African Center for Applied Machine Learning, Artificial Intelligence, and Policy: R Training Workshop Part 1

Eric Asare,Ph.D.

11/8/2020

Tidyverse Package

The tidyverse package undoubtedly is the most important package for data analysis in R. It assemblies all the packages you need for your data-analysis. In this workshop we will go through a series of data transformations on an in built R dataset called msleep from the ggplot2 package. Parameter estimation and reporting will be in workshop part 2. The # sign is used to write notes on what your code means, and since this is a comment it will not run in R. It makes it easier to understand your code when you revisit it say in month time. Also, you will be seeing a lot of <- which is used to assign values to variables. Please copy the phrase "install.packages("tidyverse")", paste it in the R console and run it; this will make sure the latest version of the package is installed. But, I have put it in the script to document it.

```
#install tidyverse to allow me use all the other data analysis tools
#install.packages("tidyverse") #installs tidyverse
library("tidyverse") # activate the package so we can use it
```

```
## -- Attaching packages --------------------------------- tidyverse 1.3.0 --

## v ggplot2 3.3.2     v purrr   0.3.4
## v tibble  3.0.3     v dplyr   1.0.2
## v tidyr   1.1.2     v stringr 1.4.0
## v readr   1.3.1     v forcats 0.5.0

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

In built data

We are going to use the inbuilt msleep data from ggplot2. GGPLOT2 is a package in tidyverse. You are going to see lots of %>%. The %>% are called pipes and it links separate data analysis activities together. For instance, below, we are going to read in the data (first activity) and then view it (second activity) all linked through %>%. The glimpe function is in dplyr also a package in tidyverse that allows us to see at a glance the data types for each column in our dataset.

```
workshop_data <- msleep # i have assigned the msleep data in ggplot2 to workshop_data
workshop_data %>% #first activity
  glimpse() #second activity
```

```
## Rows: 83
## Columns: 11
## $ name         <chr> "Cheetah", "Owl monkey", "Mountain beaver", "Greater s...
## $ genus        <chr> "Acinonyx", "Aotus", "Aplodontia", "Blarina", "Bos", "...
## $ vore         <chr> "carni", "omni", "herbi", "omni", "herbi", "herbi", "c...
## $ order        <chr> "Carnivora", "Primates", "Rodentia", "Soricomorpha", "...
## $ conservation <chr> "lc", NA, "nt", "lc", "domesticated", NA, "vu", NA, "d...
## $ sleep_total  <dbl> 12.1, 17.0, 14.4, 14.9, 4.0, 14.4, 8.7, 7.0, 10.1, 3.0...
## $ sleep_rem    <dbl> NA, 1.8, 2.4, 2.3, 0.7, 2.2, 1.4, NA, 2.9, NA, 0.6, 0....
## $ sleep_cycle  <dbl> NA, NA, NA, 0.1333333, 0.6666667, 0.7666667, 0.3833333...
## $ awake        <dbl> 11.9, 7.0, 9.6, 9.1, 20.0, 9.6, 15.3, 17.0, 13.9, 21.0...
## $ brainwt      <dbl> NA, 0.01550, NA, 0.00029, 0.42300, NA, NA, NA, 0.07000...
## $ bodywt       <dbl> 50.000, 0.480, 1.350, 0.019, 600.000, 3.850, 20.490, 0...
```

Data Wrangling

Data wrangling is a data-science term for all the series of activities you will perform on your raw and sometimes messy dataset to transform it into a clean and tidy dataset for analysis. Today we will go through the three main data wrangling essentials for cleaning data.

They are: 1. Data Selection: Selecting a sub-section of columns from the dataset. 2. Data Filtering: Selecting a sub-section of rows from dataset. 3. Data Mutating: Creating new columns from existing columns based on a formula.

We will just introduce you to some of the tools in tidyverse, in particular the dplyr package, for data wrangling. I think the dplyr package is the most important tool for data wrangling.

Data Selection

In this section we are going to select some variables from the original dataset (workshop_data) and name it as data1

```
#A1. Selecting columns with the select() function from dpylr and store the new data as data1
data1 <- workshop_data %>% #first step data
  select(name, genus, sleep_total, awake)  #second is selection

data1 %>%
  glimpse() #3 see that data
```

```
## Rows: 83
## Columns: 4
## $ name        <chr> "Cheetah", "Owl monkey", "Mountain beaver", "Greater sh...
## $ genus       <chr> "Acinonyx", "Aotus", "Aplodontia", "Blarina", "Bos", "B...
## $ sleep_total <dbl> 12.1, 17.0, 14.4, 14.9, 4.0, 14.4, 8.7, 7.0, 10.1, 3.0,...
## $ awake       <dbl> 11.9, 7.0, 9.6, 9.1, 20.0, 9.6, 15.3, 17.0, 13.9, 21.0,...
```

Data filtering

Here we are going to keep on a certain subset of our original dataset rows based on the condition that we keep only the rows in which the values in the bodywt column is greater than 4. This is an example. We could also filter based on other conditions or a combination of conditions. We have introduced the summary() function to allow us to calculate the summary statistics of the new variables in our new dataset.

