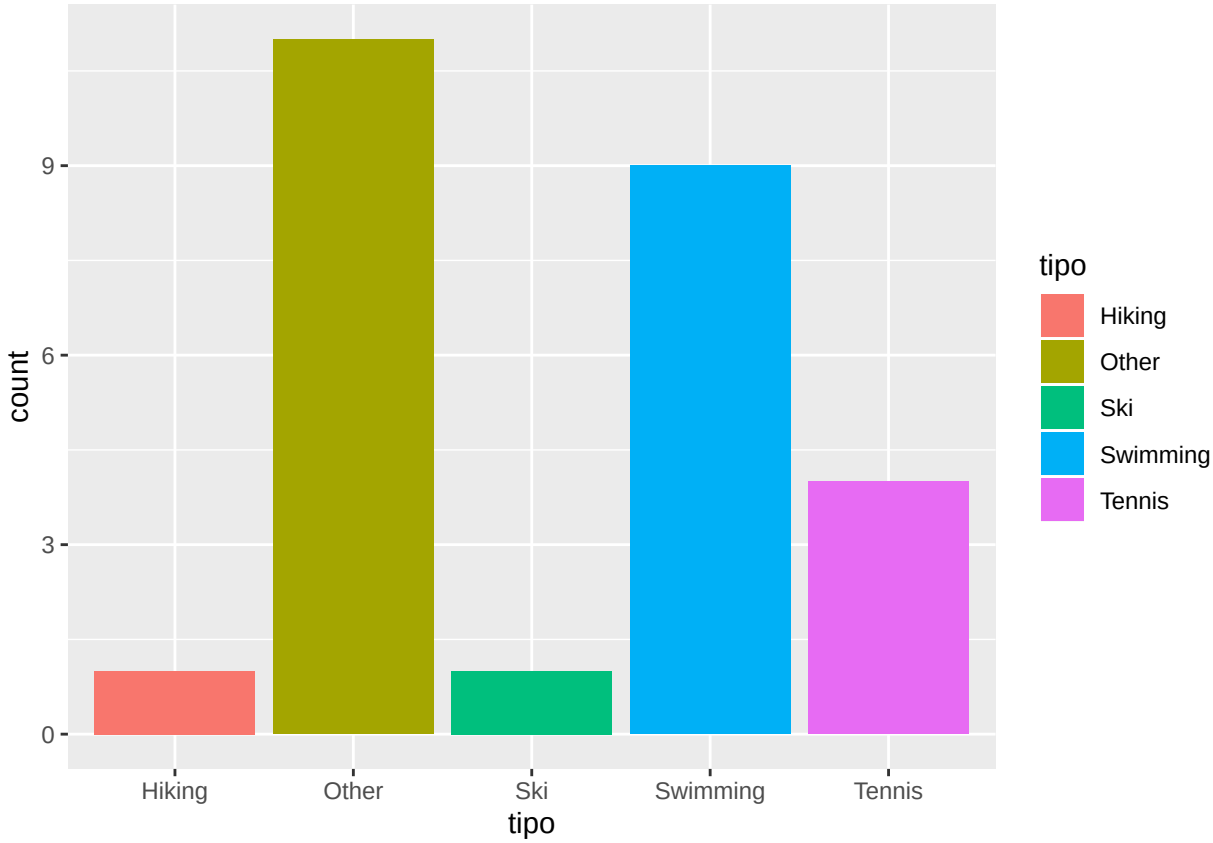
```r
Ams_dep <- filter(act_new, ciudad == "Europe/Amsterdam", tipo == "Dancing")

#Con otro summary he hecho lo mismo para Madrid
summary(Madrid)
```

```
##        de                           a
##  Min.   :2018-11-05 08:30:00   Min.   :2018-11-05 09:15:00
##  1st Qu.:2018-11-28 15:14:00   1st Qu.:2018-11-28 15:34:00
##  Median :2019-03-14 13:44:00   Median :2019-03-14 13:49:00
##  Mean   :2019-03-10 21:18:05   Mean   :2019-03-10 21:36:48
##  3rd Qu.:2019-04-26 07:26:00   3rd Qu.:2019-04-26 07:38:00
##  Max.   :2019-09-23 13:37:00   Max.   :2019-09-23 13:49:00
##
##     ciudad                tipo
##  Length:213         Walking :157
##  Class :character   Running : 24
##  Mode  :character   Other   : 11
