S2S Lab 3 Task Solutions

1 Welcome!

2 Reading in Data

2.1 Setting your working directory

2.2 `read.table()`

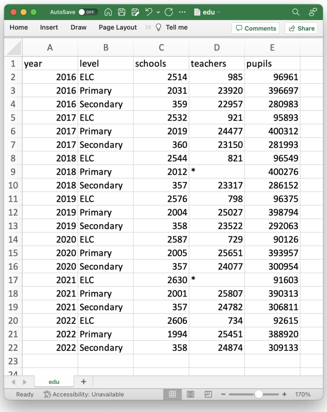
2.3 `read.csv()`

Read the file "*edu.csv*" into R and save it as a data frame called `education`.

This is a data set containing information on the total numbers of pupils and teachers in schools of different education levels in Scotland. The variables included are:

- `"year"`: the year measurements were taken in (2016-2022).
- `"level"`: the level of education measurements were taken from ("ELC", "Primary" or "Secondary").
- `"schools"`: the total number of schools across Scotland in the given year/level combination.
- `"teachers"`: the total number of teachers employed in all the schools in the given year/level combination.
- `"pupils"`: the total number of pupils attending all the schools in the given year/level combination.

If you were to look at the original file "*edu.csv*", you would see something similar to Figure @ref(fig:edu-image). Here, we can see that there are column headings and that there are two missing values denoted by "*".



	A	B	C	D	E
1	year	level	schools	teachers	pupils
2	2016	ELC	2514	985	96961
3	2016	Primary	2031	23920	396697
4	2016	Secondary	359	22957	280983
5	2017	ELC	2532	921	95893
6	2017	Primary	2019	24477	400312
7	2017	Secondary	360	23150	281993
8	2018	ELC	2544	821	96549
9	2018	Primary	2012	*	400276
10	2018	Secondary	357	23317	286152
11	2019	ELC	2576	798	96375
12	2019	Primary	2004	25027	398794
13	2019	Secondary	358	23522	292063
14	2020	ELC	2587	729	90126
15	2020	Primary	2005	25651	393957
16	2020	Secondary	357	24077	300954
17	2021	ELC	2630	*	91603
18	2021	Primary	2001	25807	390313
19	2021	Secondary	357	24782	306811
20	2022	ELC	2606	734	92615
21	2022	Primary	1994	25451	388920
22	2022	Secondary	358	24874	309133

Figure 1: Screenshot of edu.csv file

In order to read "*edu.csv*" into R, we can use the following code.

```
education <- read.csv(file = "edu.csv", na.strings = "*")
```

3 Working With Data

3.1 Checking variable types

What type of variable is schools saved as in the education data frame?

Using the `str()` function shows us that `schools` is saved as an integer variable.

```
str(education)
```

```
## 'data.frame':    21 obs. of  5 variables:
##  $ year    : int  2016 2016 2016 2017 2017 2017 2018 2018 2018 2019 ...
##  $ level   : chr  "ELC" "Primary" "Secondary" "ELC" ...
##  $ schools : int  2514 2031 359 2532 2019 360 2544 2012 357 2576 ...
##  $ teachers: int  985 23920 22957 921 24477 23150 821 NA 23317 798 ...
##  $ pupils  : int  96961 396697 280983 95893 400312 281993 96549 400276 286152 96375 ...
```

Write some code to change the variables year and level in education to be factor variables.

```
education$year <- factor(x = education$year,
                         levels = c("2016", "2017", "2018", "2019",
                                    "2020", "2021", "2022"))

education$level <- factor(x = education$level,
                          levels = c("ELC", "Primary", "Secondary"))
```

3.2 Dealing with NA values

Which rows in education have missing values?

Using `complete.cases()` shows us that rows 8 and 16 of `education` are incomplete and therefore contain `NA` values.

```
complete.cases(education)
```

```
##  [1]  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE FALSE  TRUE  TRUE  TRUE  TRUE
## [13]  TRUE  TRUE  TRUE FALSE  TRUE  TRUE  TRUE  TRUE  TRUE
```

Write code to remove all rows in education which contain NA values.

