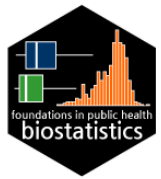
Computing and Interpreting Chi-Squared

Descriptives & graphs for 2 categorical variables

Jenine Harris
Brown School



Understanding the relationship between two categorical variables

- Chi-squared is an inferential statistical test to determine whether there is a relationship between two categorical variables.
- Before conducting any sort of inferential analyses that use sample data to understand a population, it is a best practice to get to know the sample data using descriptive statistics and graphics.
- This step in research is often called **exploratory data analysis** or **EDA**.
- The descriptive statistics for EDA examining categorical variables are frequencies and percentages.

Importing data files from SPSS formats

- The voting data set is saved with a **.sav** file extension, which meant it is **SPSS** formatted data.
- There are several ways to import SPSS data into R, including with the haven package, which works well with tidyverse.
- Import the data using `read_sav()` from the haven package and select the variables of interest: `pew1a` `pew1b`, `race`, `sex`, `mstatus` (marital status), `ownhome` (home ownership), `employ`, and `polparty` (political party affiliation).

```
# import the April 17-23 Pew Research Center data
library(package = "haven")

# import the voting data
vote <- read_sav(file = "/Users/harrisj/Box/teaching/Teaching/Fall2020/d

# select variables of interest
vote.cleaned <- vote %>%
  select(pew1a, pew1b, race, sex, mstatus, ownhome, employ, polparty)
```

Check and clean data

```
# check data
summary(object = vote.cleaned)
```

```
##      pew1a           pew1b            race             sex       
##  Min.   :1.000   Min.   :1.000   Min.   : 1.000   Min.   :1.000  
##  1st Qu.:1.000   1st Qu.:2.000   1st Qu.: 1.000   1st Qu.:1.000  
##  Median :2.000   Median :2.000   Median : 1.000   Median :2.000  
##  Mean   :1.717   Mean   :1.897   Mean   : 4.482   Mean   :1.518  
##  3rd Qu.:2.000   3rd Qu.:2.000   3rd Qu.: 2.000   3rd Qu.:2.000  
##  Max.   :9.000   Max.   :9.000   Max.   :99.000   Max.   :2.000  
##     mstatus         ownhome          employ         polparty    
##  Min.   :1.000   Min.   :1.000   Min.   :1.000   Min.   :0.000  
##  1st Qu.:2.000   1st Qu.:1.000   1st Qu.:1.000   1st Qu.:1.000  
##  Median :3.000   Median :1.000   Median :2.000   Median :2.000  
##  Mean   :3.219   Mean   :1.481   Mean   :2.527   Mean   :2.475  
##  3rd Qu.:5.000   3rd Qu.:2.000   3rd Qu.:3.000   3rd Qu.:3.000  
##  Max.   :9.000   Max.   :9.000   Max.   :9.000   Max.   :9.000
```

- `read_sav()` imports the file in a "haven labelled" format, which is a little tricky to use and changing the variables to factors would result in using the current labels which are extremely long
- Add `zap_labels()` to the data management to remove the super-long labels.
- Use `recode_factor()` to recode `pew1a` and `pew1b` starting with giving them more descriptive variable names like `ease.vote` and `require.vote`.

Data cleaning code

```
# open tidyverse
library(package = "tidyverse")

# select variables of interest and clean them
vote.cleaned <- vote %>%
  select(pew1a, pew1b, race, sex, mstatus, ownhome, employ, polparty, ed
  zap_labels() %>%
  mutate(pew1a = recode_factor(.x = pew1a,
                               `1` = 'Register to vote',
                               `2` = 'Make easy to vote',
                               `5` = NA_character_,
                               `9` = NA_character_)) %>%
  rename(ease.vote = pew1a) %>%
  mutate(pew1b = recode_factor(.x = pew1b,
                               `1` = 'Require to vote',
                               `2` = 'Choose to vote',
                               `5` = NA_character_,
                               `9` = NA_character_)) %>%
  rename(require.vote = pew1b)
```

