Problem Set 8 Answer Key

Due March 21, 2023

Instructions

- Read all of these instructions closely.
- This problem set is due Tuesday, March 21, 2023 at 4pm.
- Submit files via Github:
  1. the .Rmd (R Markdown) file
  2. the knitted .pdf file
  3. anything else the particular problem set might require
- Use a copy of this file, perhaps with your name or initials appended to the file name, to write your answers to the questions. You'll see there is a designated space where your answers should begin.
- Knitting the .Rmd file to a .pdf file *as you work* will ensure your code runs without errors and is working how you expect. Knit early and often. You've already read the instruction that a knitted .pdf is required when you submit.
- Per the syllabus, I will not accept any late work. Keep in mind the two lowest problem set scores are dropped. Turn in what you have.
- Clarification on the expectations for problem set submissions (posted in Slack, copied here):
  - Always print the output of the code I'm requesting.
    - Ex: If I want you to create a vector x with elements 1 through 10, print x after creating it so I can see it worked.
  - Write any written answers in the space outside the code chunk, not inside with an R comment.
    - R comments are great to clarify code, but not for answering the question.
  - Make sure any code or written content is not cut off in the pdf.
    - This really should only apply to code, because if you follow item 2 in this list, the pdf will compile your written answers nicely.

Overview

In problem set 4, you scraped the Notre Dame Political Science department faculty websites to create a database of their contact information, fields of study, etc. In this problem set, you'll use my copy of that dataset, and we'll practice our regular expression and tidy-data skills.

To start, read in my version of the faculty dataframe. I also used the `select` function to remove columns we will not need in this problem set.

```
library(tidyverse)
library(plyr)
library(dplyr)
faculty_df <- read_csv("faculty_df.csv")
faculty_df <- faculty_df %>%
  select(-link, -title, -office_hours, -phone)
head(faculty_df)
```

```
## # A tibble: 6 x 5
##      id name                fields                                    offic~1 email
##   <dbl> <chr>               <chr>                                     <chr>   <chr>
## 1     1 Christina Bambrick  Fields of Study: Constitutional Studi~ 2020A ~ cbam~
```

```
## 2     2 Sotirios Barber     Fields of Study: Constitutional Studi~ 2049 J~ sbar~
## 3     3 Jaimie Bleck        Fields of Study: Comparative Politics~ 2016 J~ jble~
## 4     4 David Campbell      Fields of Study: American Politics     2082 J~ dave~
## 5     5 Susan D. Collins    Fields of Study: Political Theory, Co~ 2166 J~ SCol~
## 6     6 Michael J. Coppedge Fields of Study: Comparative Politics~ 216 He~ copp~
## # ... with abbreviated variable name 1: office_location
```

Question 1–Regex

1a

Use regular expressions to grab only the user id portion of the faculty member's email. Create a new column called "user_id" with this information. For example, my email is "erossite@nd.edu", so the new variable should only contain "erossite".

In words, describe your regular expression solution. Use the `head` command to show some of the `user_id` variable.

```
#code here
```

Erin answer:

Remember the "^" symbol when used in hard brackets means "anything but". Also remember the "+" symbol means "one or more". Finally, by default, regular expression starts looking at the beginning of the string (although we also could have explicitly told it to do this.) Taken together, this regular expression starts reading characters at the begining of the string and extracts what counts as a match. The match is made up of grabbing one or more characters as long as they aren't the "@" character.

```
faculty_df$user_id <- str_extract(faculty_df$email, pattern = "[^@]+")
head(faculty_df$user_id)
```

```
## [1] "cbambrick"     "sbarber"       "jbleck"        "dave_campbell"
## [5] "SCollin5"      "coppedge.1"
```

```
# Another way, but starting to be redundant
#str_extract(faculty_df$email, pattern = "^.[^@]+")

# Another way
# .+ means number of any character
# (?=@) means followed by the @ symbol
#str_extract(faculty_df$email, pattern = ".+(?=@)")
```

1b

Use regular expressions to grab only the office number, for example, 2020A or 316. Create a new column called `office_num` with this information.

