## Getting Started

Load the csv file created from the GSS.R file. Save it as a data frame called `review` and load the usual packages.

```
review <- read.csv("https://raw.githubusercontent.com/mjclawrence/soci385/master/data/fi
library(tidyverse)
library(pander)
library(stargazer)
```

## Clean Up

Start by making all the variable names lower case:

```
names(review) <- tolower(names(review))
```

The big thing to do before starting analyses is to code all missing values as NA. Look to the do file or the excel file downloaded from the GSS website to see how missing values are coded for each variable.

You can find the codes for missing values of the `age` variable here: https://gssdataexplorer.norc.org/variables/53/vshow. Create a table of the `age` variable to see if there are missing values:

## REPLACE THIS LINE WITH YOUR CODE

```
table(review$age)
```

```
##
##  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35
##  57 128 110 150 162 176 162 226 178 212 189 223 248 217 220 218 226 216
##  36  37  38  39  40  41  42  43  44  45  46  47  48  49  50  51  52  53
## 195 234 197 226 199 208 202 209 199 170 195 193 176 199 214 214 205 237
##  54  55  56  57  58  59  60  61  62  63  64  65  66  67  68  69  70  71
## 214 227 227 214 227 223 216 197 194 192 155 189 157 177 154 152 150 132
##  72  73  74  75  76  77  78  79  80  81  82  83  84  85  86  87  88  89
## 101  92 110  94 102  97  88  78  80  66  54  58  59  40  52  36  38 105
##  99
##  34
```

There are 34 observations with a missing value code for age. Here's how we can make them NAs using `na_if` (which is efficient if there is only one missing value):

```
review <- review %>%
     mutate(age = na_if(age, 99))
```

Now those values should be removed from the table.

```
table(review$age)
```

```
##
##  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35
##  57 128 110 150 162 176 162 226 178 212 189 223 248 217 220 218 226 216
##  36  37  38  39  40  41  42  43  44  45  46  47  48  49  50  51  52  53
## 195 234 197 226 199 208 202 209 199 170 195 193 176 199 214 214 205 237
##  54  55  56  57  58  59  60  61  62  63  64  65  66  67  68  69  70  71
## 214 227 227 214 227 223 216 197 194 192 155 189 157 177 154 152 150 132
##  72  73  74  75  76  77  78  79  80  81  82  83  84  85  86  87  88  89
## 101  92 110  94 102  97  88  78  80  66  54  58  59  40  52  36  38 105
```

Here are examples of how to replace missing values using `ifelse` (which is efficient if there are multiple missing values):

```
review <- review %>%
     mutate(educ = ifelse(educ==0 | educ>=98, NA, educ),
            consci = ifelse(consci==0 | consci>=8, NA, consci),
            intspace = ifelse(intspace == 0 | intspace>=8,
                              NA, intspace))
```

Try replacing the missing values for the `hispanic` variable.

### REPLACE THIS LINE WITH YOUR CODE

```
review <- review %>%
     mutate(hispanic = ifelse(hispanic==0 | hispanic>=98,
                              NA, hispanic))
```

Let's combine values from the `race` and `hispanic` variables to make a new variable called `racehisp`. The easiest way to do this is to first make a binary variable distinguishing those who are in any hispanic category from those who are not.

```
review <- review %>%
     mutate(anyhispanic = ifelse(hispanic>1, 1, 0))
```

Now we can combine values from this new `anyhispanic` variable and the race variable to create the racehisp categories:

```
review <- review %>%
     mutate(racehisp = ifelse(anyhispanic==1, "Hispanic", race),
            racehisp = factor(racehisp,
                              labels = c("White", "Black",
                                         "Other", "Hispanic")))
```

Sometimes it's easier to create new variables instead of changing the values and labels of existing variables. Here we'll create new variables called `science` (taking the values of `consci`) and `space` (taking the values of `intspace`).

```
review <- review %>%
     mutate(science = factor(consci,
                             levels = c(3, 2, 1),
                             labels = c("Hardly any", "Only some",
                                        "A great deal")),
            space = factor(intspace,
                           levels = c(3, 2, 1),
                           labels = c("Not interested",
                                      "Moderately interested",
                                      "Very interested")))
```

We can also collapse existing categories into bigger categories. We'll use the `relig16` variable as an example, creating a new variable called `religion` with broader categories.

```
table(review$relig16)
```

```
##
##    1    2    3    4    5    6    7    9   10   11   12   13   98   99
## 6102 3827  199 1007   64   51   60   74   55  244    9    4   34   41
```

