week14.Rmd

Phoebe Hessen

4/21/2020

Libraries

```
library(tidyverse)
```

```
## -- Attaching packages ---------------------------------------------------------------------------------------
```

```
## v ggplot2 3.2.1     v purrr   0.3.3
## v tibble  2.1.3     v dplyr   0.8.3
## v tidyr   1.0.0     v stringr 1.4.0
## v readr   1.3.1     v forcats 0.4.0
```

```
## -- Conflicts ------------------------------------------------------------------------------------------------
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
library(RMariaDB)
```

Data Import and Cleaning

I began by gaining access to the server.

```
con <- dbConnect( MariaDB(),
                  user="rnlander_8960r",
                  password="rTCo.4vQ2vc-",
                  host="tntlab.org"
)
```

Then, I used the following command to see which databases I have access to and the priveleges that I have.

```
dbGetQuery(con, "SHOW DATABASES")
```

```
##             Database
## 1 information_schema
## 2      rnlander_8960
```

```
dbGetQuery(con, "SHOW GRANTS")
```

```
##                                                 Grants for rnlander_8960r@199.21.%.%
## 1 GRANT USAGE ON *.* TO 'rnlander_8960r'@'199.21.%.%' IDENTIFIED BY PASSWORD <secret>
## 2                GRANT SELECT ON `rnlander\\_8960`.* TO 'rnlander_8960r'@'199.21.%.%'
```

The databases I have access to are called "information_schema" and "rnlander_8960".

I know that the "information_Schema" database contains meta-data, so I want to explore the "rnlander_8960" database.

```
dbExecute(con, "USE rnlander_8960")
```

```
## [1] 0
```

```
dbGetQuery(con, "SHOW TABLES")
```

```
##   Tables_in_rnlander_8960
## 1                   demos
## 2               responses
## 3             socialmedia
```

There are three tables in the "rnlander_8960" database: demos, responses, and socialmedia. I want to see what columns are in each table.

```
dbExecute(con, "USE rnlander_8960")
```

```
## [1] 0
```

```
dbGetQuery(con, "SHOW COLUMNS FROM demos")
```

```
##             Field                  Type Null Key Default Extra
## 1 participant_num mediumint(8) unsigned  YES        <NA>
## 2             age           varchar(50)  YES        <NA>
## 3             sex           varchar(50)  YES        <NA>
```

```
dbGetQuery(con, "SHOW COLUMNS FROM responses")
```

```
##          Field                  Type Null Key Default Extra
## 1        ident mediumint(8) unsigned  YES        <NA>
## 2       device           varchar(50)  YES        <NA>
## 3     smu_code            tinyint(4)  YES        <NA>
## 4   rec_events           varchar(50)  YES        <NA>
## 5 rec_products           varchar(50)  YES        <NA>
## 6  rec_friends           varchar(50)  YES        <NA>
## 7 rec_policial           varchar(50)  YES        <NA>
```

```
dbGetQuery(con, "SHOW COLUMNS FROM socialmedia")
```

```
##       Field        Type Null Key Default Extra
## 1      code     int(11)  YES        <NA>
## 2  facebook varchar(50)  YES        <NA>
## 3   twitter varchar(50)  YES        <NA>
## 4 instagram varchar(50)  YES        <NA>
## 5   youtube varchar(50)  YES        <NA>
## 6  snapchat varchar(50)  YES        <NA>
## 7     other varchar(50)  YES        <NA>
```

The information that I want to obtain is 4 survey responses, a count of social media platforms the participant uses, and the age of the participant. These pieces of information appear to be contained in responses (cols 4-7), socialmedia (cols 2-7) and demos (col 2).

My next step is to find out how I can match resposnes across these datasets, to join all the information I need into a single table. From the columns names, it seems that "participant_num" from demos and "ident" from responses might be matched across participants. I want to look at them and see if they take similar form.

```
dbExecute(con, "USE rnlander_8960")
```

```
## [1] 0
```

```
dbGetQuery(con, "SELECT participant_num FROM demos LIMIT 10")
```

```
##    participant_num
## 1           351752
## 2           738262
## 3           725007
## 4           497638
## 5           725505
## 6           640505
## 7           370355
## 8           391720
## 9           164906
## 10          491943
```

```
dbGetQuery(con, "SELECT ident FROM responses LIMIT 10")
```

```
##     ident
## 1  100585
## 2  100712
## 3  100797
## 4  100831
## 5  100870
## 6  100885
## 7  101211
## 8  101221
## 9  101459
## 10 101465
```

These variables appear to be in the same format.