Check cleaned data

```
# check voting variables
summary(object = vote.cleaned)
```

```
##              ease.vote              require.vote      race             sex
##  Register to vote :408   Require to vote:205   Min.   : 1.000   Min.   :1.00
##  Make easy to vote:593   Choose to vote :806   1st Qu.: 1.000   1st Qu.:1.00
##  NA's             : 27   NA's           : 17   Median : 1.000   Median :2.00
##                                                Mean   : 4.482   Mean   :1.51
##                                                3rd Qu.: 2.000   3rd Qu.:2.00
##                                                Max.   :99.000   Max.   :2.00
##     mstatus         ownhome          employ         polparty
##  Min.   :1.000   Min.   :1.000   Min.   :1.000   Min.   :0.000
##  1st Qu.:2.000   1st Qu.:1.000   1st Qu.:1.000   1st Qu.:1.000
##  Median :3.000   Median :1.000   Median :2.000   Median :2.000
##  Mean   :3.219   Mean   :1.481   Mean   :2.527   Mean   :2.475
##  3rd Qu.:5.000   3rd Qu.:2.000   3rd Qu.:3.000   3rd Qu.:3.000
##  Max.   :9.000   Max.   :9.000   Max.   :9.000   Max.   :9.000
##       educ            income
##  Min.   : 1.000   Min.   : 1.00
##  1st Qu.: 3.000   1st Qu.: 3.00
##  Median : 4.000   Median : 6.00
##  Mean   : 5.548   Mean   :17.06
##  3rd Qu.: 6.000   3rd Qu.:12.00
##  Max.   :99.000   Max.   :99.00
```

More data cleaning code

```
# open tidyverse
library(package = "tidyverse")

# select variables of interest and clean them
vote.cleaned <- vote %>%
  select(pew1a, pew1b, race, sex, mstatus, ownhome, employ, polparty, ed
  zap_labels() %>%
  mutate(pew1a = recode_factor(.x = pew1a,
                               `1` = 'Register to vote',
                               `2` = 'Make easy to vote',
                               `5` = NA_character_,
                               `9` = NA_character_)) %>%
  rename(ease.vote = pew1a) %>%
  mutate(pew1b = recode_factor(.x = pew1b,
                               `1` = 'Require to vote',
                               `2` = 'Choose to vote',
                               `5` = NA_character_,
                               `9` = NA_character_)) %>%
  rename(require.vote = pew1b)
```

Check variable coding

```
# check voting variables
summary(object = vote.cleaned)
```

```
##              ease.vote              require.vote      race             sex
##  Register to vote :408   Require to vote:205   Min.   : 1.000   Min.   :1.00
##  Make easy to vote:593   Choose to vote :806   1st Qu.: 1.000   1st Qu.:1.00
##  NA's             : 27   NA's           : 17   Median : 1.000   Median :2.00
##                                                Mean   : 4.482   Mean   :1.51
##                                                3rd Qu.: 2.000   3rd Qu.:2.00
##                                                Max.   :99.000   Max.   :2.00
##     mstatus         ownhome          employ         polparty
##  Min.   :1.000   Min.   :1.000   Min.   :1.000   Min.   :0.000
##  1st Qu.:2.000   1st Qu.:1.000   1st Qu.:1.000   1st Qu.:1.000
##  Median :3.000   Median :1.000   Median :2.000   Median :2.000
##  Mean   :3.219   Mean   :1.481   Mean   :2.527   Mean   :2.475
##  3rd Qu.:5.000   3rd Qu.:2.000   3rd Qu.:3.000   3rd Qu.:3.000
##  Max.   :9.000   Max.   :9.000   Max.   :9.000   Max.   :9.000
##       educ            income
##  Min.   : 1.000   Min.   : 1.00
##  1st Qu.: 3.000   1st Qu.: 3.00
##  Median : 4.000   Median : 6.00
##  Mean   : 5.548   Mean   :17.06
##  3rd Qu.: 6.000   3rd Qu.:12.00
##  Max.   :99.000   Max.   :99.00
```

Other variable cleaning

- Recode `race`, `sex`, and `ownhome` variables next.

```
# examine race variable before recoding
table(vote.cleaned$race)
```

```
##
##   1   2   3   4   5   6   7   8   9  10  99
## 646 129  63  26  61  22  19   2   1  34  25
```

- The codebook does not match the data from Pew; the variable in the data set has 11 categories and the codebook shows 8 categories.
- The variable was also called `race` in the data set and `race1` in the codebook.