First let's replace the missing values!

```
review <- review %>%
     mutate(relig16 = ifelse(relig16>=98, NA, relig16))
```

Here we'll put all the respondents with values of 6-9 in the "Eastern" category, and those who are not Eastern, Protestant, Catholic, Jewish, or None in "Other":

```
review <- review %>%
     mutate(religion = ifelse(relig16 == 1, "Protestant",
          ifelse(relig16 == 2, "Catholic",
          ifelse(relig16 == 3, "Jewish",
          ifelse(relig16 == 4, "None",
          ifelse(relig16 %in% 6:9, "Eastern", "Other"))))),
               religion = factor(religion,
                    levels = c("Protestant", "Catholic", "Jewish",
                               "Eastern", "Other", "None")))
```

## Three Way Table

For each religious category, we want to know the proportion with each level of confidence in science who are in each category of interest in space. One way to do this is with `group_by()` and `summarize()`. For that approach, we would need binary variables for each of the `space`

categories. This might seem tedious, but in the long run it is more efficient since it will allow you to manipulate the variables for other purposes later.

```
review <- review %>%
     mutate(space_not_interested =
                 ifelse(space=="Not interested",1,0),
            space_moderately_interested =
                 ifelse(space=="Moderately interested",1,0),
            space_very_interested =
                 ifelse(space=="Very interested",1,0))
```

For each combination of `religion` and `science`, we can now summarize the means of each space binary variable (which represent the proportion of respondents in the related category of space interest):

```
space_summary <- review %>%
     group_by(religion, science) %>%
     summarize(not_interested =
                    round(mean(space_not_interested, na.rm=TRUE),3),
               moderately_interested =
                    round(mean(space_moderately_interested, na.rm=TRUE),3),
               very_interested =
                    round(mean(space_very_interested, na.rm=TRUE),3))
```

```
## Warning: Factor `religion` contains implicit NA, consider using
## `forcats::fct_explicit_na`
```

```
## Warning: Factor `science` contains implicit NA, consider using
## `forcats::fct_explicit_na`
```

```
space_summary
```

```
## # A tibble: 28 x 5
## # Groups:   religion [7]
##    religion   science    not_interested moderately_interes~ very_interested
##    <fct>      <fct>               <dbl>               <dbl>           <dbl>
##  1 Protestant Hardly any          0.596               0.288           0.116
##  2 Protestant Only some           0.354               0.493           0.153
##  3 Protestant A great d~          0.207               0.474           0.319
##  4 Protestant <NA>                0.342               0.459           0.2
##  5 Catholic   Hardly any          0.585               0.264           0.151
##  6 Catholic   Only some           0.346               0.47            0.184
##  7 Catholic   A great d~          0.22                0.445           0.335
##  8 Catholic   <NA>                0.313               0.449           0.238
##  9 Jewish     Hardly any          0.5                 0.5             0
## 10 Jewish     Only some           0.161               0.581           0.258
## # ... with 18 more rows
```

Those NAs for `science` and `religion` are annoying. One way to get rid of them is to filter them out. You can do that with an extra line in the chunk above. But we'll redo the whole chunk to compare them, though note it's not necessary to run this twice:

```
space_summary <- review %>%
     filter(!is.na(science), !is.na(religion)) %>%
     #Keep (filter) the observations that are not na for science or religion
     group_by(religion, science) %>%
     summarise(not_interested = round(mean(space_not_interested,
                                           na.rm=TRUE),3),
               moderately_interested = round(mean(space_moderately_interested,
                                            na.rm=TRUE),3),
               very_interested = round(mean(space_very_interested,
                                           na.rm=TRUE),3))

space_summary
```

```
## # A tibble: 18 x 5
## # Groups:   religion [6]
##    religion   science    not_interested moderately_interes~ very_interested
##    <fct>      <fct>               <dbl>               <dbl>           <dbl>
##  1 Protestant Hardly any          0.596               0.288           0.116
##  2 Protestant Only some           0.354               0.493           0.153
##  3 Protestant A great d~          0.207               0.474           0.319
##  4 Catholic   Hardly any          0.585               0.264           0.151
##  5 Catholic   Only some           0.346               0.47            0.184
##  6 Catholic   A great d~          0.22                0.445           0.335
##  7 Jewish     Hardly any          0.5                 0.5             0
##  8 Jewish     Only some           0.161               0.581           0.258
##  9 Jewish     A great d~          0.167               0.633           0.2
## 10 Eastern    Hardly any          0.5                 0               0.5
## 11 Eastern    Only some           0.2                 0.4             0.4
## 12 Eastern    A great d~          0.2                 0.333           0.467
## 13 Other      Hardly any          0.4                 0.6             0
## 14 Other      Only some           0.472               0.321           0.208
## 15 Other      A great d~          0.184               0.469           0.347
## 16 None       Hardly any          0.455               0.242           0.303
## 17 None       Only some           0.352               0.496           0.152
## 18 None       A great d~          0.23                0.378           0.393
```