Incomplete observations can be removed from `education` using any of the following lines of code.

```
na.omit(education)
education[complete.cases(education), ]
education[!is.na(education$teachers), ]
```

3.3 Sorting data frames

What is the largest value for pupils from the education data frame?

```
sort(education$pupils, decreasing = TRUE)[1]
```

```
## [1] 400312
```

Write code to sort the observations from education in decreasing order of the number of pupils.

We need to include the argument `decreasing = TRUE` within the function `order()` so that the observations are ordered from largest number of pupils to the smallest number of pupils. We can use the `order()` function within square brackets to show all variables in the data frame in order of decreasing number of pupils.

```
education[order(education$pupils, decreasing = TRUE), ]
```

```
##    year     level schools teachers pupils
## 5  2017   Primary    2019    24477 400312
## 8  2018   Primary    2012       NA 400276
## 11 2019   Primary    2004    25027 398794
## 2  2016   Primary    2031    23920 396697
## 14 2020   Primary    2005    25651 393957
## 17 2021   Primary    2001    25807 390313
## 20 2022   Primary    1994    25451 388920
## 21 2022 Secondary     358    24874 309133
## 18 2021 Secondary     357    24782 306811
## 15 2020 Secondary     357    24077 300954
## 12 2019 Secondary     358    23522 292063
## 9  2018 Secondary     357    23317 286152
## 6  2017 Secondary     360    23150 281993
## 3  2016 Secondary     359    22957 280983
## 1  2016       ELC    2514      985  96961
## 7  2018       ELC    2544      821  96549
## 10 2019       ELC    2576      798  96375
## 4  2017       ELC    2532      921  95893
## 19 2022       ELC    2606      734  92615
## 16 2021       ELC    2630       NA  91603
## 13 2020       ELC    2587      729  90126
```

3.4 Subsetting

Write some code to subset `education` to show the number of schools that have a collective total of more than 310,000 pupils in the years 2020, 2021 or 2022.

The data frame that we want to subset is `education`, so this is what we'll feed in to the argument `x =`.

Since the question asks us to look for a collective total of more than 310,000 pupils, this means we want to only see the rows where the value for `pupils` is greater than 310,000. We also only want to see rows from the years 2020, 2021 or 2022. Because `year` is a factor, we need to specify each level that we are interested in. This means that we are looking for rows in which `pupils > 310000` AND `year == "2020"` or `year == "2021"` or `year == "2021"`. This is quite a lengthy logical statement in the following code.

The question also asks us to only show the number of schools for which these statements are true i.e. the column `schools`. To do this, we simply feed this variable to the `select =` argument.

```
subset(x = education,
       subset = pupils > 310000 & year == "2020" |
         pupils > 310000 & year == "2021" |
         pupils > 310000 & year == "2022",
       select = schools)
```

A way we can shorten the logical statement in the `subset =` argument is to use the operator `%in%`. This will search for values in a vector and return the rows in which any of these values appear.

```
subset(x = education,
       subset = pupils > 310000 & year %in% c("2020", "2021", "2022"),
       select = schools)
```

```
##    schools
## 14    2005
## 17    2001
## 20    1994
```

3.5 Sumamrising data

What is the mean total number of teachers in primary schools across all years?

In order to find this value we want to use the function `tapply()`. `teachers` is the column we want to calculate the `mean` for, but make sure to split this by the different levels in the `level` column.

`teachers` contains some `NA` values, which when passed to the function `mean` will return another `NA` value unless you provide to `tapply()` the additional argument `na.rm = TRUE`. This tells R to ignore the `NA` values when calculating the mean and only use those rows which have a numerical value.

```
tapply(X = education$teachers, INDEX = list(education$level), FUN = mean, na.rm = TRUE)
```

```
##        ELC    Primary  Secondary
##   831.3333 25055.5000 23811.2857
```

3.6 Creating variables

In the `education` data frame, create a new variable called `ratio` which calculates the pupil to teacher ratio in each level of education. That is,

$$\text{ratio} = \frac{\text{puils}}{\text{teachers}}$$

```
education$ratio <- education$pupils/education$teachers
```

3.7 Merging data frames

The file *class.csv* contains information on the average primary class size in the years 2016 - 2022. Read this file into R and save it as a data frame called `class`.

