Hannah Kosinovsky
STA 141
Section: 8 am Olson
12/09/15

Assignment 6

I did this assignment by myself and developed and wrote the code for each part by myself, drawing only from class, section, Piazza posts and the Web. I did not use code from a fellow student or a tutor or any other individual.

Questions:

Part 1:

In order to find all the information that was asked for in part one, I went to http://stackoverflow.com/questions/tagged/r and inspected relevent elements to the questions.

Who posted it? : To find this information I inspected the username elment. I found the path

namepath = "//div[@class = 'user-details']/a/text()"

It gets the text under the tag 'a' where the attribute class = 'user-details' in tag div.

One thing I noticed later on in the assignment, however, was that the paths to find anonymous users, or users with no hyperlink is different. So I went back and found the path:

anon_path = "//div[@class = 'user-details']/text()"

When was it posted? : I inspected the "asked __ mins/hours/days/weeks ago" element. I found the path

"//div[@class = 'user-action-time']/span/@title"

I knew I wanted all elements with the attribute 'title' because if I printed the text() like in name it would give me something like "7 minutes ago" but I wanted an actual time and date and saw that this information was included in the title link.

The title of the post: I inspected the title hyperlink and got the path:

titlepath = "//div[@class = 'summary']/h3/a/text()"

I noticed that other than ‘a’, there was the tag ‘h3’ in the line above. The attribute class was called ‘summary’ in tag div. In this element, the title was included in the link and in text outside the link, but I decided it would be easier to capture the text and that way I wouldn’t have to worry about the extra link information other than the title.

The reputation level of the poster: I inspected the reputation score element under the username. I came up with the path:

rep_path = "//div[@class = '-flair']/span/text()"

It gets the text under the tag ‘span’ where the attribute class = '-flair'in tag div.

The current number of views for the post: I inspected the element number of views and found the path:

views_path = "//div[@class = 'views ']/text()"

I want the text in the tag ‘div’ where the attribute class was ‘views’ and all I needed was the text ouside the links. Since the information was on that line I didn’t need to reference any other tags to find it.

The current number of answers for the post: I inspected the element number of answers to find the path:

answerpath = "//div[@class = 'status answered' or @class = 'status unanswered' or @class = 'status answered-accepted']/strong/text()"

For this path, I saw that depending on the element, the class was different so I accounted for all class types. Then I used the tag ‘strong’ and saw that all I needed was the text outside of the links.

The vote "score" for the post: I inspected the vote score element and found the path:

votes_path = "//span[@class = 'vote-count-post ']/strong/text()"

This time the tag was span. The attribute class ‘vote-count-post’ and I needed to get the text from after the ‘strong’ attribute link.

The URL for the page with the post, answers and comments: I inspected the hyperlink for the title again, because I knew that this link led to the page with just the pertaining post. I found the path:

pagepath = "//div[@class = 'summary']/h3/a/@href"

This process was similar to finding the title, but instead of the text outside of the links, I wanted the link itself, so I looked for all elements with the tag 'href' because that's where the link information was.

The id (a number) uniquely identifying the post: I could not find this element anywhere on the webpage itself. I found it within the information for the post urls, so I actually used regular expressions on the info I found for post_urls to find the ids.

```
postids=regexec("[questions]\\/([0-9]{7,})", post_urls)
```

For the last questions of part 1: "**Obtain the URL for the "next" page listing posts repeat steps 1, 2, 3"**

I wrote a function to find all the information on each page.

The function:

1. Takes in the specified tag, and the number of pages the user wants to parse through as arguments.
2. Initiallizes an empty data frame to accumulate the data over all iterations
3. Use sprintf so that the tag and page number can be inputted into the url.
4. Finds all the info about the page for each iteration: tags, views, etc.
5. Skips to the next page if the number of names the names path picked up was less than 30. This is necessary because anonymous names are not picked up by the path for regular names. I chose less than 30 because I using the url for 30 posts for page and all info extracted has to have a length of 30 in order to be made into a dataframe.
6. Binds the initial data frame with all iterations.
7. Outputs the final data frame

