Assignment_04

Steven Tran

February 20, 2018

```r
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()
```

```r
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`))
)
```

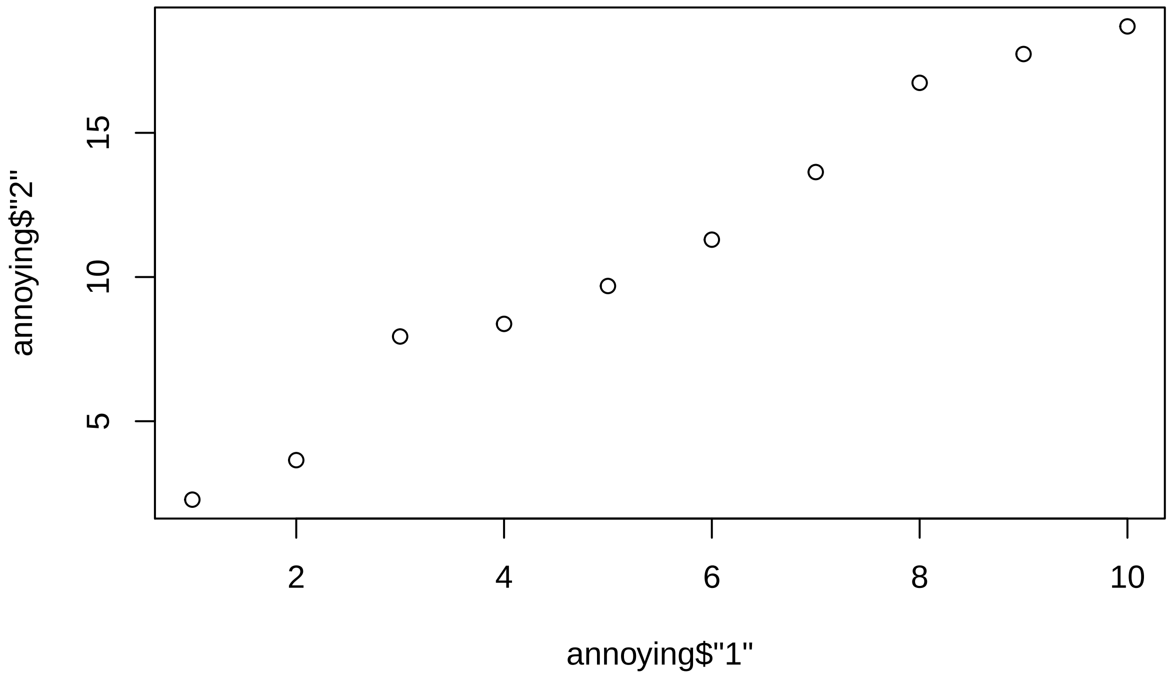
For extracting the variable called 1:

```
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:

```r
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.

12.6 Exercises

1. In this case study I set na.rm = TRUE just to make it easier to check that we had the correct values. Is this reasonable? Think about how missing values are represented in this dataset. Are there implicit missing values? What's the difference between an NA and zero?

Looking back at the original data, I noticed that any row with NA in any of its columns had an NA for every other column. This probably meant that there was no information about that specific country in that time period. Also, one can see that they are usually blocks of missing information, not randomly distributed throughout the data. I think it is reasonable to just remove all the rows with NA for this dataset. The difference between NA and zero is that NA implies that this data is missing whereas a zero indicates that the observations were made and that zero was the value for that cell.

2. What happens if you neglect the mutate() step? (mutate(key = stringr::str_replace(key, "newrel", "new_rel")))

If we negelected that step, then any rows with a key that started with newrel wouldve not been separated later on. It would have made the data messy since NA cells would pop up again.