In words, describe your regular expression solution. Use the `head` command to show some of the `office_num` variable.

```
#code here
```

Erin answer:

This regular expression grabs anything that is alphanumeric, thus it stops at the first whitespace.

```
faculty_df$office_num <- str_extract(faculty_df$office_location, pattern = "[[:alnum:]]+")
head(faculty_df$office_num)
```

```
## [1] "2020A" "2049"  "2016"  "2082"  "2166"  "216"
```

1c

This is the trickiest question in the problem set. Use the `str_extract_all` function, and write a regex pattern such that the name(s) of the faculty members' fields are extracted from the `fields` variable.

For example, you should extract one string from Prof Bambrick: "Constitutional Studies". However, you should extract two strings from Prof Bleck: "Comparative Politics" and "Methodology".

Let me know if you get stuck.

Then, use an "apply" function from the `plyr` package to add these information to the `faculty_df` dataset in the form of two new columns named `field1` and `field2`. As a starting point, note that the `str_extract_all` function will output results as a **l**ist.

In words, describe your regular expression solution. In words, describe your regular expression solution. Use the `head` command to show the first few rows the the dataset once these variables are added.

```
#code here
```

Erin answer:

The "(?<=[:punct:]\s)" sets me up to find matches preceded by a punctuation and white space. That is what the (?<=..).. part does.

Now we've established we're only looking *after* the "Fields of Study:" part. Then what counts as a match? Any consecutive set of characters as long as its not a comma "[^,]'

```
fields <- str_extract_all(faculty_df$fields, pattern = "(?<=[:punct:]\\s)[^,]+")
head(fields)
```

```
## [[1]]
## [1] "Constitutional Studies"
##
## [[2]]
## [1] "Constitutional Studies"
##
## [[3]]
## [1] "Comparative Politics" "Methodology"
##
## [[4]]
## [1] "American Politics"
##
## [[5]]
## [1] "Political Theory"       "Constitutional Studies"
##
## [[6]]
## [1] "Comparative Politics" "Methodology"
```

```
# add columns
faculty_df$field1 <- plyr::laply(fields, function(f) f[1])
faculty_df$field2 <- plyr::laply(fields, function(f) f[2])

head(faculty_df)
```

```
## # A tibble: 6 x 9
##      id name               fields   offic~1 email user_id offic~2 field1 field2
##   <dbl> <chr>              <chr>    <chr>   <chr> <chr>   <chr>   <chr>  <chr>
## 1     1 Christina Bambrick Fields ~ 2020A ~ cbam~ cbambr~ 2020A   Const~ <NA>
## 2     2 Sotirios Barber    Fields ~ 2049 J~ sbar~ sbarber 2049    Const~ <NA>
## 3     3 Jaimie Bleck       Fields ~ 2016 J~ jble~ jbleck  2016    Compa~ Metho~
```

```
## 4     4 David Campbell      Fields ~ 2082 J~ dave~ dave_c~ 2082    Ameri~ <NA>
## 5     5 Susan D. Collins    Fields ~ 2166 J~ SCol~ SColli~ 2166    Polit~ Const~
## 6     6 Michael J. Coppedge Fields ~ 216 He~ copp~ copped~ 216     Compa~ Metho~
## # ... with abbreviated variable names 1: office_location, 2: office_num
```

Question 2–pivots

This question uses the `faculty_df` object with the updated columns from Question 1. This question must be completed in order.

2a

For the purposes of this problem set, let's consider the order in which the fields are listed on the website how the faculty "rank" their fields of expertise.

First, pivot the dataset to long format so that each faculty member has two observations, one for their first ranked field and one for their second ranked field. Among other columns, your solution should look something like this:

name	field_rank	field
Christina Bambrick	field1	Constitutional Studies
Christina Bambrick	field2	NA
Jaimie Bleck	field1	Comparative Politics
Jaimie Bleck	field2	Methodology

Print the dimensions of the resulting dataframe and the first few rows.

```
#code here
```

