Data Programming with R

Isabella Gollini - 12345679

Solutions Assignment 1

General comments:

- Save the data file in the same folder as the `.Rmd` file, so that you don't have to specify the file path that is specific of the computer you are using (and other users would not be able to run your code without changing it).
- There is no need to print the full dataset, it makes the document very hard to read. Showing the structure is sufficient.

Task 1: Manipulation

1.1

```
# Load the data
crime2019 <- read.csv("EurostatCrime2019.csv",
  header = TRUE, # first row contains column names
  row.names = 1) # first column contains country names
```

1.2

The size can be found in a few ways. One way is:

```
dim(crime2019)
```

```
## [1] 41 13
```

or

```
nrow(crime2019)
```

```
## [1] 41
```

```
ncol(crime2019)
```

```
## [1] 13
```

This shows that there are 41 rows and 13 columns in the dataset.

To see how I wrote code which will include the correct number of rows for any dataset, instead of 'hard-coding' the numbers into my `.Rmd` file, have a look at the file to see how I included the number of rows and columns above.

The structure of the dataset can be found by

```
str(crime2019)
```

```
## 'data.frame':    41 obs. of  13 variables:
##  $ Intentional.homicide                                 : num  2.03 0.84 1.27 NA 1.14 0.81 1.48 0.7
##  $ Attempted.intentional.homicide                       : num  3.25 1.93 8.87 NA 0.54 2.4 1.71 0.58
##  $ Assault                                              : num  5.52 43.29 556.36 NA 39.54 ...
##  $ Kidnapping                                           : num  0.14 0.07 NA NA 1.03 0.02 0.91 0.11 I
##  $ Sexual.violence                                      : num  5.38 50.9 77.45 NA 8.64 ...
##  $ Rape                                                 : num  2.69 18.92 33.33 NA 1.87 ...
##  $ Sexual.assault                                       : num  2.69 26.64 44.12 NA NA ...
##  $ Robbery                                              : num  3.42 29.67 140.14 NA 16.9 ...
##  $ Burglary                                             : num  NA 613.2 565.9 NA 79.8 ...
##  $ Burglary.of.private.residential.premises             : num  40.4 99.3 410.1 NA NA ...
##  $ Theft                                                : num  169 1303 1952 NA 474 ...
##  $ Theft.of.a.motorized.land.vehicle                    : num  11.1 44.2 109.8 NA 18.9 ...
##  $ Unlawful.acts.involving.controlled.drugs.or.precursors: num  70.3 494.1 547.7 NA 78.1 ...
```

Dataframe with 41 rows and 13 columns.

This shows that the object is a dataframe. It again repeats the number of rows and columns, and shows that all columns are numerical recordings of crime rates. Some NAs are visible too.

1.3

1.3.(i)

Now remove those four columns. You can call the dataframe something else here. I'm going to call it something shorter than the previous name:

```
crime2 <- subset(crime2019,
                 select = -c(Rape,
                             Sexual.assault))
```

There are lots of ways to do the above operation, some of them easier than others. Let's check that it worked:

```
str(crime2)
```

```
## 'data.frame':    41 obs. of  11 variables:
##  $ Intentional.homicide                                 : num  2.03 0.84 1.27 NA 1.14 0.81 1.48 0.7
##  $ Attempted.intentional.homicide                       : num  3.25 1.93 8.87 NA 0.54 2.4 1.71 0.58
##  $ Assault                                              : num  5.52 43.29 556.36 NA 39.54 ...
##  $ Kidnapping                                           : num  0.14 0.07 NA NA 1.03 0.02 0.91 0.11 I
##  $ Sexual.violence                                      : num  5.38 50.9 77.45 NA 8.64 ...
##  $ Robbery                                              : num  3.42 29.67 140.14 NA 16.9 ...
##  $ Burglary                                             : num  NA 613.2 565.9 NA 79.8 ...
##  $ Burglary.of.private.residential.premises             : num  40.4 99.3 410.1 NA NA ...
##  $ Theft                                                : num  169 1303 1952 NA 474 ...
##  $ Theft.of.a.motorized.land.vehicle                    : num  11.1 44.2 109.8 NA 18.9 ...
##  $ Unlawful.acts.involving.controlled.drugs.or.precursors: num  70.3 494.1 547.7 NA 78.1 ...
```