A sample of my data frame (formatted in word):

names	times	titles	rep_scores	num_views
shadi	2015-12-08 00:00:32Z	Reordering the variables	17	10
Mohaa	2015-12-07 23:47:11Z	Check the visibility of an object in a webpage using its xpath? [duplicate]	125	6
ZAWD	2015-12-07 23:41:49Z	Matrix of pearson correlation and P value in R	118	8
SBryson	2015-12-07 23:31:32Z	Meaning of sd argument in R function: "sd="	1	11
Katie	2015-12-07	Post-hoc test for	1	6

	23:28:30Z	categorical variables/interaction terms in R		
Fabian	2015-12-07 23:05:55Z	Fiscal-year return and standard deviation from daily returns	1	15
Alex	2015-12-07 22:53:50Z	Prevent geom_points and their corresponding labels from overlapping	148	11

num_answers	num_votes	post_urls	ids	tags_all
1	0	http://stackoverflow.com/questions/34145665/reordering-the-variables	34145665	r program
0	0	http://stackoverflow.com/questions/34145522/check-the-visibility-of-an-object-in-a-webpage-using-its-xpath	34145522	r seleniun web-scraping rselenium
0	0	http://stackoverflow.com/questions/34145467/matrix-of-pearson-correlation-and-p-value-in-r	34145467	r matrix p value
0	-4	http://stackoverflow.com/questions/34145336/meaning-of-sd-argument-in-r-function-sd	34145336	r argument standards deviation
0	0	http://stackoverflow.com/questions/34145307/post-hoc-test-for-categorical-variables-interaction-terms-in-r	34145307	r posthoc
1	0	http://stackoverflow.com/questions/34145027/fiscal-year-return-and-standard-deviation-from-daily-returns	34145027	r financial
0	0	http://stackoverflow.com/questions/34144871/prevent-geom-points-and-their-corresponding-labels-from-overlapping	34144871	r ggplot2 scatter-plot

Part II Scraping the Posts, Answers and Comments

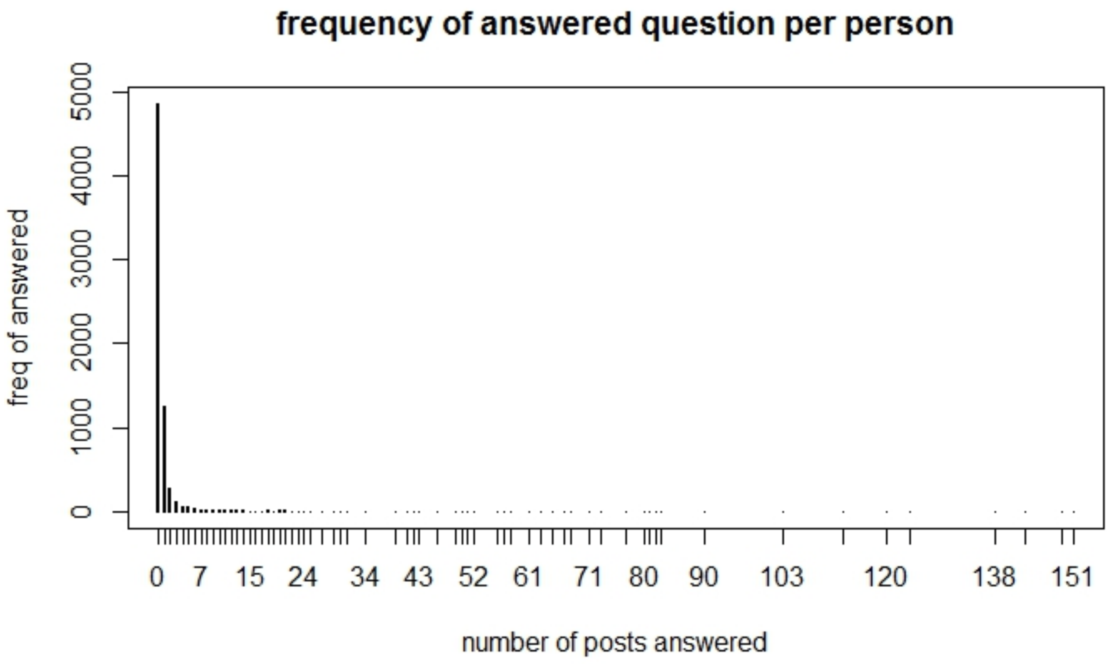
In this part, I will scrape each content of the post, included question, answer, and comments for both of them. For this part I used a similar approach in part 1 to obtain my Xpaths. I found the relevant tags, attributes, and explored where the information I needed to extract was in each element. The code for all the Xpaths can be found in my appendix.