##                     Swimming:  9
##                     Pilates :  5
##                     Tennis  :  4
##                     (Other) :  3
```

```r
Mad_dep <- filter(act_new, ciudad == "Europe/Madrid", tipo %in% c("Hiking", "Other", "Ski", "Swimming",

#Aqui he graficado las actividades de cada ciudadad individualmente y he creado
#un histograma para Amsterdam y otro para Madrid
ggplot(data = Ams_dep)+
  geom_bar(mapping = aes(x = tipo), color = "darkblue")
```




```
ggplot(data = Mad_dep)+
  geom_bar(mapping = aes(x = tipo, fill = tipo))
```

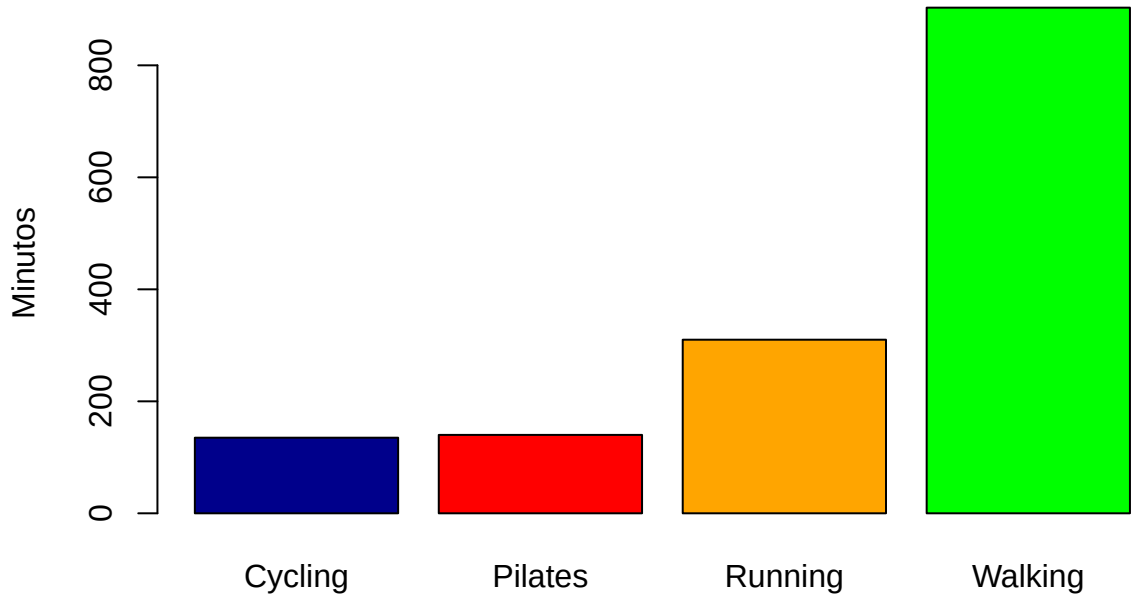



```
#8. Encontrar las fechas en las que se ha practicado bicicleta o pilates en Amsterdam en el año 2019
#he creado otro objeto solo para bicicleta y pilates.
