Creating Interactive Tutorials for Teaching Statistics Online with learnr

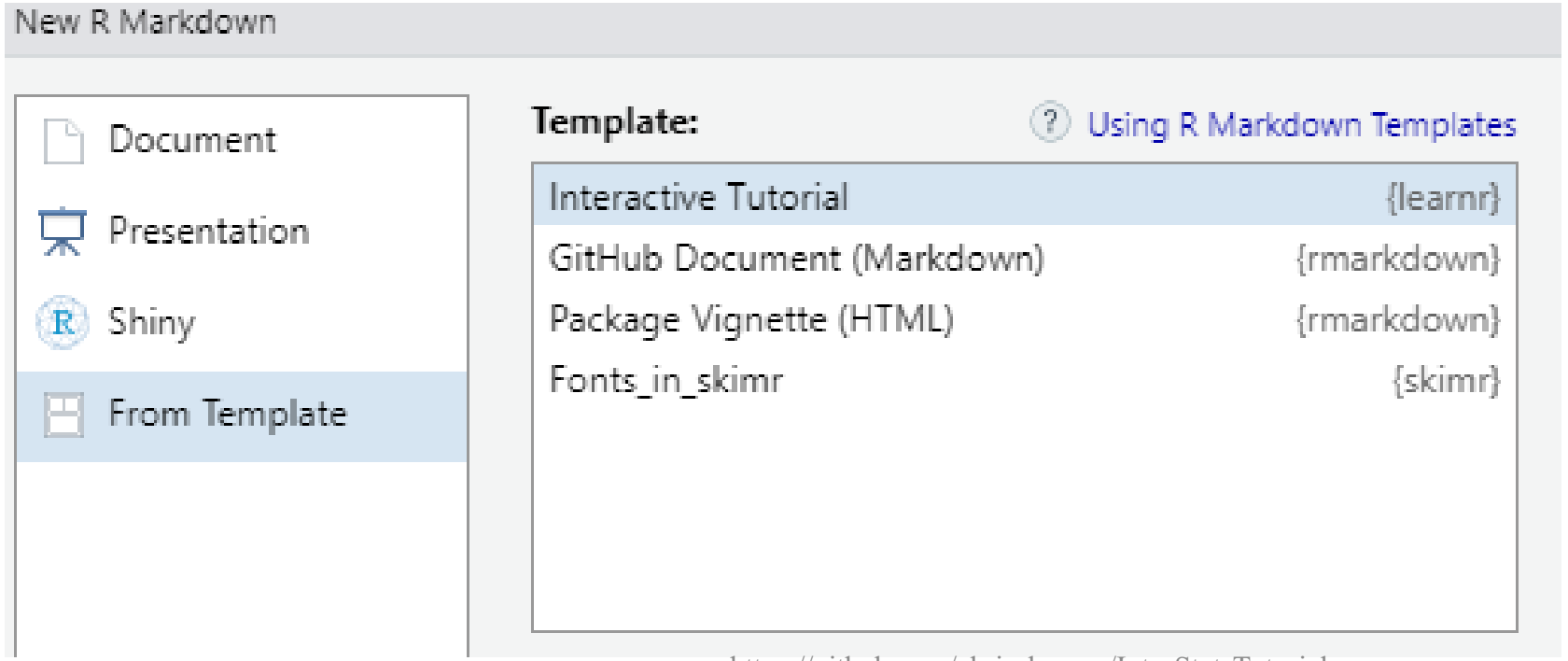
Chris Aberson
Department of Psychology
Cal Poly Humboldt
cla18@humboldt.edu
https://github.com/chrisaberson/IntroStatsTutorials