Good! Another way to do this is to set the columns you want to remove to `NULL`:

```
crime2019$Rape <- NULL
crime2019$Sexual.assault <- NULL
```

Checking its structure:

```
str(crime2019)
```

```
## 'data.frame':    41 obs. of  11 variables:
##  $ Intentional.homicide                                 : num  2.03 0.84 1.27 NA 1.14 0.81 1.48 0.7
##  $ Attempted.intentional.homicide                       : num  3.25 1.93 8.87 NA 0.54 2.4 1.71 0.58
##  $ Assault                                              : num  5.52 43.29 556.36 NA 39.54 ...
##  $ Kidnapping                                           : num  0.14 0.07 NA NA 1.03 0.02 0.91 0.11 I
##  $ Sexual.violence                                      : num  5.38 50.9 77.45 NA 8.64 ...
##  $ Robbery                                              : num  3.42 29.67 140.14 NA 16.9 ...
##  $ Burglary                                             : num  NA 613.2 565.9 NA 79.8 ...
##  $ Burglary.of.private.residential.premises             : num  40.4 99.3 410.1 NA NA ...
##  $ Theft                                                : num  169 1303 1952 NA 474 ...
##  $ Theft.of.a.motorized.land.vehicle                    : num  11.1 44.2 109.8 NA 18.9 ...
##  $ Unlawful.acts.involving.controlled.drugs.or.precursors: num  70.3 494.1 547.7 NA 78.1 ...
```

Good! So that's two different ways. There are many others, e.g., using `[ , ]` notation to select the columns you want to keep.

1.3.(ii)

I can proceed removing the columns similarly to question 1.3.(i):

```
crime2019 <- subset(crime2019,
                    select = -c(Theft,
                                Theft.of.a.motorized.land.vehicle,
                                Burglary,
                                Burglary.of.private.residential.premises))
str(crime2019)
```

```
## 'data.frame':    41 obs. of  7 variables:
##  $ Intentional.homicide                                 : num  2.03 0.84 1.27 NA 1.14 0.81 1.48 0.7
##  $ Attempted.intentional.homicide                       : num  3.25 1.93 8.87 NA 0.54 2.4 1.71 0.58
##  $ Assault                                              : num  5.52 43.29 556.36 NA 39.54 ...
##  $ Kidnapping                                           : num  0.14 0.07 NA NA 1.03 0.02 0.91 0.11 I
##  $ Sexual.violence                                      : num  5.38 50.9 77.45 NA 8.64 ...
##  $ Robbery                                              : num  3.42 29.67 140.14 NA 16.9 ...
##  $ Unlawful.acts.involving.controlled.drugs.or.precursors: num  70.3 494.1 547.7 NA 78.1 ...
```

1.3.(iii)

```
crime2019$Total <- rowSums(crime2019, na.rm = FALSE)
```

Using `na.rm = FALSE` means that when I sum up the values, if one of them is `NA`, then the sum will be still `NA`.

1.4

Now we want to select only those countries which have complete records.

```
checkNA <- crime2019[!complete.cases(crime2019), ]
rownames(checkNA)
```

```
##  [1] "Belgium"                "Bosnia and Herzegovina" "Denmark"
##  [4] "England and Wales"      "Estonia"                "France"
```

```
##  [7] "Hungary"                "Iceland"                "Liechtenstein"
## [10] "Netherlands"            "North Macedonia"        "Northern Ireland (UK)"
## [13] "Norway"                 "Poland"                 "Portugal"
## [16] "Scotland"               "Slovakia"               "Sweden"
## [19] "Turkey"
```

`complete.cases(crime2019)` returns a logical vector showing `TRUE` if all values in the row are available, and `FALSE` if at least one value is missing - `NA`. By using `!` before `complete.cases(crime2019)` I am checking when there are **no** `complete.cases`. So I essentially subset `crime2019` above and only select those rows with a `TRUE` value for `!complete.cases(crime2019)` (that is the same as selecting those rows with a `FALSE` value for `complete.cases(crime2019)`) - those which have missing data.