Merge the information from the data frames `education` and `class` together into a new data frame called `primary`, showing all variables from `education` and the average class size for primary schools only. Look carefully at which row names these two data frames have in common.

To read the file *class.csv* in to R, we can use the following code.

```
class <- read.csv(file = "class.csv")
```

In order to merge the two data frames, we want to use the function `merge()`. The data frame we provide to the argument `x =` is `education` and the data frame for the `y =` argument is `class`.

Because we want to match up the rows with the same year and the same level of education, we need to give the argument `by =` a vector of these two variables. We can use the argument `by =`, rather than `by.x =` and `by.y =`, because the columns have the same names in both data frames.

Finally, since we only want to show the rows for primary schools, we can specify `all.y =TRUE`. This will keep all the rows from the second data frame, `class`, and delete the rows from the first data frame which don't have a matching row in the second. For example, because there is no information on the average class size in secondary schools in 2016 in `class`, this row from `education` will not appear in `primary`.

```
primary <- merge(x = education, y = class, by = c("year", "level"), all.y = TRUE)
primary
```

```
##   year   level schools teachers pupils    ratio size
## 1 2016 Primary    2031    23920 396697 16.58432 23.5
## 2 2017 Primary    2019    24477 400312 16.35462 23.5
## 3 2018 Primary    2012       NA 400276       NA 23.5
## 4 2019 Primary    2004    25027 398794 15.93455 23.5
## 5 2020 Primary    2005    25651 393957 15.35835 23.1
## 6 2021 Primary    2001    25807 390313 15.12431 23.2
## 7 2022 Primary    1994    25451 388920 15.28113 23.3
```

4 Functions

4.1 Probability functions

Choose the correct function and complete the code for the following scenarios.

- You want to construct a 90% confidence interval so need to know the 95th quantile of the standard normal distribution.

```r
qnorm(p = 0.95, mean = 0, sd = 1)
```

- How would you find the value of x such that $\mathbb{P}(X \leq x) = 0.45$, where $X \sim N(100, 4^2)$?

```r
qnorm(p = 0.45, mean = 100, sd = 4)
```

- You want to know the proportion of the $N(0, 2^2)$ distribution that lies below -2. That is, $\mathbb{P}(X \leq -2)$, where $X \sim N(0, 2^2)$.

```r
pnorm(q = -2, mean = 0, sd = 2)
```

4.2 Flow control

Complete the following code to sum together the numbers 1 to 12.

We start by creating the vector `sum` to which each value 1, 2, . . . , 12 can be added. Initially it needs to take the value 0.

Within the function `for()`, we want `i` to, in turn, take each value 1, 2, . . . , 12, so we need to provide a vector of these values (`1:12`). `sum` should then be updated each time `i` takes a new value, by adding it on to the old value of `sum`.

```r
sum <- 0

for(i in 1:12){
  sum <- sum + i
}
```

This for loop starts with `sum` having the value 0. It will first assign 1 to `i` and execute the code `sum <- 0 + 1`, meaning `sum` now has the value 1.

- `i` will then be updated to take the value 2 and the for loop will run the code `sum <- 1 + 2` i.e. `sum` has the value 3.
- `i` will then be updated to take the value 3 and the for loop will run the code `sum <- 3 + 3` i.e. `sum` has the value 6.
- $\vdots$

This repeats until finally `i` is assigned the value 12 and the for loop updates `sum` for the last time.

Using the above code, what is the value of $1 + 2 + 3 + ... + 12$?

```r
sum <- 0

for(i in 1:12){
  sum <- sum + i
}

sum
```

```
## [1] 78
```

Running the code above updates `sum` several times until it takes the value 78. Therefore the value of $1 + 2 + 3 + ... + 12 = 78$.

What would the value of `y` be after running the following if statement? Try to answer without running the code yourself.

```
x <- -4

if(x > 0){
  y <- x^2
} else {
  y <- -(x^2)
}
```

Running the code above updates the value of `y` to be -16.

4.3 Creating functions