Class2_Exercise

FedericaFusi

12/23/2020

Class 2 - Exercise

Instructions: submit your R script file reporting your answers and results as comments.

A couple of new functions

It is very helpful to know how to create a 'fake' dataset in R to experiment with new functions and packages.

- When you don't know how a function or a code work, try it on a small fake dataset and 'debunk' each step.
- It is also helpful to show others how a code works.

For this scope, check out the **sample** function. You can use *help* or google it.

As mentioned in class, when learning about a new function, you generally want to retrieve three pieces of information:

- **Description** wthat the function does
- **Usage** how you are expected to write the function
- **Arguments** what each part of the function does

Once you read the help page, you should play with the function to fully understand how it works. Use small numbers and examples that make it easier to see the mechanics.

1) After reading the help page, try running the following codes and make notes of what each argument does. If you cannot fully understand try to change the value assigned (e.g., x = 6 instead than x = 5).

```
sample(x = 5)

sample(x = 5, size = 2)

sample(x = 1:100, size = 4)

sample(x = 1:10, size = 6)

sample(x = 1:5, size = 6, replace = F)

sample(x = 1:5, size = 6, replace = T)
```

Note: you do not need to write the argument of a function if you respect their order as described. You can easily see this by running these codes:

Another interesting function is **set.seed** which makes your randomly generated numbers reproducible. If you set the same seed as someone else and then use the sample function, you both will get the same random numbers (try it a couple of times!)

```
set.seed(23)
sample(5)

set.seed(34)
sample(5)

set.seed(23)
sample(5)
```

RandomNames package

We are also using the Randomnames package which will allow us to generate some random individual informations such as, gender, ethnicity and first/last name.

Look up the online documentation of the package by googling the name or by clicking here.

All we need is the **randomNames* function described in the documentation.

```
names = randomNames(4)

names

class(names)
```

De-bugging

It happened to me a couple of times to receive a warning after installing the randomNames package (e.g., "lazy-load database"). If you have issues after installing the randomNames package, you should try to restart RStudio.

You can simply do so by using the following command, which re-start RStudio without closing any of your work-in-progress files and datasets.

```
.rs.restartR()
```

Creating the dataset

Now that we are all set, we are going to create a fake dataset about unemployment.

We will use 434 as the seed for generating the same random numbers as a class.

2) Variable 1: Set your seed and generate a random of sample of 40 observations ranging from 0 to 255. This variable represents how many days each individuals was unemployed in the past full year. Name it accordingly.

3) Variable 2: Our second variable is the number of job applications submitted by each individual. Again, set your seed at 434. Use sample to generate a random variable ranging from 1 to 25. Name it accordingly.

4) Variable 3: Our third variable is a dummy which indicates the gender of the individual. Gender variables are generaly coded as 0 = male and 1 = female. Name the variable accordingly.

Hint: You can use the sample function to do this but you will need to include the "replace" argument.

Hint 2: When setting your x argument think about how many values the variable should take (i.e., only 0 or 1).

5) Finally, we use the randomNames function from the randomNames package to generate 40 random names. Since we have created a gender variable, we want to match name and gender. Name the variable acccordingly.

```
#A hint:

..... = randomNames(....., gender = .....)
```

Questions

Now that you have created all your variables, do the following activities:

6) Combine them in one dataframe.

7) Check the dimensions of the dataframe (# of columns and # of rows)

8) Check the class of each columns. Fix them if the class is not appropriate (e.g., we don't want to keep factors)

9) Run some basic statistics to explore

- Average number of unemployment day and standard deviation
- Minimum and maximum number of applications submitted as well as average number
- Number of male and female applicants included in the sample
- Average number of submitted applications for female applicants
- Maximum number of employment days for female and male applicants

10) Think about policy! While this is a fake dataset, let's start thinking as policy makers and public managers would. Based on your results, what would you tell to a policy maker planning a new policy intervention on unemployment? If needed, you can run more descriptive statistics