You can clean up the column names of this table and pander it before you knit. Note that you can also add a table caption in the pander function.

Table 1: Interest in Space Exploration by Religion and Confidence in Science

| Religion | Confidence In Science | Not Interested | Moderately Interested | Very Interested |
|---|---|---|---|---|
| Protestant | Hardly any | 0.596 | 0.288 | 0.116 |
| Protestant | Only some | 0.354 | 0.493 | 0.153 |
| Protestant | A great deal | 0.207 | 0.474 | 0.319 |
| Catholic | Hardly any | 0.585 | 0.264 | 0.151 |
| Catholic | Only some | 0.346 | 0.47 | 0.184 |
| Catholic | A great deal | 0.22 | 0.445 | 0.335 |
| Jewish | Hardly any | 0.5 | 0.5 | 0 |
| Jewish | Only some | 0.161 | 0.581 | 0.258 |
| Jewish | A great deal | 0.167 | 0.633 | 0.2 |
| Eastern | Hardly any | 0.5 | 0 | 0.5 |
| Eastern | Only some | 0.2 | 0.4 | 0.4 |
| Eastern | A great deal | 0.2 | 0.333 | 0.467 |
| Other | Hardly any | 0.4 | 0.6 | 0 |
| Other | Only some | 0.472 | 0.321 | 0.208 |
| Other | A great deal | 0.184 | 0.469 | 0.347 |
| None | Hardly any | 0.455 | 0.242 | 0.303 |
| None | Only some | 0.352 | 0.496 | 0.152 |
| None | A great deal | 0.23 | 0.378 | 0.393 |

## Dealing With NAs In Other Functions

For mean and standard deviation, remove NAs by adding `na.rm = TRUE`:

```
mean(review$age)
```

```
## [1] NA
```

```
mean(review$age, na.rm = TRUE)
```

```
## [1] 48.72003
```

```
sd(review$educ)
```

```
## [1] NA
```

```
sd(review$educ, na.rm = TRUE)
```

```
## [1] 2.996095
```

