Worksheet 02 - Intro to R programming - NCBS MSc WL (Answers)

Akshay Surendra, Anand M Osuri

26 October 2020

Problem set

0. Installing a package. We've so far used two packages `dplyr`,`readr` and `magrittr` - but we will be using a few more, all linked to each other. Together, they are all wrapped in a super package called `tidyverse`. Going forward, we can simply load this `tidyverse` instead of all the other packages individually. For this question, install the `tidyverse` package from CRAN (check that you have a stable internet connection - this might take a while) and make sure dependencies are also installed. Check if tidyverse is installed by loading the package and running the function `tidyverse_packages()`

```
# install.packages("tidyverse", dependencies = TRUE)
library(tidyverse)
tidyverse_packages()
```

```
##  [1] "broom"      "cli"        "crayon"     "dbplyr"     "dplyr"
##  [6] "forcats"    "ggplot2"    "haven"      "hms"        "httr"
## [11] "jsonlite"   "lubridate"  "magrittr"   "modelr"     "pillar"
## [16] "purrr"      "readr"      "readxl"     "reprex"     "rlang"
## [21] "rstudioapi" "rvest"      "stringr"    "tibble"     "tidyr"
## [26] "xml2"       "tidyverse"
```

Field data is usually large and messy, and can be parsed in many ways to give different insights. For this problem set, we will use data that is associated with a paper by Karkarey et al (2017) Alternative reproductive tactics and inverse size-assortment in a high-density fish spawning aggregation. Science runs largely on public money and always on trust and thus, sharing data with tax payers and fellow scientists are, respectively, the right thing to do. One place to access a lot of such datasets associated with published journal articles is Data Dryad.

The data set from this dryad location has been sent to you (`male_activitybudget.csv`) over email. Reading the paper and then going over the data is recommended to get a feel for the data, although not necessary to answer the following questions.

1. load the data frame as a tibble into RStudio by first setting the working directory and print the first three rows of the dataset in the console. Can you print the last 6 rows of the tibble using a function and not indexing?

```
# setwd("D:/2020_IntroToR_NCBS/IntroR_2020_NCBS_content/Worksheet 02/")
# replace above with your path

dat <- read_csv("male_activitybudget.csv")

tail(dat) # built-in function
```

```
## # A tibble: 6 x 6
##    Year ID     habitat   sec State       new.state
##   <dbl> <chr>  <chr>   <dbl> <chr>       <chr>
## 1  2013 908_sl Slope       4 Interaction Aggression
## 2  2013 908_sl Slope      13 Movement    Rove
## 3  2013 908_sl Slope      28 Movement    Rest
## 4  2013 908_sl Slope       6 Interaction Female
## 5  2013 908_sl Slope       3 Movement    Rove
## 6  2013 908_sl Slope      10 Interaction Aggression
```

```
a <- (nrow(dat)-5)
b <- nrow(dat)
dat[a:b,] # more explicitly supply
```

```
## # A tibble: 6 x 6
##    Year ID     habitat   sec State       new.state
##   <dbl> <chr>  <chr>   <dbl> <chr>       <chr>
## 1  2013 908_sl Slope       4 Interaction Aggression
## 2  2013 908_sl Slope      13 Movement    Rove
## 3  2013 908_sl Slope      28 Movement    Rest
## 4  2013 908_sl Slope       6 Interaction Female
## 5  2013 908_sl Slope       3 Movement    Rove
## 6  2013 908_sl Slope      10 Interaction Aggression
```

2. Identify the column names that begin with the letter **s** using code completion. Select those columns (i) without using `dplyr` functions (ii) using `dplyr` functions without `%>%` operator (iii) using `dplyr` functions AND `%>%` operator; save each of these to new objects. Check whether there are any differences between them.

```
names(dat)
```

```
## [1] "Year"      "ID"        "habitat"   "sec"       "State"     "new.state"
```

Two columns begin with **s**, `State` and `sec` - *either (or both) are valid answers*

```
#(i)

char1 <- substr(x = names(dat),start = 1,stop = 1)
char1_index <- which(char1=="s" |  char1=="S") # either uppercase OR lowercase

s_columns_1 <- dat[,char1_index]

#(ii)
s_columns_2 <- select(.data = dat, starts_with(match = "S", ignore.case = TRUE))
```

```
# check help ?select for starts_with() and other modifier functions

#(iii)
s_columns_3 <- dat %>% select(starts_with(match = "S", ignore.case = TRUE))

identical(s_columns_1,s_columns_2)
```

```
## [1] TRUE
```

```
identical(s_columns_1,s_columns_3)
```

```
## [1] TRUE
```

```
# checking if all three vectors (objects) are identical, pairwise
# if yes, TRUE else FALSE
```

3. Create a new time using filter to retain observations where *State* is `Movement` and `new state` is `Rest`. In this object, what is the mean and median time spent by each individual?

```
newtime1 <- dat %>% filter(State=="Movement",new.state == "Rest")
# alternatives:
# newtime2 <- dat %>% filter(State=="Movement" & new.state == "Rest")
# newtime3 <- filter(.data = dat, State=="Movement",new.state == "Rest")
# newtime4 <- dat[which(dat$State=="Movement" & dat$new.state == "Rest"),]

newtime1 %>%
  group_by(ID) %>%
  summarise(medianval = median(sec),
            meanval = mean(sec)) %>%
  head()
```

```
## # A tibble: 6 x 3
##   ID      medianval meanval
##   <chr>       <dbl>   <dbl>
## 1 1001_sh      14      14
## 2 1002_sh      15.5    35.5
## 3 1002_sl       9      17
## 4 1003_sh      13.5    18.2
## 5 1003_sl      41      35
## 6 1004_sh      25      30.2
```