Part 3:

What is the distribution of the number of questions each person answered?:

For this question I used the package “data.table”. It allowed me to make a column of ones so that I could get the count of answers.

Then I found the conditional sum of ones for type “answer” and grouped by user id. I then made a plot of the table:



Most people do not answer posts frequently, however, there are several very active users.

What are the most common tags?:

I decided that for this question it was best to use the data frame I got from running my large function on top 150 times so that I could get information about the tags from many pages. So then I accessed the column tags in my data frame, split them by the space and unlisted them, and then made a table to see the most common ones.

```
> head(sort(table(tags), decreasing = TRUE))
tags
     r      ggplot2     plot    data.frame     shiny    data.table
   4140      322        165      161            160        150
```

It makes sense that r is the most common tag because we are scraping the web for questions tagged r.

How many questions are about ggplot?:

For this question I need to look through the text column in the rQAs data frame. I simply used a regular expression to see how many times ggplot was mentioned. I found 1186 questions about ggplot.

How many questions involve XML, HTML or Web Scraping?:
Similarly to the previous question, I used regular expressions on the text column. I found 1336 questions about xml, html or web scraping.
What are the names of the R functions referenced in the titles of the posts?:

First, I realized that in the data frame rQAs you could not access the row.names column. So I added a column called rownames at the end of the data frame so that I could access the information about the titles. I then gsubbed out the beginning of the title urls and split them by "-" so that I am left with separate words from the title. I then wrote a function that:

1. Uses sprintf to specify index in array of words

I define a function as an entity that will give a result when I time a question command in the r console. In other words, if ?word returns a valid help file, then word is a function. The problem I couldn't account for is functions from packages I have not downloaded. In order for console to recognize "?word" as a command, I needed to use the function eval() as shown in the appendix where my function is. using function class(), I can see that the proper type of a ?word command is "help_files_with_topic". Running eval() function on a carachter string of form "?word" into a "help_files_with_topic" type. When I applied the eval function in this manner, I found that if a word was a function, eval("?word") return object of length If "word" is not a function, eval("?word") returns object of length 0.

2. Adds a word to the list if it returns a length of 1 as specified above.
3. Uses tryCatch in order to avoid fatal crashes in function if eval led to an error condition (as it does in cases such as "while" and "in")
4. Returns list of functions

My function found 464 functions in the titles. A sample of the output of my function :

```
"text"      "numeric"    "sum"       "data"        "files"
"get"       "factor"     "windows"   "find"        "which"
"is"        "by"         "plot"      "boxplot"     "help"
"legend"    "vector"     "table"     "with"        "frame"
"show"      "frequency"  "all"       "pairs"       "objects"
"replace"   "colon"      "apply"     "interactive" "match"
```

What are the names of the R functions referenced in the accepted answers and comments of the posts? We can do better than we did in processing the title of the posts as there is HTML markup and we can find content in code blocks:

For this question I used the same method as for the previous question, but I searched through the subset of the text column for just type “comment” and “answer”. Then I split the text by an empty space and found unique elements so that my function would not query the same word multiple times.

A sample of my output:

```
1       help
2       library
3       is
4       as
5       factor
6       replicate
7       iris
8       example
9       image
10      try
11      I
12      boxplot
13      box
14      length
15      terms
16      max
17      IQR
18      range
19      quantile
20      with
21      Reduce
22      text
23      barplot
24      axis
25      text()
26      on-topic
27      mean
28      files
29      all
30      list.files("path")
31      remove
```