learnr

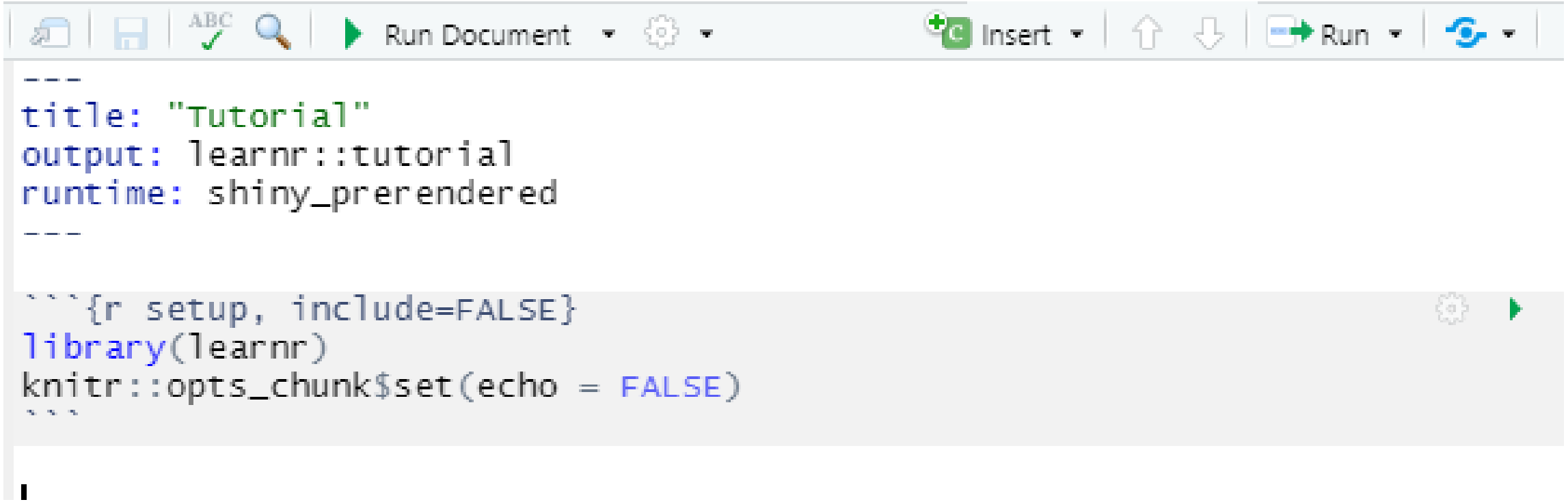
- learnr is developed by members of the RStudio team to provide an easy-to-use (relatively) approach to building interactive tutorials
- Tutorials can include videos, quizzes, images, equations, shiny apps, and a space to run analyses directly inside the tutorial
- learnr tutorials are shiny apps but you don't have to know anything about building shiny apps to develop them
- The link on each of these slides provides tutorials for my full intro stats course (or simply google Aberson GitHub)
- (Tour of a learnr tutorial/Markdown file)

Getting Started

- Choose file – new file – R Markdown – from Template
- (or just use someone else's existing tutorial and modify)



Template



- Top part is the header – don't mess with that, it is fussy
- First code chunk loads learnr, add whatever packages you are using inside this chunk
- Template includes examples of questions, exercises, etc.

Components: Adding Pages

- Organize logically using different pages
- Formatting the look of the tutorial involves three different levels
- These are noted with #
- # Title of Tutorial
- ## New Page
- ### New section within a page
- Be sure to put a space after the #, otherwise it is read as a comment

Examples: Adding Pages

```
# Data Visualization

## Overview

This tutorial focuses on data visualization (i.e., graphing) The tutorial
includes a combination of videos, text, knowledge check quizzes, and exercises.

The PowerPoint slides for the presentation in the videos are on Canvas if you
want a copy. For those not enrolled in my class, these files can be found here:
https://osf.io/9tgxm/

The videos (as well as others) can also be found on my YouTube channel
https://www.youtube.com/channel/UC5kDZTyHZlgSgSEa3YQXOig

## Packages

This tutorial uses the following packages:

* `ggplot2` for fancy graphs
* `learnr` and `shiny` and `rmarkdown` for aspects of the tutorial
```