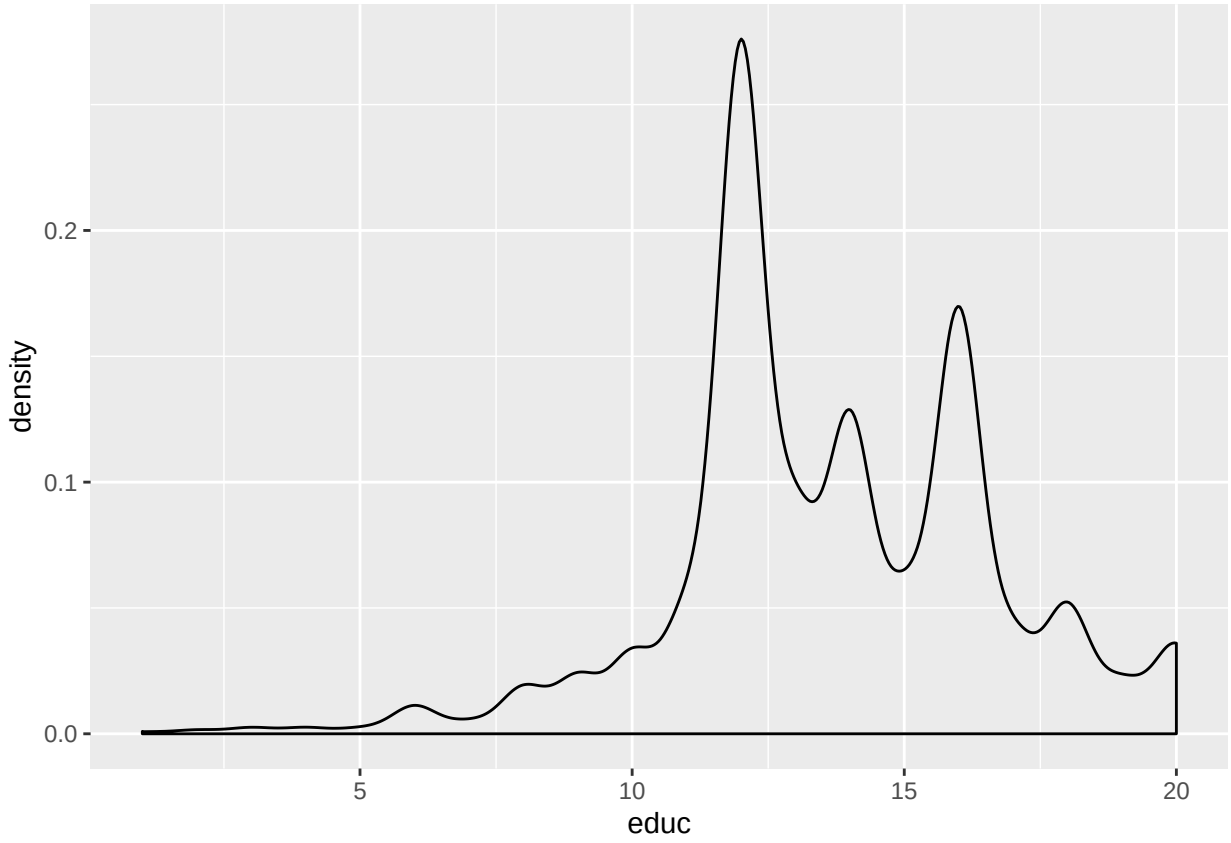
For correlation, restrict the estimation to cases with values for both variables by adding `use = "complete"`:

```
cor(review$age, review$educ, use = "complete")
```

```
## [1] -0.0304108
```

For ggplot, R knows to only use complete cases but will warn you that it is doing so. To drop the warning, add `warning = FALSE` to the start of the code chunk:

```
plot <- ggplot(review, aes(x = educ))
plot + geom_density()
```




Remember to change the axis labels and add a title to the figure above!

## REPLACE THIS LINE WITH YOUR CODE

Basic linear models also know to drop NAs. The notes section of the summary informs you how many cases have been deleted from the estimates (in the example below, 4210 observations are deleted due to missingness).

***This is new: notice how we are redefining the science factor variable to have a numeric scale in the chunk below. Each of the three factor levels will be assigned a number from 1-3. Since we asserted that the order of levels is "Hardly any" / "Only some" / "A great deal", now higher scores tell us that respondents have more confidence in scientific institutions. (This is a neat***

***trick, but in general be careful with this approach. It only works if you can assume that the distance between each level is even.)***

```
model <- lm(as.numeric(science) ~ educ, data = review)
summary(model)
```

```
##
## Call:
## lm(formula = as.numeric(science) ~ educ, data = review)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -1.6058 -0.3684 -0.2497  0.5525  1.1460
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 1.814402   0.032259   56.24   <2e-16 ***
## educ        0.039569   0.002295   17.24   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.5935 on 7559 degrees of freedom
##   (4210 observations deleted due to missingness)
## Multiple R-squared:  0.03782,    Adjusted R-squared:  0.0377
## F-statistic: 297.1 on 1 and 7559 DF,  p-value: < 2.2e-16
```

Use stargazer with this model for your final knitted version:

Table 2: Model Predicting Confidence in Science by Years of Education

| | Confidence in Science |
|---|---|
| Education (in years) | .040*** |
| | (.002) |
| Constant | 1.814*** |
| | (.032) |
| Observations | 7,561 |
| $R^2$ | .038 |
| *Notes:* | *P < .05 |
| | **P < .01 |
| | ***P < .001 |

By default, the `fitted()` function will not work if there are NAs in your model. If you want

to save predicted values from a model with missing values, add `na.action = na.exclude` to your `lm()` code. Now when you run the `fitted()` function any observations not inlcuded in your model will have NA as their predicted value.