Appendix:

```
install.packages("RCurl")
library(RCurl)
install.packages("XML")
```

```
library(XML)
url =
'http://stackoverflow.com/questions/tagged/r?page=1&sort=newest&pagesize=30
'
doc = getURLContent(url)
html = htmlParse(doc, asText = TRUE)

#noticed theres anonymous ppl not being printed in names
anon_path = "//div[@class = 'user-details']/text()"
anon1 =xpathSApply(html, anon_path, xmlValue)
anon = gsub("[^a-zA-Z0-9]", "", anon1)

# forum_tag is the form we want to query (e.g. "R""javascript")
# max_page is the maximum number of pages user wants to be scraped for the given
forum
scraping = function(forum_tag, max_page)
{
  #initialize empty data frame
  DFF = data.frame()
  #get the number of the last page from the first page url
  url =
sprintf("http://stackoverflow.com/questions/tagged/%s?page=1&sort=newest&pa
gesize=30", forum_tag)
  last_page_xpath = "//div[@class = 'pager fl']/a[last()-1]"
  last_page = xpathSApply(html, last_page_xpath, xmlValue)
  last_page = as.integer(gsub("^[0-9]|[[:space:]]", "", last_page))
  for(pagenum in 1:min(last_page,max_page))
  {
    url =
sprintf("http://stackoverflow.com/questions/tagged/%s?page=%d&sort=newest&
pagesize=30", forum_tag, pagenum)
    doc = getURLContent(url)
    html = htmlParse(doc, asText = TRUE)
    #get people
    namepath = "//div[@class = 'user-details']/a/text()"
    names=xpathSApply(html, namepath, xmlValue)
    #if some of the names are anonymous, skip the whole page
    if(length(names) != 30){
      next
    }
    #time of post
    timepath = "//div[@class = 'user-action-time']/span/@title"
    times = unlist(getNodeSet(html, timepath))
    names(times) = NULL

    #title of post
    titlepath = "//div[@class = 'summary']/h3/a/text()"
    titles= xpathSApply(html, titlepath, xmlValue)

    #reputation of poster
    rep_path = "//div[@class = '-flair']/span/text()"
    rep_scores = as.integer(xpathSApply(html, rep_path, xmlValue))

    #current number of views
    views_path = "//div[@class = 'views ']/text()"
    views1 = xpathSApply(html, views_path, xmlValue)
    views = gsub("\r|\n|view(s)|[[:space:]]|,", "", views1)

    #current number of answers
    answerpath = "//div[@class = 'status answered' or @class = 'status unanswered'
or @class = 'status answered-accepted']/strong/text()"
    num_answers = xpathSApply(html, answerpath, xmlValue)

    #vote score for post
    votes_path = "//span[@class = 'vote-count-post ']/strong/text()"
    num_votes = xpathSApply(html, votes_path, xmlValue)

    #the URL for the page with the post, answers and comments
    pagepath = "//div[@class = 'summary']/h3/a/@href"
    post_urls = xpathSApply(html, pagepath, `[[`, 1)
    post_urls  = getRelativeURL(post_urls, url)
    names(post_urls) = NULL

    #ids uniquely identifying post
    postids=regexec("[questions]\\/([0-9]{7,})", post_urls)
    post_matches = regmatches(post_urls, postids)
    ids = sapply(post_matches, function(x) x[2])

    #tags per post
    tags_path = "//div[starts-with(@class,'tags')]"
    tags_all = xpathSApply(html, tags_path, xmlValue, trim = TRUE)

    DF = data.frame(names, times, titles, rep_scores, views,
                    num_answers, num_votes,  post_urls, ids, tags_all)
    DFF <- rbind(DFF, DF)
  }
  return(DFF)
}
DFF=scraping("r",150)

########################################
# part 2
```

```
### function for comments

read.comment = function(posttype, doc){

  # user
  node.comment = getNodeSet(doc, paste("//div[@class='",posttype,"']
                                //div[contains(@id, 'comments-') and contains(@class,
'comments ')]
                                //a[contains(@class,'comment-user')]", sep=''))
  user.comment = xmlSApply(node.comment, xmlValue)

  # reputation
  reputation.comment = xmlSApply(node.comment, function(x) xmlGetAttr(x, 'title'))
  reputation.comment = gsub("[^0-9]", '', reputation.comment)

  # userid
  userid.comment = xmlSApply(node.comment, function(x) xmlGetAttr(x, 'href'))