socialmedia doesn't appear to have a participant number, but it does have a "code" column that could be matched to "smu_code" in responses.

```
dbExecute(con, "USE rnlander_8960")
```

```
## [1] 0
```

```
dbGetQuery(con, "SELECT code FROM socialmedia LIMIT 10")
```

```
##    code
## 1     0
## 2     1
## 3     2
## 4     3
## 5     4
## 6     5
## 7     6
## 8     7
## 9     8
## 10    9
```

```
dbGetQuery(con, "SELECT smu_code FROM responses LIMIT 10")
```

```
##    smu_code
## 1         8
## 2         0
## 3        13
## 4        32
## 5        31
## 6         8
## 7        40
## 8         1
## 9        39
## 10        1
```

These variables also appear to be in the same format. I now know how I can match information from the three tables. Because responses has both participant numbers to match with demos, and social media codes to match with socialmedia, I will join information from those tables into responses.

Tidyverse path

Since pieces of the information I am interested in are contained in each of the three tables in the database, I will import all three tables.

```
dbExecute(con, "USE rnlander_8960")
```

```
## [1] 0
```

```
demos <- dbGetQuery(con, "SELECT * FROM demos")
responses <- dbGetQuery(con, "SELECT * FROM responses")
social_media <- dbGetQuery(con, "SELECT * FROM socialmedia")
```

Now I have to create a final dataset, which contains the columns that I want from each of the three tables. I can do this by joining on the previously identified variables.

```
tidy_tbl <- responses %>%
    left_join(social_media, by = c("smu_code" = "code")) %>%
    left_join(demos, by = c("ident" = "participant_num"))
```

Now I have a tidy dataset that contains all the raw variables I am interested in for my analysis.

SQL Only Path

To do the same thing in SQL only:

Now that I have identified the variables that cases are matched on, I can select my columns and perform joins to gather my dataset.

```
dbExecute(con, "USE rnlander_8960")
```

```
## [1] 0
```

```
sql_tbl <- dbGetQuery(con, "SELECT * FROM responses
           LEFT JOIN socialmedia
            ON smu_code = code
           LEFT JOIN demos
            ON ident = participant_num;")
```

Now I once again have a tidy dataset with all of my raw variables.

Final Cleaning

First, I tidied the dataset in a way that allowed me to perform numeric operations on the variables that I would need to in later steps.

```
tidy_tbl <- tidy_tbl %>%
    mutate_at(vars(4:7), na_if, "Refused") %>%
    mutate_at(vars(4:7,14), na_if, "NA") %>%
    mutate_at(vars(4:7), recode, "Not acceptable at all" = 1,
              "Not very acceptable" = 2,
              "Somewhat acceptable" = 3,
              "Very acceptable" = 4) %>%
    mutate_at(vars(8:13), recode, "Not selected" = 0, .default = 1) %>%
    mutate_at(vars(4:13), as.numeric) %>%
    mutate_at(14, as.factor)
```

Then, I added a column that held the mean of the privacy questions and a column that held the sum of the binary coded social media questions.

```
tidy_tbl$privacy <- rowMeans(tidy_tbl[startsWith(names(tidy_tbl),"rec_")], na.rm = TRUE)
tidy_tbl$social_media <- rowSums(tidy_tbl[8:13])
```

Finally, I narrowed my dataset down to only the columns I am interested in for my analysis. I also removed cases where the participant did not have any social media, because there was no privacy data for those participants, making analysis of those cases nonsensical.

```r
tidy_tbl <- select(tidy_tbl, c(age, privacy, social_media))
tidy_tbl <- tidy_tbl[complete.cases(tidy_tbl), ]
```

The dataset is now clean and ready for analysis.

Analysis

The research question I am interested in is whether there is a relationship between number of social media platforms used and acceptance of privacy intrusions, and whether this relationship is moderated by age. To begin, I fit an OLS model predicting privacy based on social_media.

```r
mod1 <- lm(privacy ~ social_media, tidy_tbl)
summary(mod1)
```

```
##
## Call:
## lm(formula = privacy ~ social_media, data = tidy_tbl)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.87508 -0.50311 -0.00109  0.49689  1.74488
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)  2.131130   0.023694   89.94   <2e-16 ***
## social_media 0.123992   0.008892   13.95   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.7297 on 4299 degrees of freedom
## Multiple R-squared:  0.04328,    Adjusted R-squared:  0.04305
## F-statistic: 194.5 on 1 and 4299 DF,  p-value: < 2.2e-16
```

The estimate of the coefficient for social media is 0.12, and the p-value is quite small (less than .001), indicating that there is a significant positive relationship between how many social media platforms a person has, and how accepting they are of privacy intrusions.

Next, I fit two OLS models, both with age and social media as predictors, and one including their interaction and the other not including their interaction.