Figuring out what is wrong

- Pew included a text file called *readme.txt* with the downloaded data that included recommendations from the Pew Research Center on how to recode race.
- Although the recommended recoding used a different coding language than R, it is not too difficult to figure out what to do.
- The code suggested that Pew recommended recoding into five categories of race-ethnicity:
    - White non-Hispanic
    - Black non-Hispanic
    - Hispanic
    - Other
    - Don't know/Refused

Recoding variables before analysis

- Decide whether to code Don't know/Refused as `NA`.

```
# select variables of interest and clean them
vote.cleaned <- vote %>%
  select(pew1a, pew1b, race, sex, mstatus, ownhome, employ, polparty, ed
  zap_labels() %>%
  mutate(pew1a = recode_factor(.x = pew1a,
                               `1` = 'Register to vote',
                               `2` = 'Make easy to vote',
                               `5` = NA_character_,
                               `9` = NA_character_)) %>%
  rename(ease.vote = pew1a) %>%
  mutate(pew1b = recode_factor(.x = pew1b,
                               `1` = 'Require to vote',
                               `2` = 'Choose to vote',
                               `5` = NA_character_,
                               `9` = NA_character_)) %>%
  rename(require.vote = pew1b) %>%
  mutate(race = recode_factor(.x = race,
                              `1` = 'White non-Hispanic',
                              `2` = 'Black non-Hispanic',
                              `3` = 'Hispanic',
                              `4` = 'Hispanic',
                              `5` = 'Hispanic',
```

Check the recoding

```
# check recoding
summary(vote.cleaned)
```

```
##              ease.vote           require.vote                    race
##  Register to vote :408   Require to vote:205   White non-Hispanic:646
##  Make easy to vote:593   Choose to vote :806   Black non-Hispanic:129
##  NA's             : 27   NA's           : 17   Hispanic           :150
##                                                 Other              : 78
##                                                 NA's               : 25
##
##      sex          mstatus         ownhome         employ          polparty
##  Male  :495   Min.   :1.000   Owned :678   Min.   :1.000   Min.   :0.000
##  Female:533   1st Qu.:2.000   Rented:328   1st Qu.:1.000   1st Qu.:1.000
##               Median :3.000   NA's  : 22   Median :2.000   Median :2.000
##               Mean   :3.219                Mean   :2.527   Mean   :2.475
##               3rd Qu.:5.000                3rd Qu.:3.000   3rd Qu.:3.000
##               Max.   :9.000                Max.   :9.000   Max.   :9.000
##       educ             income
##  Min.   : 1.000   Min.   : 1.00
##  1st Qu.: 3.000   1st Qu.: 3.00
##  Median : 4.000   Median : 6.00
##  Mean   : 5.548   Mean   :17.06
##  3rd Qu.: 6.000   3rd Qu.:12.00
##  Max.   :99.000   Max.   :99.00
```

Descriptive statistics for two categorical variables

- Using `tidyverse` is an option.

```
# voting ease by race-eth
vote.cleaned %>%
  drop_na(ease.vote) %>%
  drop_na(race) %>%
  group_by(ease.vote, race) %>%
  summarize(freq.n = n())
```

```
## # A tibble: 8 x 3
## # Groups:   ease.vote [2]
##   ease.vote         race                freq.n
##   <fct>             <fct>                <int>
## 1 Register to vote  White non-Hispanic     292
## 2 Register to vote  Black non-Hispanic      28
## 3 Register to vote  Hispanic                51
## 4 Register to vote  Other                   27
## 5 Make easy to vote White non-Hispanic     338
## 6 Make easy to vote Black non-Hispanic      98
## 7 Make easy to vote Hispanic                97
## 8 Make easy to vote Other                   46
```

Using spread to format

- The `spread()` function can help with format:

```
# voting ease by race-eth with spread
vote.cleaned %>%
  drop_na(ease.vote) %>%
  drop_na(race) %>%
  group_by(ease.vote, race) %>%
  summarize(freq.n = n()) %>%
  spread(key = race, value = freq.n)
```

```
## # A tibble: 2 x 5
## # Groups:   ease.vote [2]
##   ease.vote         `White non-Hispanic` `Black non-Hispanic` Hispanic Other
##   <fct>                            <int>                <int>    <int> <int>
## 1 Register to vote                   292                   28       51    27
## 2 Make easy to vote                  338                   98       97    46
```