bike_pil <- filter(Amsterdam, tipo %in% c("Cycling", "Pilates"))
print(bike_pil$de)
```

```
## [1] "2019-03-19 07:15:00 UTC" "2019-03-21 06:45:00 UTC"
## [3] "2019-04-06 18:00:00 UTC" "2019-04-07 01:00:00 UTC"
## [5] "2019-04-19 19:30:00 UTC" "2019-07-26 12:00:00 UTC"
## [7] "2019-09-20 07:00:00 UTC" "2019-09-20 08:30:00 UTC"
## [9] "2019-09-20 15:00:00 UTC"
```

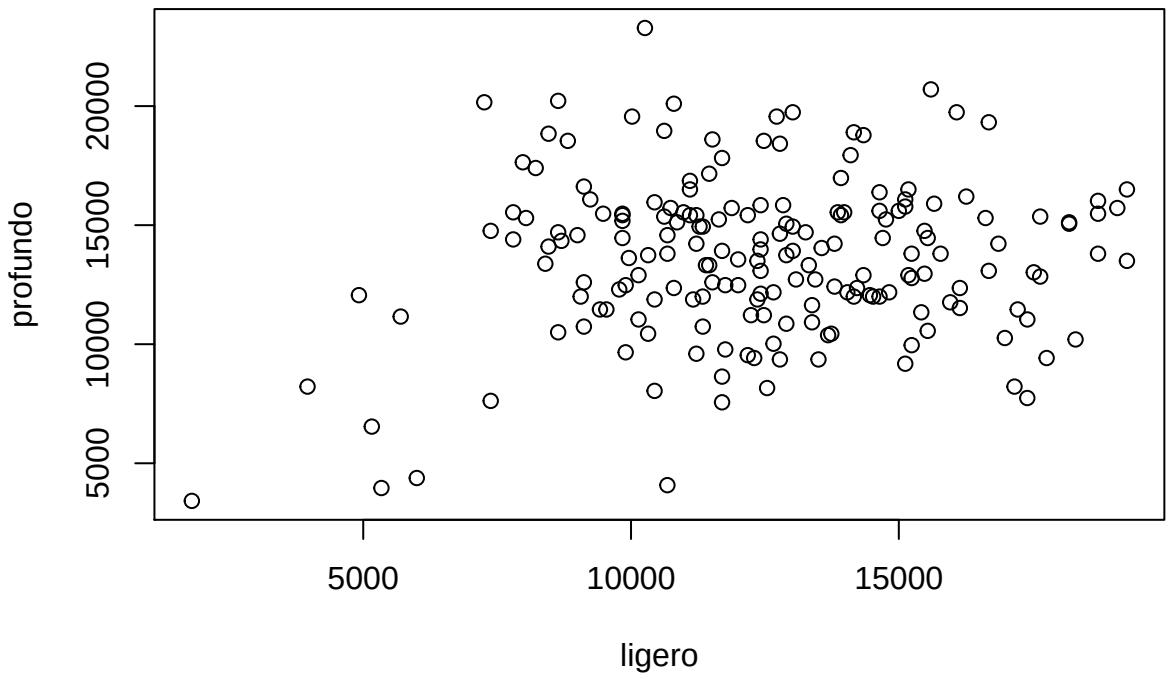
```
#9. Crear una nueva variable dif con los minutos de realización de cada actividad en Amsterdam
# y realizar una representación gráfica de los resultados con plot y determinar que deporte o deportes
# se han practicado durante dos horas o mas

#a partir de aqui he hecho lo mismo pero de diferentes maneras
minutes <- data.frame(Amsterdam$a - Amsterdam$de)
minutes <- rename(minutes, minutes = Amsterdam.a...Amsterdam.de)
Amsterdam <- cbind(Amsterdam, minutes)
Amsterdam <- rename(Amsterdam, minutos = minutes)


#en este punto me ayudo un compañero
Amsterdam$tipo <- as.factor(Amsterdam$tipo)
Amsterdam_min <- Amsterdam %>%
```

```
  group_by(tipo) %>%
  summarize(suma = sum(minutos)) %>%
