Assignment 4

Re Francesco Ignazio

21 febbraio 2018

(I am exchange student, my written English may result not correct at times)

1

How can you tell if an object is a tibble?

Tibbles display a limited numbers of rows. In tibbles you can see on the top of each column the class of its elements. Moreover tibbles have class `tbl_df` and `tbl` in addition to `data.frame`.

```r
class(iris)
```

```
## [1] "data.frame"
```

```r
class(as_tibble(iris))
```

```
## [1] "tbl_df"     "tbl"        "data.frame"
```

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?

```r
df <- data.frame(abc = 1, xyz = "a")

#1.a
df$x

#1.b
df[, "xyz"]

#1.c
df[,c("abc", "xyz")]

df <- tibble(abc = 1, xyz = "a")

#2.a
df$x
```

```
## Warning: Unknown or uninitialised column: 'x'.
```

```r
#2.b
df[, "xyz"]

#2.c
df[,c("abc", "xyz")]
```

We can see how in the case of defining df as a data.frame, the statement 1.a returned the column "xyz", as if we had written `df$xyz`. In tibbles instead, unless you type the correct name of the column, it will give an error as we can see if we run 2.a. In 1.b and 1.c we see that according to the number of columns inserted between brackets, the data frame returns a factored vector or a dataframe This may be confusing in the scenario that we were passing a variable whose length is unknown.

3

If you have the name of a variable stored in an object, how can you extract the reference variable from a tibble?

You can use the following: `tibble_name[[var]]`.

4

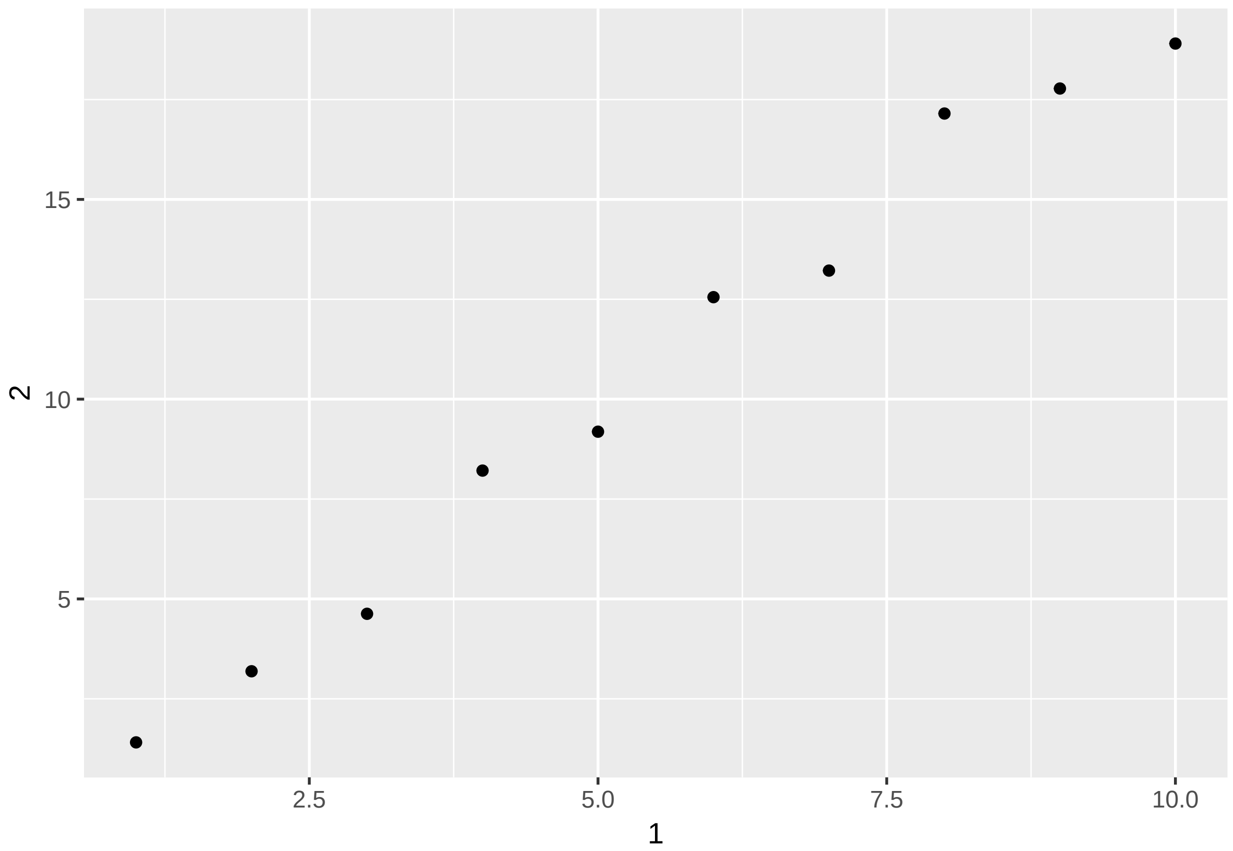
Practice referring to non-syntactic names in the following data frame by:

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`))
)


#1
annoying[["1"]] #or
annoying$`1`

#2
ggplot(annoying, mapping = aes(x =`1`, y=`2`)) + geom_point()
```



```
#3

mutate(annoying, `3` = `2`/`1`) #or
annoying[["3"]] <- annoying[["2"]] / annoying[["1"]]


#4
annoying <- rename(annoying, one = `1`, two = `2`, three = `3`)
```

5

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

enframe() converts named atomic vectors or lists to two-column data frames. For unnamed vectors, the natural sequence is used as name column. It's useful because it creates a tidy data-frame as every observation forms a row, every variable ("name" and "value""") forms a column and each type of observational unit forms a table.

```
enframe(c(a = 5, b = 7))
```

```
## # A tibble: 2 x 2
##   name  value
##   <chr> <dbl>
## 1 a      5.00
## 2 b      7.00
```

6

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

The option `n_extra` in the function `print` determines the number of extra columns to print information for.