3. I claimed that iso2 and iso3 were redundant with country. Confirm this claim.

```r
head(who)
```

```
## # A tibble: 6 x 60
##   country     iso2  iso3   year new_sp_m014 new_sp_m1524 new_sp_m2534
##   <chr>       <chr> <chr> <int>       <int>        <int>        <int>
## 1 Afghanistan AF    AFG    1980          NA           NA           NA
## 2 Afghanistan AF    AFG    1981          NA           NA           NA
## 3 Afghanistan AF    AFG    1982          NA           NA           NA
## 4 Afghanistan AF    AFG    1983          NA           NA           NA
## 5 Afghanistan AF    AFG    1984          NA           NA           NA
## 6 Afghanistan AF    AFG    1985          NA           NA           NA
## # ... with 53 more variables: new_sp_m3544 <int>, new_sp_m4554 <int>,
## #   new_sp_m5564 <int>, new_sp_m65 <int>, new_sp_f014 <int>,
## #   new_sp_f1524 <int>, new_sp_f2534 <int>, new_sp_f3544 <int>,
## #   new_sp_f4554 <int>, new_sp_f5564 <int>, new_sp_f65 <int>,
## #   new_sn_m014 <int>, new_sn_m1524 <int>, new_sn_m2534 <int>,
## #   new_sn_m3544 <int>, new_sn_m4554 <int>, new_sn_m5564 <int>,
## #   new_sn_m65 <int>, new_sn_f014 <int>, new_sn_f1524 <int>,
## #   new_sn_f2534 <int>, new_sn_f3544 <int>, new_sn_f4554 <int>,
## #   new_sn_f5564 <int>, new_sn_f65 <int>, new_ep_m014 <int>,
## #   new_ep_m1524 <int>, new_ep_m2534 <int>, new_ep_m3544 <int>,
## #   new_ep_m4554 <int>, new_ep_m5564 <int>, new_ep_m65 <int>,
## #   new_ep_f014 <int>, new_ep_f1524 <int>, new_ep_f2534 <int>,
## #   new_ep_f3544 <int>, new_ep_f4554 <int>, new_ep_f5564 <int>,
## #   new_ep_f65 <int>, newrel_m014 <int>, newrel_m1524 <int>,
## #   newrel_m2534 <int>, newrel_m3544 <int>, newrel_m4554 <int>,
## #   newrel_m5564 <int>, newrel_m65 <int>, newrel_f014 <int>,
## #   newrel_f1524 <int>, newrel_f2534 <int>, newrel_f3544 <int>,
## #   newrel_f4554 <int>, newrel_f5564 <int>, newrel_f65 <int>
```

```r
tail(who)
```

```
## # A tibble: 6 x 60
##   country  iso2  iso3   year new_sp_m014 new_sp_m1524 new_sp_m2534
##   <chr>    <chr> <chr> <int>       <int>        <int>        <int>
## 1 Zimbabwe ZW    ZWE    2008         127          614            0
## 2 Zimbabwe ZW    ZWE    2009         125          578           NA
## 3 Zimbabwe ZW    ZWE    2010         150          710         2208
## 4 Zimbabwe ZW    ZWE    2011         152          784         2467
## 5 Zimbabwe ZW    ZWE    2012         120          783         2421
## 6 Zimbabwe ZW    ZWE    2013          NA           NA           NA
## # ... with 53 more variables: new_sp_m3544 <int>, new_sp_m4554 <int>,
## #   new_sp_m5564 <int>, new_sp_m65 <int>, new_sp_f014 <int>,
## #   new_sp_f1524 <int>, new_sp_f2534 <int>, new_sp_f3544 <int>,
## #   new_sp_f4554 <int>, new_sp_f5564 <int>, new_sp_f65 <int>,
## #   new_sn_m014 <int>, new_sn_m1524 <int>, new_sn_m2534 <int>,
## #   new_sn_m3544 <int>, new_sn_m4554 <int>, new_sn_m5564 <int>,
## #   new_sn_m65 <int>, new_sn_f014 <int>, new_sn_f1524 <int>,
## #   new_sn_f2534 <int>, new_sn_f3544 <int>, new_sn_f4554 <int>,
## #   new_sn_f5564 <int>, new_sn_f65 <int>, new_ep_m014 <int>,
## #   new_ep_m1524 <int>, new_ep_m2534 <int>, new_ep_m3544 <int>,
## #   new_ep_m4554 <int>, new_ep_m5564 <int>, new_ep_m65 <int>,
## #   new_ep_f014 <int>, new_ep_f1524 <int>, new_ep_f2534 <int>,
## #   new_ep_f3544 <int>, new_ep_f4554 <int>, new_ep_f5564 <int>,
## #   new_ep_f65 <int>, newrel_m014 <int>, newrel_m1524 <int>,
## #   newrel_m2534 <int>, newrel_m3544 <int>, newrel_m4554 <int>,
## #   newrel_m5564 <int>, newrel_m65 <int>, newrel_f014 <int>,
## #   newrel_f1524 <int>, newrel_f2534 <int>, newrel_f3544 <int>,
## #   newrel_f4554 <int>, newrel_f5564 <int>, newrel_f65 <int>
```

No matter which observation one picks, iso2 and iso3 changes accordingly with country and is redundant.

4. For each country, year, and sex compute the total number of cases of TB. Make an informative visualisation of the data.

```r
whoTidy <- who %>%
  gather(code, value, new_sp_m014:newrel_f65, na.rm = TRUE) %>%
  mutate(code = stringr::str_replace(code, "newrel", "new_rel")) %>%
  separate(code, c("new", "var", "sexage")) %>%
  select(-new, -iso2, -iso3) %>%
  separate(sexage, c("sex", "age"), sep = 1) %>%
  group_by(country, year, sex) %>%
  summarize(Number =n())
whoTidy
```

```
## # A tibble: 6,921 x 4
## # Groups:   country, year [?]
##    country      year sex   Number
##    <chr>       <int> <chr>  <int>
##  1 Afghanistan  1997 f          7
##  2 Afghanistan  1997 m          7
##  3 Afghanistan  1998 f          7
##  4 Afghanistan  1998 m          7
##  5 Afghanistan  1999 f          7
##  6 Afghanistan  1999 m          7
##  7 Afghanistan  2000 f          7
##  8 Afghanistan  2000 m          7
##  9 Afghanistan  2001 f          7
## 10 Afghanistan  2001 m          7
## # ... with 6,911 more rows
```