**Editing Text Variables**

* **Example data**
* **Fixing character vectors – tolower(), toupper()**

Names(cameraData)

Tolower(cameraData)

* **Fixing character vectors – strsplit()**

splitNames = strsplit(names(cameraData, “\\”))

splitNames[[5]]

* **Quick aside – lists**

mylist <- list(letters = c(“A”, “b”, “c”), numbers = 1:3, matrix(1:25, ncol = 5))

Head(mylist)

Mylist$letters

* **Using sapply()**

firstElement <- function(x) { x[1]}

sapply(splitNames, firstElement)

* **sub()**

names(reviews)

sub(“\_”, “”, names(reviews),)

* **gsub()**

testName <- “this\_is\_a\_test”

gsub(“\_”, “”, testName)

* **finding values – grep(), grepl()**

grep(“Alameda”, cameraData$intersection)

table(grepl(“Alameda”, cameraData$intersection))

cameraData2 <- cameraData[!grepl(“Alameda”, cameraData$intersection)]

* **More on grep()**

grep(“Alameda”, cameraData$intersection, value = TRUE)

grep(“JeffStreet”, cameraData$intersection)

length(grep(“JeffStreet”, cameraData$intersection))

* **Other string functions**

Nchar(“Jeffrey Leek”)

Substr(“Jeffrey”, 1, 7)

Paste(“Jeffrey”, “Leek”)

Paste0()

Str\_trim(“Jeff “)

* **Important problems**
  + - Names should be
      * All lowercases
      * Descriptive
      * Not duplicated
      * No underscore, dots, white spaces
    - Variables with character values
      * Should be made into factor variables
      * Should be descriptive (MALE/FEMALE, TRUE/FALSE instead of 1/0)

**Regular expressions 1**

* **Regular expressions**
  + - A combination of literals and metacharacters
* **Literals** 
  + - Simplest patterns consists only of literals
* **We need a way to express**
  + - Whitespace word boundaries
    - Sets of literals
    - The beginning and the end of a line
    - Alternatives (“war” or “peace”)
* **Metacharacters** 
  + - $represent the end of a line
    - For example: morning$
* **Character classes with []**
  + - We can list a set of characters we will accept
    - Example [Bb] shall accept both B and b
    - Or
      * ^[Ii] 🡺 beginning of a line, both I and i
    - Or
      * ^[0 – 9] [a – zA – Z]
    - Or
      * [^?.]$

**Regular expressions 2**

* **“.” Is used to refer to any character**
  + - Example
      * 9.11 will match 9-11, 9/11, and many more
* **| means “or”**
  + - Example
      * Flood | earthquake | hurricane
* **( and )**
  + - Example
      * ^([Gg]ood | [Bb]ad)
* **“?” means that the expression is optional**
  + - Example
      * [Gg]eorge ([Ww]\.)? [Bb]ush
* **\* and +**
  + - \* means any number, including none
    - + means at least one of the item
* **{ and }**
  + - Specify the min and max number of matches of an expression

**Working with dates**

* **Starting simple**

D1 <- date()

D1

Class(d1)

D2 <- Sys.Date()

D2

* **Formatting dates**

Format(d2, “%a %b %d”)

* **Creating dates**

X <- c(“1jan1984”)

Z <- as.Date(x, “%d%b%Y”)

Z

* **Converting to Julian**

Weekdays(d2)

Months(d2)

Julian(d2)

Attr(, “origin”)

* **Lubridate**

Library(lubridate)

Ymd(“20140408”)

Mdy(“04/08/2019”)

Dmy(“23-05-2018”)

* **Dealing with times**

Ymd\_hms(“”, tz = )

?Sys.timezone

* Others

X <- dmy()

Wday(x[1])

Wday(x[1], label = TRUE)