Lecture 11: Working with Strings and Date/Time Variables

Managing and Manipulating Data Using R

Introduction

Logistics

Reading

- GW chapter 14 (Strings)
- GW chapter 16 (Dates and Times)

No class next week (11/13)

- Problem set for strings + date/times still due on 11/13
- No problem set due on 11/20

What we will do today

Load the packages we will use today (output omitted)

- **you must run this code chunk after installing these packages**

```
library(tidyverse)
library(stringr)
```

If package not yet installed, then must install before you load. Install in "console" rather than .Rmd file

- Generic syntax: `install.packages("package_name")`
- Install "tidyverse": `install.packages("stringr")`

Note: when we load package, name of package is not in quotes; but when we install package, name of package is in quotes:

- `install.packages("tidyverse")`
- `library(tidyverse)`

Working with Strings

String basics

What are strings?

String refers to a “data type” used in programming to represent text rather than numbers (although it can include numbers)

- Strings have `character` types

```
string1<- "Apple"
typeof(string1) #type is charater
#> [1] "character"
```

- Create strings using `" "`

```
string2 <- "This is a string"
```

- If string contains a quotation, use `' " " '`

```
string3 <- 'example of a "quote" within a string'
```

- To print a string, use `writeLines()`

```
print(string3) #will print using \
#> [1] "example of a \"quote\" within a string"
writeLines(string3)
#> example of a "quote" within a string
```

Common uses of strings

Basic uses:

- Names of files and directories

```
acs_tract <- read_csv("https://raw.githubusercontent.com/ozanj/rclass/master/da
#> Warning: Missing column names filled in: 'X1' [1]
#> Parsed with column specification:
#> cols(
#>   .default = col_double(),
#>   tract_name = col_character(),
#>   tract = col_character(),
#>   race_brks_nonwhiteasian = col_character(),
#>   inc_brks = col_character()
#> )
#> See spec(...) for full column specifications.
```

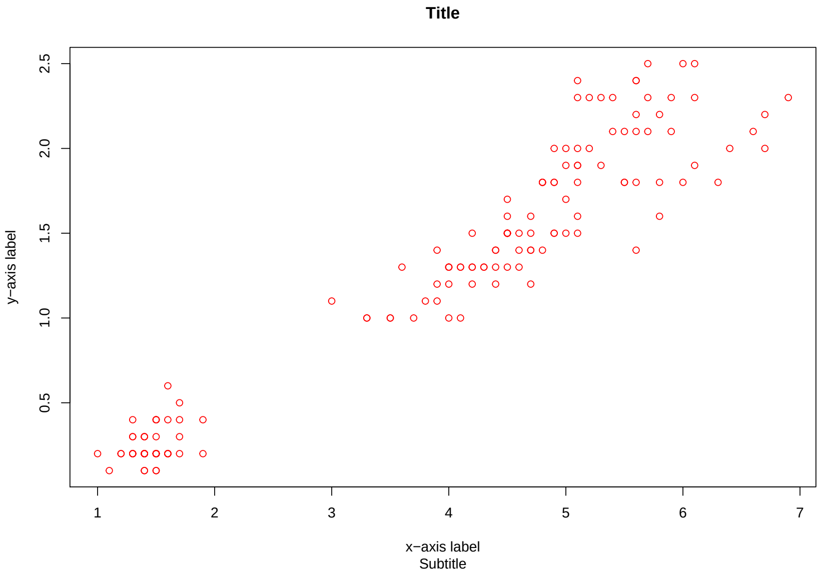
- Names of elements in data objects

```
num_vec <- 1:5
names(num_vec) <- c('uno', 'dos', 'tres', 'cuatro', 'cinco')
num_vec
#>    uno    dos   tres cuatro  cinco
#>      1      2      3      4      5
```

Common uses of strings

- Text elements displayed in plots and graphs

```r
plot(iris$Petal.Length, iris$Petal.Width, main = 'Title', sub = 'Subtitle',
     xlab = 'x-axis label', ylab = 'y-axis label', col = 'red')
```



Common uses of strings

More advanced uses:

- Dealing with identification numbers (leading or trailing zeros)

```
typeof(acs_tract$fips_county_code)
#> [1] "double"

acs_tract <- acs_tract %>%
  mutate(char_county=
  str_pad(as.character(fips_county_code), side = "left" ,3, pad="0"))
```

-Regular expressions

Common uses of strings

-Complex reshaping (tidying) of data

- Problem: multiple variables crammed into the column names
    - new_ prefix = new cases
    - sp/rel/sp/ep describe how the case was diagnosed
    - m/f gives the gender
    - digits are age ranges

```
who %>% pivot_longer(
  cols = new_sp_m014:newrel_f65,
  names_to = c("diagnosis", "gender", "age"),
  names_pattern = "new_?(.*)_(.)(.*)",
  values_to = "count"
)
```