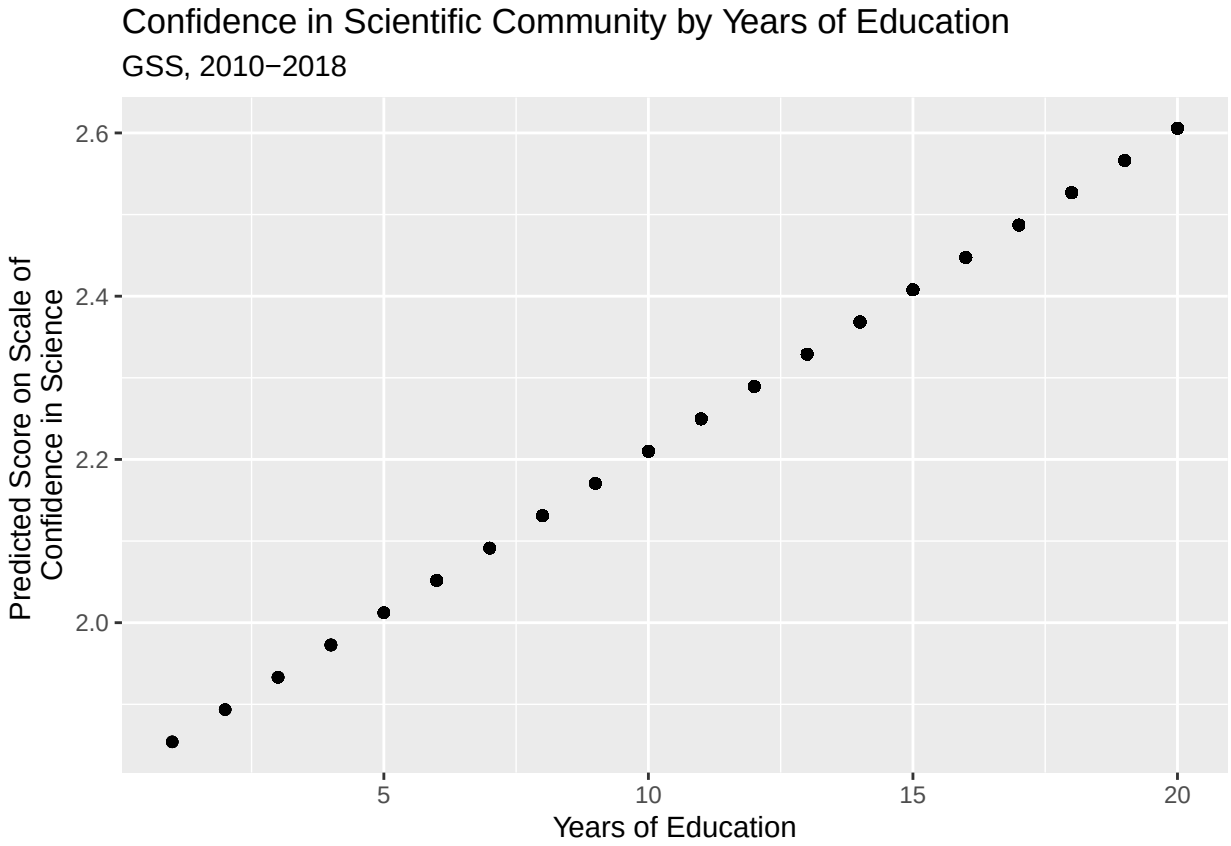
```
model <- lm(as.numeric(science) ~ educ,
            data = review,
            na.action = na.exclude)

review$predicted_science <- fitted(model)

summary(review$predicted_science)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.    NA's
##   1.854   2.289   2.368   2.358   2.448   2.606    4210
```

## More Thoughts On Plotting

Always remember to label your axes and plots!

```
plot <- ggplot(review, aes(x = educ, y = predicted_science))
plot + geom_point() +
     labs(x = "Years of Education",
          y = "Predicted Score on Scale of \nConfidence in Science",
          title = "Confidence in Scientific Community by Years of Education",
          subtitle = "GSS, 2010-2018")
```




# Using Markdown For Reports

## Hiding Code and Inline Code

Let's start with a case where your output is a single number, like a mean. Imagine you are working on the descriptives part of your project and want to include the mean of age. The place to start is with a regular code chunk with the `mean()` function:

```
mean(review$age, na.rm=TRUE)
```

```
## [1] 48.72003
```

But say you want R to run a code chunk and have only the output - not the code! - show up in your file. Simply add `echo = FALSE` to the first fence:

```
## [1] 48.72003
```

If you want to integrate a single number into your document, you can use inline code. Without opening a full code chunk, just use one backtick to open and close your fence. Then write a sentence as you normally would, and let R Markdown replace your code with the output:

The mean of age is 48.72.