Using `table()` instead of tidyverse

```
# voting ease by race-eth with table
table(vote.cleaned$ease.vote, vote.cleaned$race)
```

```
##
##                     White non-Hispanic Black non-Hispanic Hispanic Other
##   Register to vote                  292                  28       51    27
##   Make easy to vote                 338                  98       97    46
```

- Adding options to `table()` and `prop.table()`

```
# table of percents voting ease by race-eth
prop.table(x = table(Voting.ease = vote.cleaned$ease.vote,
                     Race.eth = vote.cleaned$race))
```

```
##                     Race.eth
## Voting.ease          White non-Hispanic Black non-Hispanic   Hispanic      Ot
##   Register to vote          0.29887410         0.02865916 0.05220061 0.02763
##   Make easy to vote         0.34595701         0.10030706 0.09928352 0.04708
```

Getting row or column percents in `prop.table()`

```
# table of percents voting ease by race-eth
prop.table(x = table(Voting.ease = vote.cleaned$ease.vote,
                     Race.eth = vote.cleaned$race),
           margin = 2)
```

```
##                     Race.eth
## Voting.ease          White non-Hispanic Black non-Hispanic  Hispanic     Othe
##   Register to vote           0.4634921           0.2222222 0.3445946 0.369863
##   Make easy to vote          0.5365079           0.7777778 0.6554054 0.630137
```

```
# table of percents voting required by race-eth
prop.table(x = table(Voting.requirement = vote.cleaned$require.vote,
                     Race.eth = vote.cleaned$race),
           margin = 2)
```

```
##                    Race.eth
## Voting.requirement White non-Hispanic Black non-Hispanic  Hispanic     Other
##    Require to vote          0.1502347           0.3228346 0.3401361 0.189189
##    Choose to vote           0.8497653           0.6771654 0.6598639 0.810810
```

Using graphs to examine two categorical variables

- Bar graphs for two categorical variables works well.

```
# graph the relationship between registration ease and race eth
ease.graph <- vote.cleaned %>%
  drop_na(ease.vote) %>%
  drop_na(race) %>%
  group_by(ease.vote, race) %>%
  count() %>%
  group_by(race) %>%
  mutate(perc = 100*n/sum(n)) %>%
  ggplot(aes(x = race, y = perc, fill = ease.vote)) +
  geom_bar(position = "dodge", stat = "identity") +
  theme_minimal() +
  scale_fill_manual(values = c("gray", "#7463AC"), name = "Opinion on\nv
  labs(x = "", y = "Percent within group") +
  theme(axis.text.x = element_blank())
```

Using graphs to examine two categorical variables

- Bar graphs for two categorical variables works well.

```
# graph the relationship between registration ease and race eth
ease.graph <- vote.cleaned %>%
  drop_na(ease.vote) %>%
  drop_na(race) %>%
  group_by(ease.vote, race) %>%
  count() %>%
  group_by(race) %>%
  mutate(perc = 100*n/sum(n)) %>%
  ggplot(aes(x = race, y = perc, fill = ease.vote)) +
  geom_bar(position = "dodge", stat = "identity") +
  theme_minimal() +
  scale_fill_manual(values = c("gray", "#7463AC"), name = "Opinion on\nv
  labs(x = "", y = "Percent within group") +
  theme(axis.text.x = element_blank())
```

Another bar graph

```
# graph the relationship between required voting and race eth
req.graph <- vote.cleaned %>%
  drop_na(require.vote) %>%
  drop_na(race) %>%
  group_by(require.vote, race) %>%
  count() %>%
  group_by(race) %>%
  mutate(perc = 100*n/sum(n)) %>%
  ggplot(aes(x = race, y = perc, fill = require.vote)) +
  geom_bar(position = "dodge", stat = "identity") +
  theme_minimal() +
  scale_fill_manual(values = c("gray", "#7463AC"), name = "Opinion on vo
  labs(x = "Race-ethnicity group", y = "Percent within group")
```

Using gridExtra to combine plots

```
# open gridExtra to put graphs together
library(package = "gridExtra")
grid.arrange(ease.graph, req.graph, nrow = 2)
```