Another way to answer the same question is by using the `apply` function:

```
checkNA <- apply(crime2019, 1, function(x) any(is.na(x)))
rownames(crime2019)[checkNA]
```

```
##  [1] "Belgium"                "Bosnia and Herzegovina" "Denmark"
##  [4] "England and Wales"      "Estonia"                "France"
##  [7] "Hungary"                "Iceland"                "Liechtenstein"
## [10] "Netherlands"            "North Macedonia"        "Northern Ireland (UK)"
## [13] "Norway"                 "Poland"                 "Portugal"
## [16] "Scotland"               "Slovakia"               "Sweden"
## [19] "Turkey"
```

1.5

Again, there are different ways to answer this question, for example:

```
crime2 <- crime2019[checkNA != 1, ]
str(crime2)
```

```
## 'data.frame':    22 obs. of  8 variables:
##  $ Intentional.homicide                                : num  2.03 0.84 1.14 0.81 1.48 0.76 1.59 0
##  $ Attempted.intentional.homicide                      : num  3.25 1.93 0.54 2.4 1.71 0.58 5.96 2.
##  $ Assault                                             : num  5.52 43.29 39.54 18.06 20.09 ...
##  $ Kidnapping                                          : num  0.14 0.07 1.03 0.02 0.91 0.11 0.02 5
##  $ Sexual.violence                                     : num  5.38 50.9 8.64 21.05 1.94 ...
##  $ Robbery                                             : num  3.42 29.67 16.9 20.56 6.28 ...
##  $ Unlawful.acts.involving.controlled.drugs.or.precursors: num  70.3 494.1 78.1 272.2 117.8 ...
##  $ Total                                               : num  90 621 146 335 150 ...
```

or by using `complete.cases`:

```
crime2019 <- crime2019[complete.cases(crime2019), ]
str(crime2019)
```

```
## 'data.frame':    22 obs. of  8 variables:
##  $ Intentional.homicide                                : num  2.03 0.84 1.14 0.81 1.48 0.76 1.59 0
##  $ Attempted.intentional.homicide                      : num  3.25 1.93 0.54 2.4 1.71 0.58 5.96 2.
##  $ Assault                                             : num  5.52 43.29 39.54 18.06 20.09 ...
##  $ Kidnapping                                          : num  0.14 0.07 1.03 0.02 0.91 0.11 0.02 5
##  $ Sexual.violence                                     : num  5.38 50.9 8.64 21.05 1.94 ...
##  $ Robbery                                             : num  3.42 29.67 16.9 20.56 6.28 ...
##  $ Unlawful.acts.involving.controlled.drugs.or.precursors: num  70.3 494.1 78.1 272.2 117.8 ...
##  $ Total                                               : num  90 621 146 335 150 ...
```

1.6

```
dim(crime2019)
```

```
## [1] 22  8
```

Dataframe with 22 observations (rows) and 8 variables (columns).

Task 2: Analysis

2.1

```
# select the row containing data for Ireland and remove the last column which contains the Total
Ireland <- crime2019["Ireland", -ncol(crime2019)]
# sort the crimes in Ireland in decreasing order
Ireland <- sort(Ireland, decreasing = TRUE)
# select the names of the top three crimes
names(Ireland[1:3])
```

```
## [1] "Unlawful.acts.involving.controlled.drugs.or.precursors"
## [2] "Assault"
## [3] "Sexual.violence"
```

It can also be done in a single row of code:

```
names(sort(crime2019["Ireland", -ncol(crime2019)], decreasing = TRUE)[1:3])
```

```
## [1] "Unlawful.acts.involving.controlled.drugs.or.precursors"
## [2] "Assault"
## [3] "Sexual.violence"
```

The 3 most common crimes in Ireland in 2019 were Unlawful.acts.involving.controlled.drugs.or.precursors, Assault, Sexual.violence.

