Problem set 2

Dongyu Lang
SID: 24174288

September, 13, 2017

1 Question 1 (a)

```
chars <- sample(letters, 1e+06, replace = TRUE)
write.table(chars, file = "tmp1.csv", row.names = FALSE, quote = FALSE, col.names = FALSE)
system("ls -l tmp1.csv", intern = TRUE)

## [1] "-rw-r--r--  1 YaleLang  staff  2000000 Sep 15 00:58 tmp1.csv"
```

The reason that the file has 2,000,000 bytes is that all the letters have 1,000,000 bytes and the "/n"s have 1,000,000 bytes.

```
chars <- paste(chars, collapse = "")
write.table(chars, file = "tmp2.csv", row.names = FALSE, quote = FALSE, col.names = FALSE)
system("ls -l tmp2.csv", intern = TRUE)

## [1] "-rw-r--r--  1 YaleLang  staff  1000001 Sep 15 00:58 tmp2.csv"
```

The reason that the file has 1,000,001 bytes is that all the letters have 1,000,000 bytes; however, there is only one "/n" in this situation, which is only 1 byte.

```
nums <- rnorm(1e+06)
save(nums, file = "tmp3.Rda")
system("ls -l tmp3.Rda", intern = TRUE)

## [1] "-rw-r--r--  1 YaleLang  staff  7678253 Sep 15 00:58 tmp3.Rda"

write.table(nums, file = "tmp4.csv", row.names = FALSE, quote = FALSE, col.names = FALSE,
    sep = ",")
system("ls -l tmp4.csv", intern = TRUE)

## [1] "-rw-r--r--  1 YaleLang  staff  18160815 Sep 15 00:58 tmp4.csv"
```

```
write.table(round(nums, 2), file = "tmp5.csv", row.names = FALSE, quote = FALSE,
    col.names = FALSE, sep = ",")
system("ls -l tmp5.csv", intern = TRUE)

## [1] "-rw-r--r--  1 YaleLang  staff  5377686 Sep 15 00:58 tmp5.csv"
```

The reason that tmp4.csv has over 18,000,000 bytes is that the delimiters have 1,000,000 bytes, and each number is 17 digits; thus it is 17 bytes or 18 bytes(if it is negative). Thus, The numbers have in total over 17,000,000 bytes.

The reason that tmp5.csv has over 5,000,000 bytes is that the delimiters have 1,000,000 bytes, and each number is 4 digits; thus it is 4 bytes or 5 bytes (if it is negative). Thus, the numbers have in total over 5,000,000 bytes.

The reason that tmp3.Rda has just over 7,000,000 bytes is that we saved the numbers all in binary format; thus, each number takes fewer bytes, and in total, they have fewer bytes than in tmp4.csv. In addition, save() compressed the file.

2 Question 1 (b)

```
chars <- sample(letters, 1e+06, replace = TRUE)
chars <- paste(chars, collapse = "")
save(chars, file = "tmp6.Rda")
system("ls -l tmp6.Rda", intern = TRUE)

## [1] "-rw-r--r--  1 YaleLang  staff  635271 Sep 15 00:58 tmp6.Rda"

chars <- rep("a", 1e+06)
chars <- paste(chars, collapse = "")
save(chars, file = "tmp7.Rda")
system("ls -l tmp7.Rda", intern = TRUE)

## [1] "-rw-r--r--  1 YaleLang  staff  1056 Sep 15 00:58 tmp7.Rda"
```

I noticed that both of the files are not too large compared with the files in part(a). The reason is that the function save() can reduce the file size considerably by compression. The reason that tmp7.Rda is way smaller than the size of tmp6.Rda is that the save() detects the pattern for "a"; thus, save() makes it much smaller.

3 Question 2 (a)

The function takes in an input that is the researcher's name. after typing in the researcher's name in Google Scholar, We can find out the URL has the pattern that the name is between the URL; thus, I constructed the URL using the

researcher's name, and get all the hyperlinks in the website. I grep the citation page using the pattern of "citations?user=". And also get the researcher ID from the citation page URL.

```
getURLID = function(researcher) {
    Sepname = strsplit(researcher, " ")[[1]]
    Firstname = Sepname[1]
    Lastname = Sepname[length(Sepname)]
    ## Get the URL for the search results
    URL = paste("https://scholar.google.com/scholar?q=", Firstname, "+", Lastname,
        "&btnG=&hl=en&as_sdt=0%2C5", sep = "")
    gethtml = paste(readLines(URL, warn = F), collapse = "\n")
    ## get all the hyperlinks
    hyperlink = str_match_all(gethtml, "<a href=\"(.*?)\"")
    IDindex = grep("citations\\?user=", hyperlink[[1]][, 2])
    ## get the URL for the citation page
    CitationPage = paste("https://scholar.google.com", hyperlink[[1]][, 2][IDindex[1]],
        sep = "")
    ## get the research ID
    ResearcherID = stri_extract_all_regex(CitationPage, "=(.*?)&")[[1]][1]
    ResearcherID = sub("=", "", ResearcherID)
    ResearcherID = sub("&", "", ResearcherID)
    if (is.na(ResearcherID)) {
        return("The author is not in Google Scholar")
    }
    return(c(CitationPage, ResearcherID))
}
getURLID("Geoffrey Hinton")

## [1] "https://scholar.google.com/citations?user=JicYPdAAAAAJ&amp;hl=en&amp;oe=ASCII&amp;oi
## [2] "JicYPdAAAAAJ"
```