Components: Videos

- Adding video is simple. All you need is a web link
- This will run in an window inside the tutorial
- I am working on keeping videos about 5-10 minutes long

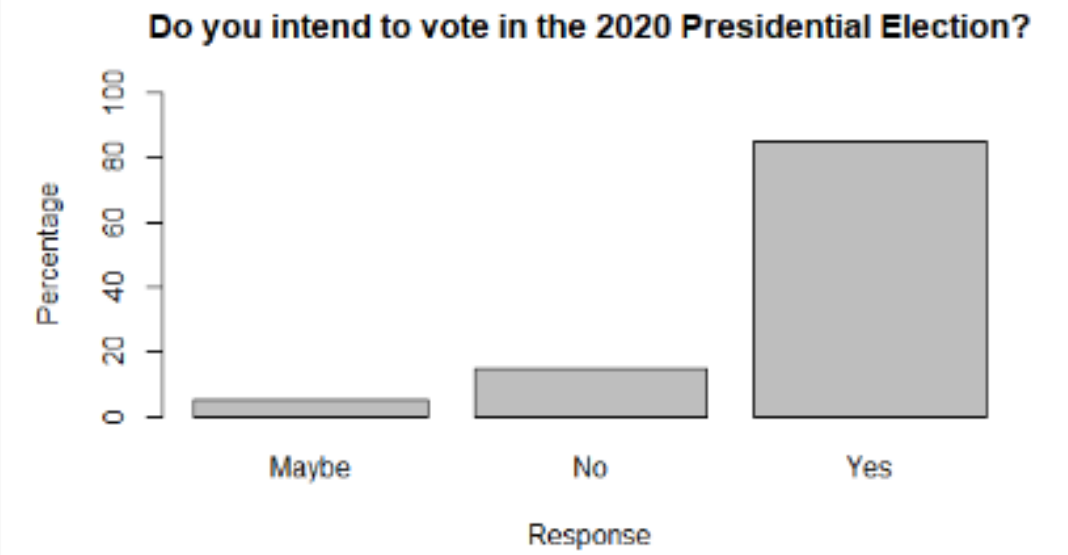
```
## Video 1

![Video 1: Introduction](https://youtu.be/Eov3a3eQrRo)
```

Components: Images

- Just like a video

```
![Do you intend to vote?](images/quiz1.1.png)
```



Components: Quizzes

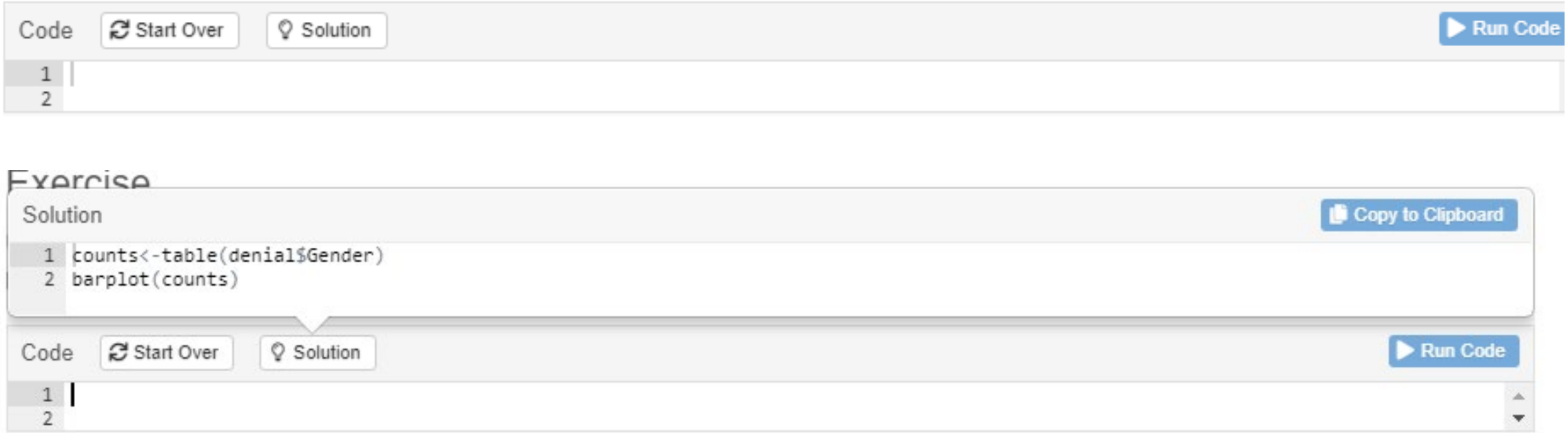
- Quizzes are a bit challenging
- Require a very specific format
- Quizzes do not record data. They are for students to check their knowledge
- Can provide feedback
- Can allow student to try again (I always do this as I want them to eventually land on the right answer)
- This is a nice way to bring in low stakes testing as it allows the student to work toward mastery