4. In the original dataset, identify the individual that was observed the longest (hint: `sum()` is an in-built function in R, `group_by()` may help)

```
dat %>%
  group_by(ID) %>%
  summarise(total_time = sum(sec)) %>%
  arrange(desc(total_time)) %>% # arranged in descending order
  pull(ID) %>% # pulling out the fish ID column
  .[1] # equivalent to storing above lines in a vector and calling vector[1]
```

```
## [1] "1006_sl"
```

```
# for the first element
```

5. In the original dataset, identify which individual spends most time roving (hint: we only care about rove state, all else can be removed for this Q)

```
dat %>%
  filter(new.state == "Rove") %>%
  group_by(ID) %>%
  summarise(total_rovetime = sum(sec)) %>%
  arrange(desc(total_rovetime)) %>%
  pull(ID) %>%
  .[1]
```

```
## [1] "905_sl"
```

6. In the original dataset, did the researchers spend more total time observing fish in 2013 or 2014? Can you substantiate that with numbers computed using `dplyr` functions?

```
dat %>%
  group_by(Year) %>%
  summarize(totaltimes = sum(sec))
```

```
## # A tibble: 2 x 2
##    Year totaltimes
##   <dbl>      <dbl>
## 1  2013       6828
## 2  2014       6679
```

As seen above, more fish were seen by researchers in 2013 than 2014

7. In the original dataset, What is the mean time spent in each new state?

```
dat %>%
  group_by(new.state) %>%
  summarize(meantime = mean(sec))
```

```
## # A tibble: 4 x 2
##   new.state  meantime
##   <chr>         <dbl>
## 1 Aggression     10.9
## 2 Female         10.3
## 3 Rest           28.9
## 4 Rove           24.9
```

8. What is the mean time spent in each new state in 2013 and 2014 separately? What is the mean time spent in each new state in slope and shelf habitat?

```r
dat %>%
  group_by(Year,new.state) %>% # make 2 (2 levels in the Year column) x 4
# (4 levels in new.state) = 8 groups
  summarise(meantime8a = mean(sec))
```

```
## # A tibble: 8 x 3
## # Groups:   Year [2]
##    Year new.state  meantime8a
##   <dbl> <chr>           <dbl>
## 1  2013 Aggression       8.75
## 2  2013 Female           7.42
## 3  2013 Rest            34.4
## 4  2013 Rove            24.7
## 5  2014 Aggression      13.7
## 6  2014 Female          16
## 7  2014 Rest            23.1
## 8  2014 Rove            25.1
```

```r
# dat %>%
#   group_by(habitat,new.state) %>%
#   summarise(meantime8b = mean(sec))
# the above line of code makes different groups slope/Slope and Slope/Shelf
# (upper and lower case treated differently)

dat$habitat[which(dat$habitat=="Shelf")] <- "shelf"
dat$habitat[which(dat$habitat=="Slope")] <- "slope"
# replacing uppercase ones with lowercase ones

dat %>%
  group_by(habitat,new.state) %>%
  summarise(meantime8b = mean(sec))
```

```
## # A tibble: 8 x 3
## # Groups:   habitat [2]
##   habitat new.state  meantime8b
##   <chr>   <chr>           <dbl>
## 1 shelf   Aggression       5.76
## 2 shelf   Female          10
## 3 shelf   Rest            29.4
## 4 shelf   Rove            21.7
## 5 slope   Aggression      14.8
## 6 slope   Female          10.7
## 7 slope   Rest            28.2
## 8 slope   Rove            29.1
```

9. Can you calculate the standard deviation (or spread) of time spent in each new state in the two habitat categories?

```r
dat %>%
  group_by(new.state,habitat) %>%
  summarise(meantime8b = sd(sec))
```

```
## # A tibble: 8 x 3
## # Groups:   new.state [4]
##   new.state  habitat meantime8b
##   <chr>      <chr>        <dbl>
## 1 Aggression shelf         4.86
## 2 Aggression slope        18.2
## 3 Female     shelf        11.4
## 4 Female     slope        10.1
## 5 Rest       shelf        27.6
## 6 Rest       slope        24.1
## 7 Rove       shelf        18.0
## 8 Rove       slope        33.9
```

10. Save the New state column to an empty vector using indexing and $ operation, and coerce that vector back into a data frame, and then to a tibble

```r
vec10 <- NULL
vec10 <- dat$new.state
# vec10 <- dat[,"new.state"] # alternative

dat10 <- as.data.frame(vec10)
tib10 <- as_tibble(dat10) # as.tibble() also works but is not recommended

# alternatives without coercing data types:
# dat10 <- data.frame(col1 = vec10)
# tib10 <- tibble(col1 = vec10)
```

11. Save this data frame as a CSV both in the working directory and on your desktop

```r
write_csv(x = dat10,path = "newdat10.csv")
# automatically goes to working directory

write_csv(x = dat10,path = "C:/Users/aksha/Desktop/newdat10.csv")
# saves to desktop, path is slightly different depending on your computer & OS
```