  userid.comment = sapply(userid.comment, function(x) gsub(".*/([0-9]*)/.*", '\\1',
x), USE.NAMES = FALSE)

  # date
  node.comment = getNodeSet(doc,
paste("//div[@class='",posttype,"']//div[contains(@id, 'comments-')
                                and contains(@class, 'comments
')]//span[@class='relativetime-clean']", sep=''))
  date.comment = xmlSApply(node.comment, function(x) xmlGetAttr(x, "title"))

  # content
  node.comment = getNodeSet(doc,
paste("//div[@class='",posttype,"']//div[contains(@id, 'comments-')
                                and contains(@class, 'comments ')]//span[@class='comment-
copy']", sep=''))
  content.comment = xmlSApply(node.comment, xmlValue)

  # parent
  node.comment = getNodeSet(doc, paste("//div[@class='",posttype,"']", sep = ''))
  parent.comment = xmlSApply(node.comment, function(x) xmlGetAttr(x,
paste("data-", posttype, "id", sep = '')))

  # vote
  vote.comment = NA

  # posttype
  posttype.comment = 'comment'
  # store in dataframe
  comment = data.frame(user = user.comment, type = posttype.comment, userid =
userid.comment, date = date.comment,
                   reputation = reputation.comment, vote = vote.comment, content =
content.comment,
                   parent = parent.comment)

  return(comment)

}

### function for posts

read.post = function(posttype, doc){

  if (posttype!='answer') posttype = 'post'

  node.question = getNodeSet(doc, paste("//div[@class='", posttype, "']", sep = ''))

  # user
  node.post = getNodeSet(doc, paste("//td[@class='",posttype, "cell']
                              //div[@class='user-details']
                              //a", sep = ''))
  user.post = xmlSApply(node.post, xmlValue)

  # userid
  userid.post = xmlSApply(node.post, function(x) xmlGetAttr(x, 'href'))
  userid.post = gsub(".*/([0-9]*)/.*", '\\1', userid.post)

  # date
  node.post = getNodeSet(doc, paste("//td[@class='", posttype, "cell']
                              //div[@class='user-action-time']
                              //span", sep = ''))
  date.post = xmlSApply(node.post, function(x) xmlGetAttr(x, "title"))[1]

  # reputatoin
  node.post = getNodeSet(doc, paste("//td[@class='", posttype, "cell']
                              //div[@class='user-details']
                              //span[@class='reputation-score']", sep = ''))
  reputation.post = xmlSApply(node.post, xmlValue)

  # vote
  if (posttype!='answer') posttype_rename = 'question' else posttype_rename =
'answer'
  note.post = getNodeSet(doc, paste("//div[@class='", posttype_rename,
"']//td[@class='votecell']
```

```
                              //span[@itemprop='upvoteCount']", sep = ''))
  vote.post = xmlSApply(node.post, xmlValue)

  # content
  if (posttype!='answer') posttype_ind = 'question' else posttype_ind = 'answers'
  node.post = getNodeSet(doc, paste("//div[@id='", posttype_ind, "']
                              //td[@class='", posttype, "cell']
                              //div[@class='post-text']", sep = ''))
  content.post = sapply(node.post, function(x) paste(capture.output(print(x)),
collapse = ''))

  # parent
  parent.post = NA

  # posttype
  node.posttype = posttype_rename

  # id
  node.id = getNodeSet(doc, paste("//div[@class='", posttype_rename, "']", sep = ''))
  id.post = xmlSApply(node.id, function(x) xmlGetAttr(x, paste("data-",
posttype_rename, "id", sep = '')))

  post = data.frame(user = user.post, posttype = node.posttype, userid = userid.post,
date = date.post,
                 reputation = reputation.post, vote = vote.post, content = content.post,
                 parent = parent.post)

  return(post)

}

### function to read everything

read.url = function(link){

  # read URL
  txt = getURLContent(link)

  # parse as html
  doc = htmlParse(txt, asText = TRUE)

  # read comment for question
  comment.question = tryCatch(read.comment('question', doc), error=function(e)
NULL)

  # read comment for answer
  comment.answer = tryCatch(read.comment('answer', doc), error=function(e)
NULL)