Example: Quizzes

```
```{r quiz1}
learnr::quiz(
  learnr::question("The graph titled 'do you intend to vote' shows which of the
following?",
    learnr::answer("Yes is the most common answer", correct = TRUE),
    learnr::answer("No is the most common answer"),
    learnr::answer("Maybe is the most common answer"),
    learnr::answer("None of the above"),
    correct = "Correct! Yes is the most common response with 85%",
    incorrect = "Sorry, that's incorrect. Try again.",
    random_answer_order = TRUE,
    allow_retry = T
  ),
  learnr::question("The graph titled 'On a scale of 1 to 100 ...' shows ... ",
    learnr::answer("Most answers were between 70 and 90", correct = TRUE),
    learnr::answer("Most answers were between 50 and 100"),
    learnr::answer("Most people are not certain they will vote"),
    learnr::answer("None of the above"),
    correct = "Correct. The graph show that most (i.e., more than half at least)
of the scores fall between 70 and 90",
    random_answer_order = TRUE,
    incorrect = "Sorry, that's incorrect. Try again.",
    allow_retry = T
  )
)

```
```

Adding text

- You literally just type text
- *italics*
- **bold**
- Superscript^2^
- Subscript~2~
- Help menu in RStudio has a useful markdown cheat sheet and quick reference

Running Code

- The true power, in my opinion, of learnr comes from the ability to execute code
- This is a relatively simple thing to do
- Major applications
  - Go line by line through complex code with a description after each (you don't need learnr for this, but it is a strategy I like)
  - Provide code that generates output for interpretation
  - Ask questions and have students write and execute code to address question (with an option to add a solution – the correct code – if they get stuck)

Creating exercises

- The code below creates an exercise that asks the student to perform a specific analysis
- The first chunk names the exercise (ex2) and indicates how big to make the code box.
- The 2nd optional (but highly recommended) chunk provides the correct answer

```
```{r ex2, exercise = TRUE, exercise.lines = 2}

```
```

```
```{r ex2-solution}
counts<-table(denial$Gender)
barplot(counts)
```
```

Exercise Rendered

Using the denial data create a bar graph for the variable **Gender**. Try to write the code yourself. If you get stuck, you can use the "solution" button.



Tips regarding exercises

- Test. Test. Test. Definitely make sure it is working.
- Clear your workspace when testing – something may only work because you have certain packages loaded or you opened the data file earlier.
- You likely have materials that would lend themselves to these exercises already!
- I like to follow each exercise with a quiz (generally interpreting the output)

Code Examples

- github/chrisaberson
- Tutorials available for Intro Stats, Regression/Multivariate (Grad; only two), and ANOVA (Grad)
- Repositories are IntroStatsTutorials, ANOVAtutorials, and Mvstats
- Please feel from to borrow!