Look at the Rmd file again to see how I wrote the line above, so that even if the data changes, and I re-run my script, it will still pick out the top 3 crimes, whatever they happen to be in this new dataset.

2.2

```
crime2019["Ireland",]$Assault / crime2019["Ireland",]$Total
```

```
## [1] 0.1605316
```

2.3

```
rownames(crime2019)[which.max(crime2019$Kidnapping)]
```

```
## [1] "Luxembourg"
```

This shows that Luxembourg is the country with the highest record of kidnapping.

2.4

The country with the lowest record of offences was

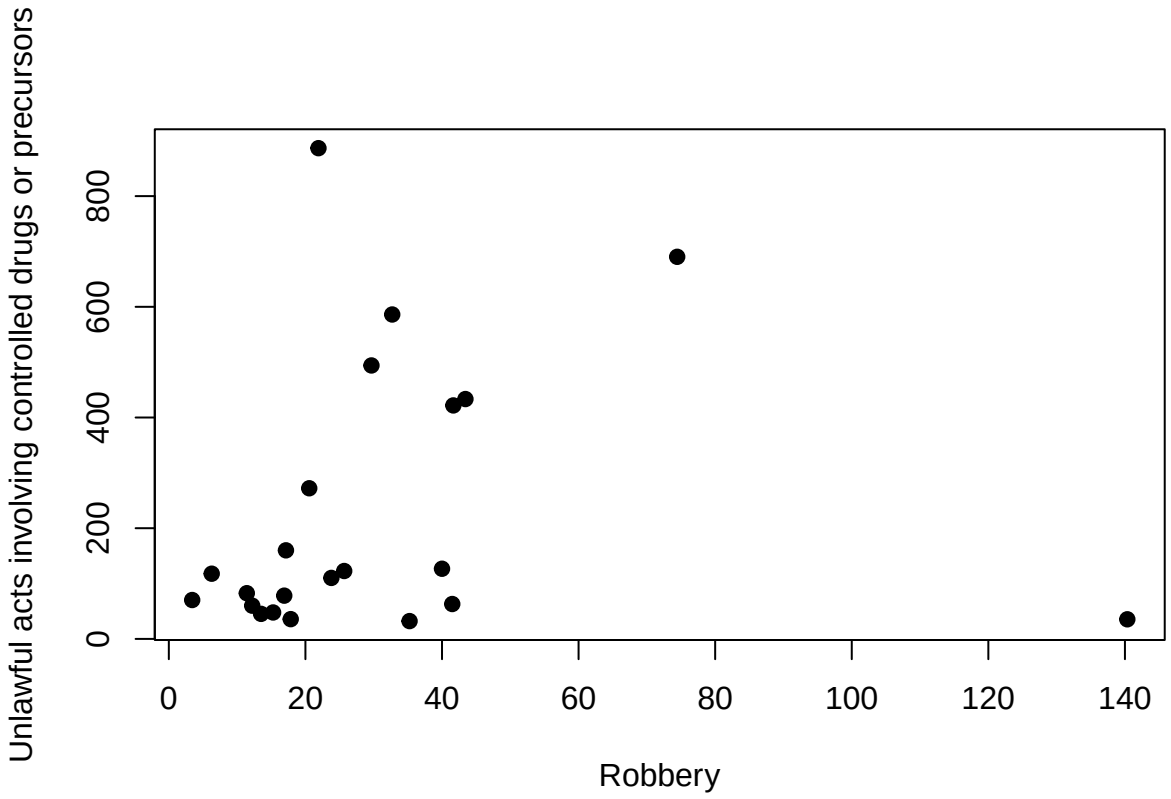
```
rownames(crime2019)[which.min(crime2019$Total)]
```

```
## [1] "Romania"
```

This shows that Romania is the country with the highest overall record of offences.

2.5

```
plot(crime2019$Robbery,
     crime2019$Unlawful.acts.involving.controlled.drugs.or.precursors,
     xlab = "Robbery",
     ylab = "Unlawful acts involving controlled drugs or precursors",
     pch = 19)
```



Task 3: Creativity

This task was up to you. But you must describe your findings, not just produce a plot etc., with no comment on it.