  # read question
  post.question = tryCatch(read.post('question', doc), error=function(e) NULL)

  # read answer
  post.answer = tryCatch(read.post('answer', doc), error=function(e) NULL)

  posts = rbind(post.question, post.answer, comment.question, comment.answer)
  posts = data.frame(uniqueid = link, posts)

  return(posts)

}

# where url is a vector contains all urls
# and NumOfPages contraint how many urls to read

post = function(url, NumOfPages){
  list.post = lapply(url[1:NumOfPages], function(x) read.url(x))
  all.post = do.call(rbind, list.post)
}

test.url = "http://stackoverflow.com/questions/34121530/run-rscript-from-
command-line-until-it-succeeds"
post(test.url, 1)

#############################################
#part 3
load("~/Downloads/rQAs.rda")
rQAs = cbind(rQAs, row.names(rQAs))
colnames(rQAs)[11] = "rownames"

#1: distribution of number of questions each person answered

install.packages("data.table")
library(data.table)
# made a column of ones so that I could get the count
#of answers
rQAs$ones = 1
dt = data.table(rQAs)
#find conditional sum of ones for type answer and grouped by user id
da =dt[, sum(ones[type=='answer']), by = userid]

answers_per_id = as.factor(sort(table(da$V1), decreasing = TRUE))
```

```
plot(table(da$V1),  main = "frequency of answered question per person",
     xlab = "number of posts answered", ylab= "freq of answered"
     , xlim = c(1,150))

#2
# most common tags out of the pages I ran
tags_all_final = as.character(DFF$tags_all)
split_tags = strsplit(tags_all_final, " ")
tags =unlist(split_tags)
head(sort(table(tags), decreasing = TRUE))

#3 how many questions are about ggplot
get_ggplot = regexec("(ggplot)", rQAs$text, ignore.case = TRUE)
ggplot_matches = regmatches(rQAs$text, get_ggplot)
ggplots = sapply(ggplot_matches, function(x) x[2])

ggplot_rQAs = rQAs[which(!is.na(ggplots)),]
length(unique(ggplot_rQAs$qid))
##1186

#4 How many questions involve XML, HTML or Web Scraping
get_webscrape = regexec("(html)|(xml)|(scraping)", rQAs$text, ignore.case = TRUE)
web_matches = regmatches(rQAs$text, get_webscrape)
web_questions = sapply(web_matches, function(x) x[2])

web_rQAs = rQAs[which(!is.na(web_questions)),]
length(unique(web_rQAs$qid))
#1336

#5 What are the names of the R functions referenced in the titles of the posts?

all_titles = as.character(rQAs$rownames)
minus_stack = gsub("http://stackoverflow.com/questions/[0-9]{8}/", "", all_titles)
split_tags = unlist(strsplit(minus_stack, "-"))

#http://www.talkstats.com/showthread.php/16043-Solved-R-Convert-character-
to-command
#http://mazamascience.com/WorkingWithData/?p=912

all_titles = as.character(rQAs$rownames)
minus_stack = gsub("http://stackoverflow.com/questions/[0-9]{8}/", "", all_titles)
split_tags = unique(unlist(strsplit(minus_stack, "-")))

list_title_functions = c()
for( i in 1:length(split_tags)){
  tryCatch({
    if(length(eval(parse(text = sprintf("?%s",split_tags[i]))))==1){
      list_title_functions =c(list_title_functions, split_tags[i])
    }
  }, warning = function(w) {
    #do nothing
  }, error = function(e) {
    #do nothing
  }, finally = {
    #do nothing
  })
}
list_title_functions
```

#6 What are the names of the R functions referenced in the accepted answers and comments of the posts?

```
answers_comments = rQAs[rQAs$type == c("comment","answer"),]
all_comm_ans = as.character(answers_comments$text)
split_all = unique(unlist(strsplit(all_comm_ans, " ")))

list_functions = c()
for( i in 1:length(split_all)){
  tryCatch({
    if(length(eval(parse(text = sprintf("?%s",split_all[i]))))==1){
      #count_of_functions = count_of_functions + 1
      list_functions =c(list_functions, split_all[i])
    }
  }, warning = function(w) {
    #do nothing
  }, error = function(e) {
    #do nothing
  }, finally = {
    #do nothing
  })
}
list_functions
```

Resources I used besides Piazza, Office Hours, discussion, and class:

http://www.talkstats.com/showthread.php/16043-Solved-R-Convert-character-to-command
http://mazamascience.com/WorkingWithData/?p=912