## Other Options For Hiding Code

If you want to run the code chunk so you can see the output in your notebook but with neither the code nor the output showing up in your knitted file, use `include = FALSE`.

I would probably recommend starting with `include = FALSE` for your final project, so you can see all your output but then selectively choose what to include and what not to include in your knitted report.

If for some reason you want to show the code but not the output, use `eval = FALSE`.

```
mean(review$age, na.rm=TRUE)
```

## R Markdown Tips

Some other things to know about writing in R Markdown...

Use hashtags for headings. One hashtag is for a big heading; additional hashtags shrink the size. For example:

# Biggest Heading

## Big Heading

### Small Heading

#### Smallest Heading

If you want to italicize text, *wrap it within single asterisks*. If you want to bold text, **wrap it within double asterisks**. And if you want to italicize *and* bold text, ***wrap it within triple asterisks***.

It can sometimes be helpful to highlight original variable names or unusual terms within tickmarks. But note this is similar to the inline code we saw earlier. As long as the word or phrase does not start with a single r, R will not try to run it as code. See the preview file for the difference in what these tickmarks represent:

The mean of `age` is 48.72.

To create an ordered list, leave an empty line and then:

- Start
- Each
- Item
- With
- A

- Dash

To create a numbered list, leave an empty line and then:

1. Start
2. Each
3. Item
4. With
5. A
6. Number and a period

To add a horizontal line rule, include at least three dashes on a single line:

---

And to add a page break:

This should be the start of a new page!

# It's Also The Start Of A New Section

## Formatting Summary Tables

We have seen `pander()` a lot. It's great. Use it.

One additional way to use `pander()` is to combine it with `group_by()` and `summarize()` to make a nice summary table. Let's start with the code for getting means and standard deviations of the `age` and `educ` variables for each `religion` group:

If we pander this table, we'll have the religion categories in the rows and the means and standard deviations in the columns:

| religion | mean_age | sd_age | mean_educ | sd_educ |
|---|---|---|---|---|
| Protestant | 51.05 | 17.87 | 13.68 | 2.78 |
| Catholic | 47.88 | 17.05 | 13.5 | 3.27 |
| Jewish | 56.24 | 18.01 | 15.96 | 2.67 |
| Eastern | 41.58 | 16.4 | 15.24 | 3.2 |
| Other | 38.38 | 15.45 | 13.42 | 2.68 |
| None | 41.61 | 15.96 | 13.62 | 3.04 |

Note that `pander()` also works well with `t.test()`...

Table 4: Welch Two Sample t-test: `review$age[review$religion == "Jewish"]` and `review$age[review$religion == "Eastern"]`

| Test statistic | df | P value | Alternative hypothesis | mean of x | mean of y |
|---|---|---|---|---|---|
| 8.329 | 379.8 | 1.5e-15 * * * | two.sided | 56.24 | 41.58 |

...and `prop.test()`...

Table 5: 2-sample test for equality of proportions with continuity correction: `space_religion_table`

| Test statistic | df | P value | Alternative hypothesis | prop 1 | prop 2 |
|---|---|---|---|---|---|
| 13.97 | 1 | 0.0001856 * * * | two.sided | 0.7913 | 0.7137 |

...and `chisq.test()`...

Table 6: Pearson's Chi-squared test: `review$sex` and `review$consci`

| Test statistic | df | P value |
|---|---|---|
| 60.36 | 2 | 7.815e-14 * * * |

...and `fisher.test()`...

```
## Warning in chisq.test(educ_11_years_only$religion,
## educ_11_years_only$space): Chi-squared approximation may be incorrect
```

```
##                             educ_11_years_only$space
## educ_11_years_only$religion Not interested Moderately interested
##                  Protestant     45.9183673            54.5918367
##                  Catholic       30.4897959            36.2489796
##                  Jewish          0.3673469             0.4367347
##                  Eastern         0.3673469             0.4367347
##                  Other           3.3061224             3.9306122
##                  None            9.5510204            11.3551020
##                             educ_11_years_only$space
## educ_11_years_only$religion Very interested
##                  Protestant      24.4897959
##                  Catholic        16.2612245
##                  Jewish           0.1959184
##                  Eastern          0.1959184
##                  Other            1.7632653
##                  None             5.0938776
```

Table 7: Fisher's Exact Test for Count Data with simulated p-value (based on 2000 replicates): `educ_11_years_only$religion` and `educ_11_years_only$space`

| P value | Alternative hypothesis |
|---|---|
| 0.04798 * | two.sided |