```r
data_filter <- workshop_data %>% #first step data
  filter(bodywt > 4) #second is selection

data_filter %>%
  glimpse() #3 see that data
```

```
## Rows: 31
## Columns: 11
## $ name         <chr> "Cheetah", "Cow", "Northern fur seal", "Dog", "Roe dee...
## $ genus        <chr> "Acinonyx", "Bos", "Callorhinus", "Canis", "Capreolus"...
## $ vore         <chr> "carni", "herbi", "carni", "carni", "herbi", "herbi", ...
## $ order        <chr> "Carnivora", "Artiodactyla", "Carnivora", "Carnivora",...
## $ conservation <chr> "lc", "domesticated", "vu", "domesticated", "lc", "lc"...
## $ sleep_total  <dbl> 12.1, 4.0, 8.7, 10.1, 3.0, 5.3, 10.0, 3.9, 2.9, 3.1, 1...
## $ sleep_rem    <dbl> NA, 0.7, 1.4, 2.9, NA, 0.6, 0.7, NA, 0.6, 0.4, 1.1, 0....
## $ sleep_cycle  <dbl> NA, 0.6666667, 0.3833333, 0.3333333, NA, NA, NA, NA, 1...
## $ awake        <dbl> 11.90, 20.00, 15.30, 13.90, 21.00, 18.70, 14.00, 20.10...
## $ brainwt      <dbl> NA, 0.4230, NA, 0.0700, 0.0982, 0.1150, NA, 4.6030, 0....
## $ bodywt       <dbl> 50.000, 600.000, 20.490, 14.000, 14.800, 33.500, 4.750...
```

```r
# the summary fucntion helps us to view the summary stats of the new dataset
data_filter %>%
  summary() #3 see that data
```

```
##      name              genus               vore              order
##  Length:31          Length:31          Length:31          Length:31
##  Class :character   Class :character   Class :character   Class :character
##  Mode  :character   Mode  :character   Mode  :character   Mode  :character
##
##
##
##
##  conservation        sleep_total       sleep_rem      sleep_cycle
##  Length:31          Min.   : 1.900   Min.   :0.100   Min.   :0.3333
##  Class :character   1st Qu.: 3.850   1st Qu.:0.600   1st Qu.:0.4417
##  Mode  :character   Median : 8.000   Median :1.000   Median :0.6667
##                     Mean   : 7.519   Mean   :1.371   Mean   :0.7697
##                     3rd Qu.:10.050   3rd Qu.:1.500   3rd Qu.:0.9500
##                     Max.   :18.100   Max.   :6.100   Max.   :1.5000
##                                      NA's   :10      NA's   :20
##      awake          brainwt           bodywt
##  Min.   : 5.90   Min.   :0.0250   Min.   :   4.23
##  1st Qu.:13.95   1st Qu.:0.1150   1st Qu.:  22.86
##  Median :16.00   Median :0.1790   Median :  62.00
##  Mean   :16.48   Mean   :0.7377   Mean   : 443.30
##  3rd Qu.:20.15   3rd Qu.:0.4230   3rd Qu.: 180.16
##  Max.   :22.10   Max.   :5.7120   Max.   :6654.00
##                  NA's   :10
```

Data Mutate

Here are going to create another column, the square of bodywt, from our existing column (bodywt). Again we have introduced another function called View(), which helps us to open the new data in a new window.

```r
# Mutate
data_mutate <- workshop_data %>% #first step data
  mutate(bodywt_sq = bodywt * bodywt) %>%
  select(bodywt, bodywt_sq) # we select only the original and new column to view

data_mutate %>%
  View()
```

Putting it all together

In a typical data cleaning workflow we would normally do the above steps all chained together. This is where tidyverse comes in handy. It automates all our work for us so that in the future we can just work on the script to produce new results easily. We will be using this strategy in our future workshops. The goal is to make data analysis as efficient and enjoyable as possible!

```r
Final_data <- workshop_data %>% #first step data
  select(name, genus, sleep_total, awake, bodywt) %>% # sel variables
  filter(sleep_total > 3) %>% #filter rows based to satisfy this condition sleep_total > 20
  mutate(bodywt_sq = bodywt * bodywt,
         bodywt_cube = bodywt * bodywt * bodywt)
```

Conclusion

We have gone through essential skills that would allow for efficient data wrangling in R. In the future workshop II we will add two other essential skils, model estimation and also reporting. The goal is to give you a taste of what R can do. In the future will be organising tailored workshops on varied topics to make you an expert user in R. We will also provide you with certificates and a way to show your R experience to potential employers through Github.