  filter(suma>=120)

#aqui he hecho un grafico de lo anterior
Amsterdam_min<-as.data.frame(Amsterdam_min)
Amsterdam_min$suma<-as.double(Amsterdam_min$suma)
barplot(Amsterdam_min$suma, names.arg = c("Cycling", "Pilates", "Running", "Walking"), ylab = "Minutos"
```

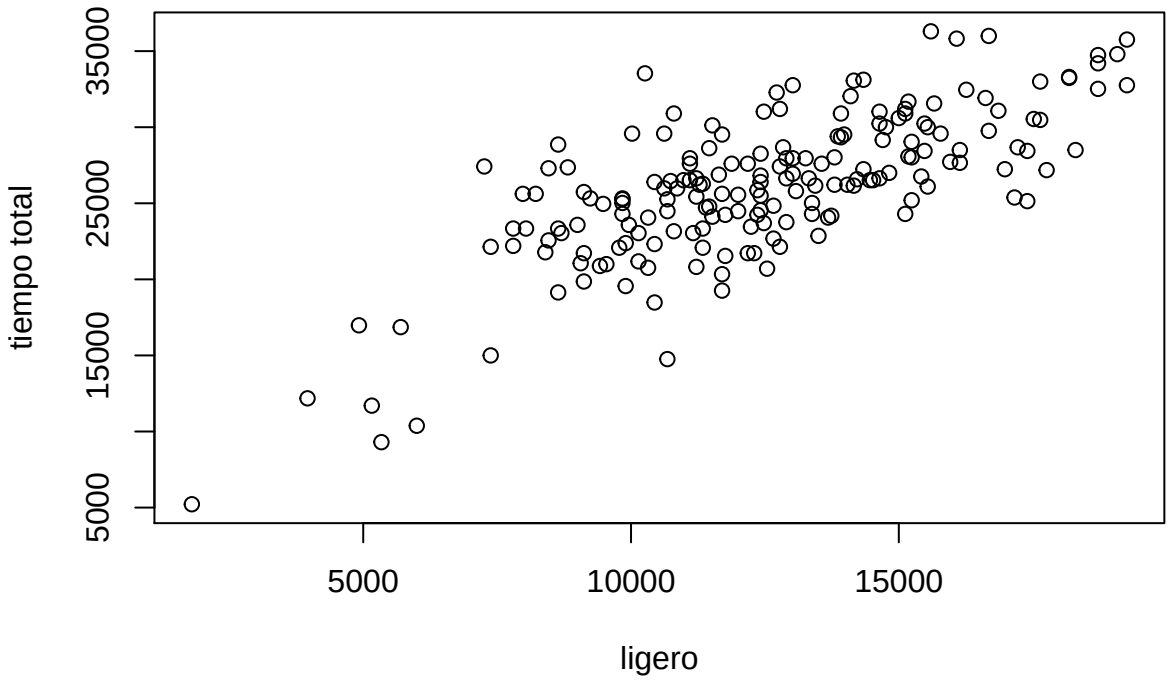



```
#10. Guardar el nuevo dataset en un archivo llamado "act_new.csv"

write_csv(act_new, "act_new.csv")

# 11. Cargar el dataset sleep en un objeto llamado sleep


#12. crear un nuevo data set llamado sleep_new que contenga solo las variables
#que contengan información, que no sean todo cero.

sleep_new <- select(sleep, de, a, `ligero (s)`, `profundo (s)`, `despierto (s)`, despertar, `Duration t


#13. Renombrar las variables de sleep_new a nombres cortos:
```

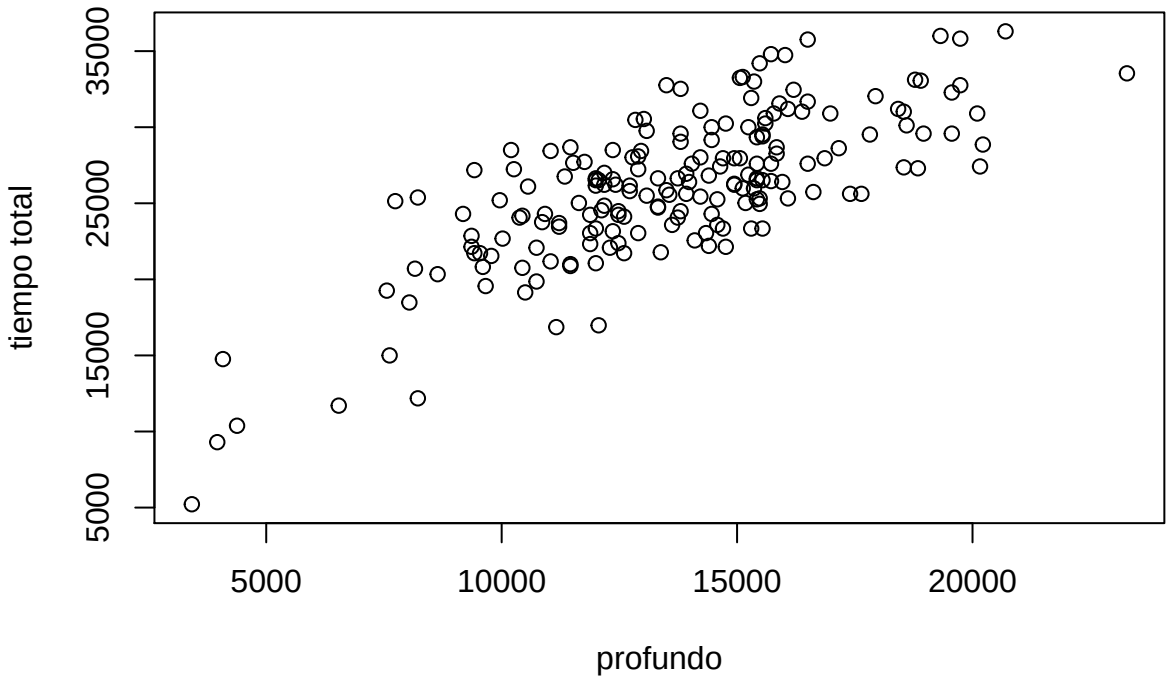
```r
sleep_new<-rename(sleep_new, ligero = `ligero (s)`, profundo = `profundo (s)`, despierto = `despierto (s`

#14. Eliminar todas las filas que contengan algún NA
sleep_new <-na.omit(sleep_new)
# 15. Calcular cuanto tiempo en total se ha dormido cada noche: ligero+profundo
sleep_new <- mutate(sleep_new,  tiempo_total  = ligero  + profundo)