4 Question 2 (b)

I look at the developer tool from Chrome, when right click the title, and click inspect. I find out that it is in the class of gsc_a_at. Similarly, the author and journal are in gs_gray. Year is in gdc_a_h, and number of citation is in gsc_a_ac. Thus, I get all the related information from these nodes.

```
infodata = function(citaionpage) {
    intocitation = read_html(citaionpage)
    ## get the title based on .gsc_a_at
    title = html_text(html_nodes(intocitation, ".gsc_a_at"))
    year = html_text(html_nodes(intocitation, ".gsc_a_h"))[-1]
```

```
    Citedby = html_text(html_nodes(intocitation, ".gsc_a_ac"))
    authorandjournal = html_text(html_nodes(intocitation, ".gs_gray"))
    author = authorandjournal[(seq(1, length(authorandjournal), 2))]
    journalyear = strsplit(authorandjournal[(seq(2, length(authorandjournal),
        2))], ",(?=[^,]+$)", perl = TRUE)
    journal = list(0)
    for (i in 1:length(title)) {
        journal[i] = journalyear[[i]][1]
    }
    journal = unlist(journal)
    ## create the data frame
    citationdataframe = data.frame(title = title, author = author, journal = journal,
        year = year, cited_by = Citedby)
    return(citationdataframe)
}
head(infodata(getURLID("Michael Jordan")[1]))
```

```
##                                                                      title
## 1                                          Latent dirichlet allocation
## 2                    On spectral clustering: Analysis and an algorithm
## 3                                 Adaptive mixtures of local experts
## 4 Sharing clusters among related groups: Hierarchical Dirichlet processes
## 5                 Hierarchical mixtures of experts and the EM algorithm
## 6           An introduction to variational methods for graphical models
##                                       author
## 1                  DM Blei, AY Ng, MI Jordan
## 2                  AY Ng, MI Jordan, Y Weiss
## 3   RA Jacobs, MI Jordan, SJ Nowlan, GE Hinton
## 4         YW Teh, MI Jordan, MJ Beal, DM Blei
## 5                        MI Jordan, RA Jacobs
## 6 MI Jordan, Z Ghahramani, TS Jaakkola, LK Saul
##                                                        journal year
## 1       Journal of machine Learning research 3 (Jan), 993-1022 2003
## 2   Advances in neural information processing systems, 849-856 2002
## 3                          Neural computation 3 (1), 79-87 1991
## 4 Advances in neural information processing systems, 1385-1392 2005
## 5                        Neural computation 6 (2), 181-214 1994
## 6                         Machine learning 37 (2), 183-233 1999
##   cited_by
## 1    20010
## 2     6065
## 3     3551
## 4     3081
## 5     3016
## 6     2666
```

5 Question 2 (c)

I want to test first if the function in part A returns the right results. Secondly, I test if the answer in part B returns the right dimension. Lastly, I test if the function can return "The author is not in Google Scholar" if I type in an nonexist researcher.

```
test_that("test for the citation URL, researcher ID and dataframe", {

    totest = getURLID("Geoffrey Hinton")
    datatotest = infodata(totest[1])
    noauthor = getURLID("asdaxzc asdasfasdasd")

    expect_that(totest[1], equals("https://scholar.google.com/citations?user=JicYPdAAAAAJ&am
    expect_that(totest[2], equals("JicYPdAAAAAJ"))
    expect_that(dim(datatotest), equals(c(20, 5)))
    expect_that(noauthor, equals("The author is not in Google Scholar"))
})
```

6 Question 2 (d)

I noticed that when clicked on the right arrow next to "show more", the URL added a "cstart" and a "pagesize"; so I created a new function that tried to find out the maximum cstart that wouldn't cause an error when I called the function in part (b). And the function "getallpub" just combinded each data frame for each cstart, and created the whole new data frame.

```
getmaxcstart = function(citationpage, cstart = 0, pagesize = 20) {
    whentostop = T
    ## want to find out what is the maximum cstart to make the while loop stop
    while (whentostop) {
        newpage = paste(citationpage, "&cstart=", cstart, "&pagesize=", pagesize,
            sep = "")
        anyerror = try(infodata(newpage), silent = T)
        Sys.sleep(2)
        if (class(anyerror) == "try-error") {
            whentostop = F
        } else {
            cstart = cstart + pagesize
        }
    }
    return(cstart - pagesize)
}
```

```r
getallpub = function(citationpage, cstart = 0, pagesize = 20) {
    dataframe = infodata(citationpage)
    cstart = cstart + pagesize
    tostop = getmaxcstart(citationpage)
    while (cstart <= tostop) {
        Sys.sleep(2)
        newpage = paste(citationpage, "&cstart=", cstart, "&pagesize=", pagesize,
            sep = "")
        dataframe = rbind(dataframe, infodata(newpage))
        cstart = cstart + pagesize
    }
    return(dataframe)
}

dim(getallpub(getURLID("Geoffrey Hinton")[1]))
```

```
## [1] 553   5
```