Erin answer:

```
faculty_long <- faculty_df %>%
  pivot_longer(
    cols = c("field1", "field2"),
    names_to = "field_rank",
    values_to = "field"
  )

dim(faculty_long)
```

```
## [1] 100   9
```

```
head(faculty_long)
```

```
## # A tibble: 6 x 9
##      id name              fields    offic~1 email user_id offic~2 field~3 field
##   <dbl> <chr>             <chr>     <chr>   <chr> <chr>   <chr>   <chr>   <chr>
## 1     1 Christina Bambrick Fields o~ 2020A ~ cbam~ cbambr~ 2020A   field1  Cons~
## 2     1 Christina Bambrick Fields o~ 2020A ~ cbam~ cbambr~ 2020A   field2  <NA>
## 3     2 Sotirios Barber    Fields o~ 2049 J~ sbar~ sbarber 2049    field1  Cons~
## 4     2 Sotirios Barber    Fields o~ 2049 J~ sbar~ sbarber 2049    field2  <NA>
## 5     3 Jaimie Bleck       Fields o~ 2016 J~ jble~ jbleck  2016    field1  Comp~
## 6     3 Jaimie Bleck       Fields o~ 2016 J~ jble~ jbleck  2016    field2  Meth~
## # ... with abbreviated variable names 1: office_location, 2: office_num,
## #   3: field_rank
```

2b

Notice that people who do not have a second field listed still result in two rows. Use `dplyr` functions (not base R) with piping to remove those rows. Print the dimensions of the resulting dataframe and the first few rows. You should have 73 rows after cleaning.

Erin answer:

```
faculty_long <- faculty_long %>%
  filter(!is.na(field))

head(faculty_long)
```

```
## # A tibble: 6 x 9
##      id name               fields    offic~1 email user_id offic~2 field~3 field
##   <dbl> <chr>              <chr>     <chr>   <chr> <chr>   <chr>   <chr>   <chr>
## 1     1 Christina Bambrick Fields o~ 2020A ~ cbam~ cbambr~ 2020A   field1  Cons~
## 2     2 Sotirios Barber    Fields o~ 2049 J~ sbar~ sbarber 2049    field1  Cons~
## 3     3 Jaimie Bleck       Fields o~ 2016 J~ jble~ jbleck  2016    field1  Comp~
## 4     3 Jaimie Bleck       Fields o~ 2016 J~ jble~ jbleck  2016    field2  Meth~
## 5     4 David Campbell     Fields o~ 2082 J~ dave~ dave_c~ 2082    field1  Amer~
## 6     5 Susan D. Collins   Fields o~ 2166 J~ SCol~ SColli~ 2166    field1  Poli~
## # ... with abbreviated variable names 1: office_location, 2: office_num,
## #   3: field_rank
```

```
dim(faculty_long)
```

```
## [1] 73  9
```

2c

Using your cleaned long-format dataframe, now pivot back to wide format. Print the dimensions of the resulting dataframe and the first few rows. You should have 50 observations like the original dataframe.

Erin answer:

```
faculty_wide <- faculty_long %>%
  pivot_wider(
    names_from = "field_rank",
    values_from = "field"
  )

head(faculty_df)
```

```
## # A tibble: 6 x 9
##      id name                fields   offic~1 email user_id offic~2 field1 field2
##   <dbl> <chr>               <chr>    <chr>   <chr> <chr>   <chr>   <chr>  <chr>
## 1     1 Christina Bambrick  Fields ~ 2020A ~ cbam~ cbambr~ 2020A   Const~ <NA>
## 2     2 Sotirios Barber     Fields ~ 2049 J~ sbar~ sbarber 2049    Const~ <NA>
## 3     3 Jaimie Bleck        Fields ~ 2016 J~ jble~ jbleck  2016    Compa~ Metho~
## 4     4 David Campbell      Fields ~ 2082 J~ dave~ dave_c~ 2082    Ameri~ <NA>
## 5     5 Susan D. Collins    Fields ~ 2166 J~ SCol~ SColli~ 2166    Polit~ Const~
## 6     6 Michael J. Coppedge Fields ~ 216 He~ copp~ copped~ 216     Compa~ Metho~
## # ... with abbreviated variable names 1: office_location, 2: office_num
```

```
dim(faculty_wide)
```

```
## [1] 50  9
```