# 16. Visualizacion de la relacion ligero-profundo-total
plot(sleep_new$ligero, sleep_new$profundo, xlab = "ligero", ylab = "profundo")
```




```r
plot(sleep_new$ligero, sleep_new$tiempo_total, xlab = "ligero", ylab = "tiempo total")
```




```
plot(sleep_new$profundo, sleep_new$tiempo_total,  ylab = "tiempo total", xlab = "profundo")
```



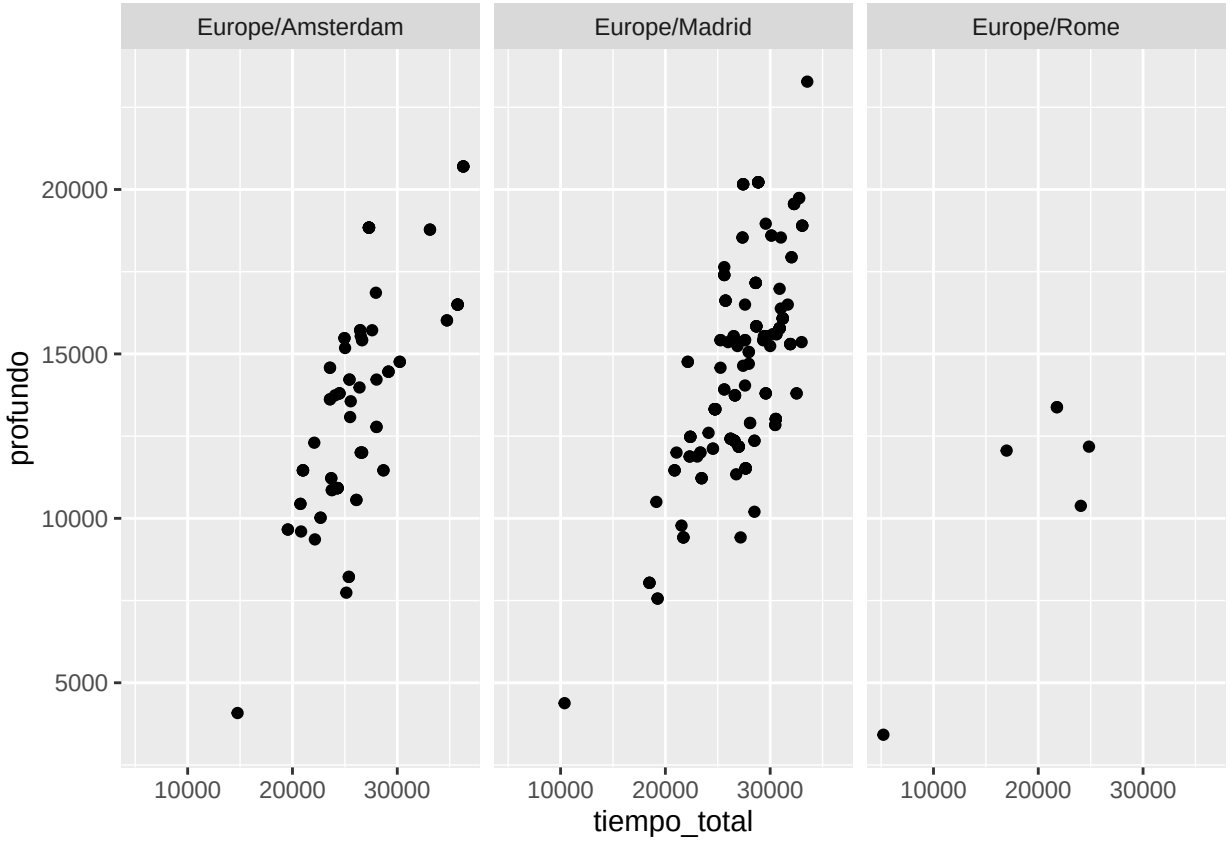

```
# A la vista de los resultados, que tipo de sueño es mas relevante?
# 17. Realizar un analisis de diferencias entre los dos tipos de sueño e interpretar los resultados
# usar la función ICalpha o el 'One sample t-test' de TeachingDemos: t.test()

ICalpha<-function(ModeloA, ModeloB, alfa=0.05)
{
  n<-length(ModeloA)
  diferencias<-ModeloA-ModeloB
  mediad<-mean(diferencias)
  #mediad2<-mean(diferencias^2)
  s<-sqrt(var(diferencias))
  #s<-sqrt(mediad2-mediad^2)
  valort<-qt(alfa/2,n-1,lower.tail = F)
  valor<-valort*s/sqrt(n)
  cotaInf<-mediad-valor
  cotaSup<-mediad+valor
  df<-data.frame(cotaInf, cotaSup)
  return(df)
}

