Assignment_04

Steven Tran

February 20, 2018

```
library(tidyverse)
```

```
## -- Attaching packages ----------------------------------------------- tidyverse 1.2.1 --
```

```
## v ggplot2 2.2.1     v purrr   0.2.4
## v tibble  1.4.2     v dplyr   0.7.4
## v tidyr   0.8.0     v stringr 1.2.0
## v readr   1.1.1     v forcats 0.2.0
```

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

```
library(dplyr)
```

R for Data Science

10.5 Exercises

1. How can you tell if an object is a tibble?

A tibble is a special type of data frame. It will print the first ten rows when it's called and each of its columns will specify their datatype below the column names. In addition to that, it will require the $ or [[]] operators in order to extract data from a tibble. ### 2. Compare and contrast the following operations on a data.frame and equivalent tibble. What is different? Why might the default data frame behaviours cause you frustration?

df <- data.frame(abc = 1, xyz = "a") df$x df[, "xyz"] df[, c("abc", "xyz")]

The operations on the tibble gives more data (datatype of column) and doesn't forgive any mistakes. For instance df$x works on the data.frame but not on the tibble. This is because the data.frame assumes you are talking about the variable x but tibble rejects this. Default data frame behaviours can cause frustration because you may receive some data, try to compare a few observations, and receive errors because you didn't check the class of the data you are comparing or can't convert it to the appropriate type.

3. If you have the name of a variable stored in an object, e.g. var <- "mpg", how can you extract the reference variable from a tibble?

You can use the [[]] operators in order to extract the reference variable. For instance, as_tibble(mtcars)[[var]] where var <- "mpg" would give the column mpg. Similarly, you can use pipes with the . operator like in as_tibble(mtcars) %>% [[var]] where var <- "mpg" to get the same result.

4.Practice referring to non-syntactic names in the following data frame by:

1.Extracting the variable called 1.

2. Plotting a scatterplot of 1 vs 2.

3. Creating a new column called 3 which is 2 divided by 1.

4. Renaming the columns to one, two and three.

```
annoying <- tibble(
  `1` = 1:10,
  `2` = `1` * 2 + rnorm(length(`1`))
)
```

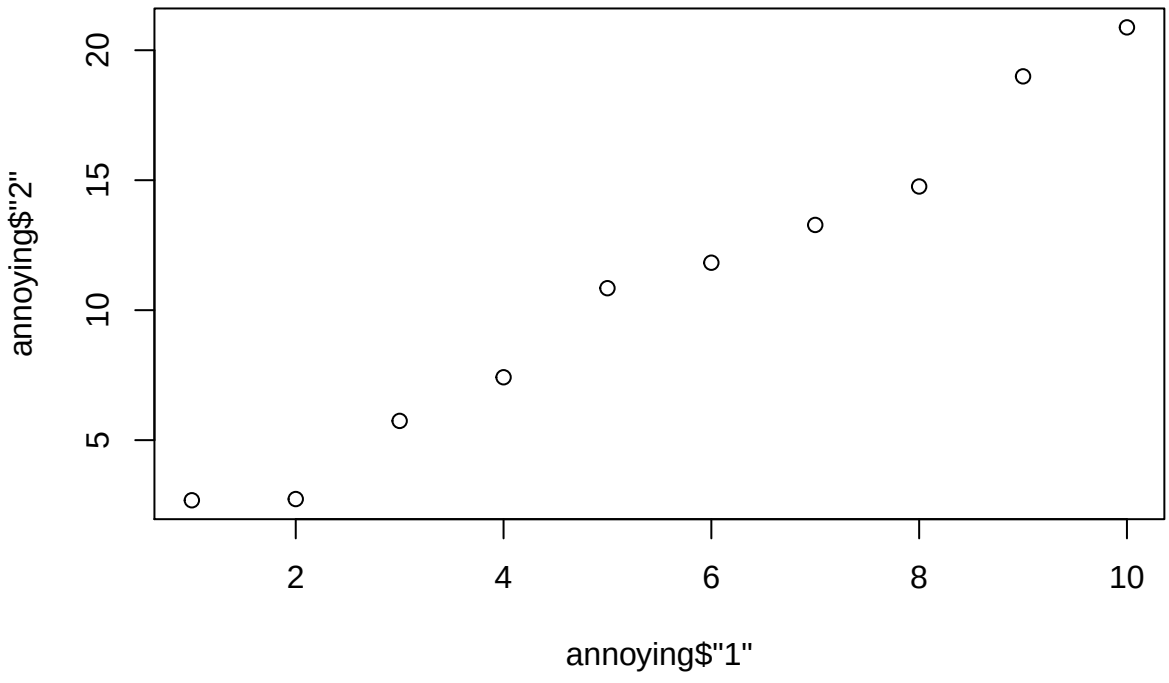
For extracting the variable called 1:

```
annoying$`1`
```

```
##  [1]  1  2  3  4  5  6  7  8  9 10
```

For plotting a scatterplot of 1 vs 2:

```
plot(annoying$'1', annoying$'2')
```



For creating a new column called 3 which is 2 divided by 1:

```
annoying <- mutate(annoying, annoying$'1'/ annoying$'2')
colnames(annoying) <- c(colnames(annoying)[-length(colnames(annoying))],"3")
```

For renaming the columns to one, two, and three:

```
colnames(annoying) <- c("one","two","three")
```

5. What does tibble::enframe() do? When might you use it?

enframe() converts vectors or lists to a dataframe. The opposite to this is deframe(). I would use enframe() when I am given a vector or list to analyze.

What option controls how many additional column names are printed at the footer of a tibble?

tibble.width controls how many additional column names are printed.