names(cameraData)

[1] "address" "direction" "street" "crossStreet" "intersection" "Location.1"

tolower(names(cameraData))

[1] "address" "direction" "street" "crossstreet" "intersection" "location.1"

Fixing character vectors - strsplit()

Good for automatically splitting variable names

Important parameters: x, split

splitNames = strsplit(names(cameraData),"\\.")

splitNames[[6]]

[1] "Location" "1"

Programmatically take only the 1st element (before, say, a period) from variable names

Fixing character vectors - sapply():Applies a function to each element in a vector or list

Important parameters: X,FUN

splitNames[[6]][1]

[1] "Location"

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

**sapply(splitNames,firstElement)**

[1] "address" "direction" "street" "crossStreet" "intersection" "Location"

Most Practical so Far

Fixing character vectors - sub()

Important parameters: pattern, replacement, x

names(reviews)

[1] "id" "solution\_id" "reviewer\_id" "start" "stop" "time\_left"

[7] "accept"

**sub("\_","",names(reviews),)**

[1] "id" "solutionid" "reviewerid" "start" "stop" "timeleft" "accept"

\***gsub() removes all cases of the pattern in var-name,**

testName <- "this\_is\_a\_test" #sub() will only remove the first “\_”. gsub() will get them all.

Finding values - grep(),grepl()

grep() returns the indices that contain the string you have searched for. You can also have grep return the values themselves for you to look at w/ value=TRUE

grepl() returns a logical vector contain T/F at each position for whether the pattern was matched.

**grep("Alameda",cameraData$intersection)**

[1] 4 5 36

**grep("Alameda",cameraData$intersection,value=TRUE)**

[1] "The Alameda & 33rd St" "E 33rd & The Alameda" "Harford \n & The Alameda"

**table(grepl("Alameda",cameraData$intersection))**

FALSE TRUE

77 3

#grepl() can be used to subset. Take the rows where the column intersection does not contain “Alamed”

**cameraData2 <- cameraData[!grepl("Alameda",cameraData$intersection),]**

More useful string functions

paste0("Jeffrey","Leek") #paste() would paste w/a space between the ‘y’ and ‘L’

[1] "JeffreyLeek"

str\_trim("Jeff ") #remove excess spaces

[1] "Jeff"

EXERCISES:

Question 4

Load the Gross Domestic Product data for the 190 ranked countries in this data set:

https://d396qusza40orc.cloudfront.net/getdata%2Fdata%2FGDP.csv

Load the educational data from this data set:

https://d396qusza40orc.cloudfront.net/getdata%2Fdata%2FEDSTATS\_Country.csv

Match the data based on the country shortcode. Of the countries for which the end of the fiscal year is available, how many end in June?

Original data sources:

http://data.worldbank.org/data-catalog/GDP-ranking-table

http://data.worldbank.org/data-catalog/ed-stats

#ed contains the country codes in a column called CountryCode. Change gdp to match so dplyr’s inner\_join can join them.

gdp <- rename(gdp, CountryCode = X)

ijGE <- inner\_join(gdp,ed)

#Special.Notes column contains info about the fiscal year

#got to look for rows that contain both ‘fiscal year end’ & ‘june’

JuneFisYear <- grepl("(june).\*(fiscal year end)|(fiscal year end).\*(june)", tolower(ijGE$Special.Notes))

summary(JuneFisYear) ##13

QUESTION 5

You can use the quantmod (http://www.quantmod.com/) package to get historical stock prices for publicly traded companies on the NASDAQ and NYSE. Use the following code to download data on Amazon's stock price and get the times the data was sampled.

library(quantmod)

amzn = getSymbols("AMZN",auto.assign=FALSE)

sampleTimes = index(amzn)

How many values were collected in 2012? How many values were collected on Mondays in 2012?

> str(sampleTimes)

Date[1:2097], format: "2007-01-03" "2007-01-04" "2007-01-05" "2007-01-08" "2007-01-09" "2007-01-10" ...

> head(sampleTimes)

[1] "2007-01-03" "2007-01-04" "2007-01-05" "2007-01-08" "2007-01-09" "2007-01-10"

> twenty12 <- grep("2012",sampleTimes,value=TRUE)

> head(twenty12)

[1] "2012-01-03" "2012-01-04" "2012-01-05" "2012-01-06" "2012-01-09" "2012-01-10"

> length(twenty12)

[1] 250

#converty twenty12 to type ‘date’

twentyDates <- as.Date(twenty12)

#convert dates to days of week

days12 <- weekdays(twentyDates)

#Count the number of Mondays

AreMondays <- grepl("Monday",days12)

> summary(AreMondays)

[1] 47