ICalpha(sleep_new$ligero, sleep_new$profundo)
```

```
##     cotaInf   cotaSup
## 1 -1774.672 -527.8845
```

```r
ICalpha(sleep_new$profundo, sleep_new$ligero)
```

```
##    cotaInf  cotaSup
## 1 527.8845 1774.672
```

```r
#A la vista de los datos obtenidos podemos afirmar que es mas relevante
#el sueño profundo ya que tiene un mayor valor para su media aritmetica.
#18. Crear una nueva variable 'ciudad' en sleep_new con la informacion de act_new.

sleep_new$date <- substr(sleep_new$de, 1, 10)
act_new$date<- substr(act_new$de, 1, 10)

sleep_new<-merge(act_new, sleep_new, by="date")
sleep_new <-select(sleep_new,  date, de.x, a.x, ciudad, tipo, ligero, profundo, despierto, duracion_dorm

#19. Representar la relación totalsleep y profundo usando como facetas el factor ciudad

ggplot(data = sleep_new) +
  geom_point(mapping = aes(x = tiempo_total , y = profundo)) +
  facet_grid(~ciudad)
```




```r
#20. Guardar el dataset sleep_new en un archivo "sleep_new.csv"

write_csv(sleep_new, "sleep_new.csv")
```

```
#21. Guardar el proyecto completo. Subir la carpeta del proyecto al campus.
```