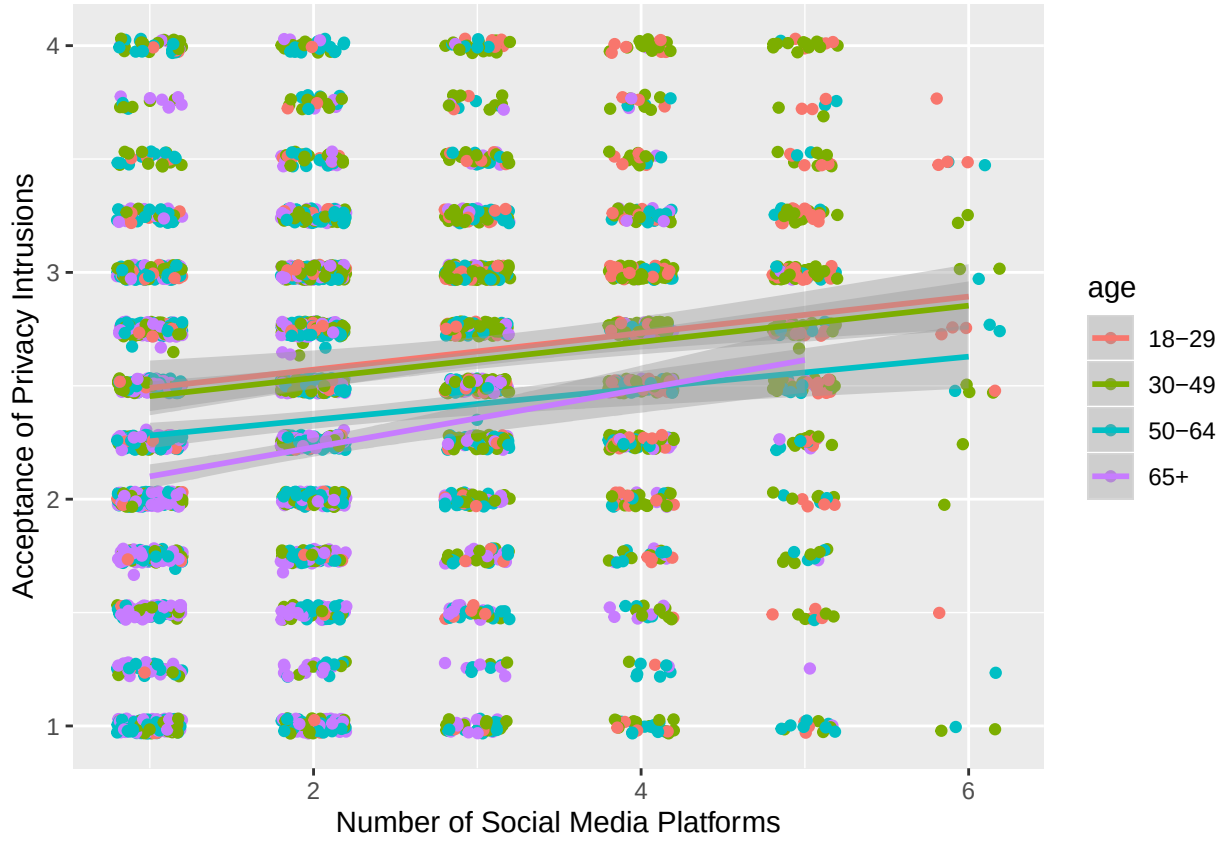
```r
mod2 <- lm(privacy ~ social_media + age, tidy_tbl)
mod3 <- lm(privacy ~ social_media + age + social_media*age, tidy_tbl)
aov.mod <- anova(mod2, mod3)
aov.mod
```

```
## Analysis of Variance Table
##
## Model 1: privacy ~ social_media + age
## Model 2: privacy ~ social_media + age + social_media * age
##   Res.Df    RSS Df Sum of Sq      F Pr(>F)
## 1   4296 2223.3
## 2   4293 2220.9  3    2.3674 1.5254 0.2058
```

Running an ANOVA on these two models demonstrates that there is not a significant difference in predictive power between the two models. This suggests that the relationship between number of social media platforms used and acceptance of privacy intrusions is not moderated by age.

Visualization

```
ggplot(tidy_tbl, aes(x = social_media, y = privacy, color = age)) +
                    geom_jitter(width = 0.2) +
                    geom_smooth(method = "lm", data = tidy_tbl) +
                    labs(x = "Number of Social Media Platforms", y = "Acceptance of Privacy Intrusions")
```



This plot further supports the idea that the relationship between social media use and acceptance of privacy intrusions is not moderated by age - the slopes for the different age categories appear very similar.