DataCamp: An Interactive Teaching Tool

Tessa Mcdonnel, Shannon Pileggi
Statistics Department
California Polytechnic State University, San Luis Obispo

January 3, 2018

1 Introduction

It is no surprise that technology plays a vital role in the field of statistics. As the technological revolution continues to unfold, the use of statistical software has become increasingly emphasized in education [American Statistical Association, 2016]. By integrating the capabilities of technology in statistics courses, students can explore fundamental concepts by analyzing and interpreting real data sets [Chance et al., 2007, Hardin et al., 2015, Horton et al., 2014], giving them an authentic view of how statistics is practiced outside of the classroom [Wang, 2017].

The goal for any statistics course is to help students develop the ability to think statistically [American Statistical Association, 2016] and introducing technology into the curriculum can greatly strengthen this ability by illustrating concepts such as variability, randomness and hypothesis testing.

Although numerous statistical technologies exist, R and R Studio provide extensive computational abilities (free of charge) and are commonly used in statistics courses. R allows students to explore real-world data sets which can not only enhance their understanding of concepts, but can also spark an interest in the domain of statistics [Wang, 2017].

1.1 Interactive tools to learn R

With this increased use of R in classrooms, an effective tool to familiarize students with statistical software is needed [Baumer et al., 2014]. While R software provides users with extraordinary statistical capabilities, there is certainly a steep learning curve. To facilitate this transition to R programming, popular interactive teaching technologies have been created including *swirl* [Kross et al., 2017] and Try R [Try, 2014]. Interactive tools allow students to work through coding problems while receiving immediate feedback and can be used in and outside of the classroom.

1.1.1 *swirl*

swirl [Kross et al., 2017] is an open source R package that allows students to learn R programming in the R console. This interactive learning tool contains several pre-existing lessons but educators can also create their own lesson content to fit specific curriculum [Carchedi, 2014]. While *swirl* can be linked to

Massive Open Online Courses (MOOCs) like Coursera, integration of learning management systems (LMS) is not available for the classroom.

1.1.2 Try R

Try R [Try, 2014] is another interactive R programming course but unlike *swirl*, Try R is web-based. The R console can be intimidating to new statistics students so an advantage of web-based teaching tools is the clean and easy to use interface. User-friendliness is an important quality given that students often struggle more with the software used than the statistical concepts themselves [Hare and Kaplan, 2017]. When students only need their web browser to participate, all installation complications are avoided and valuable class time can be focused on the lesson. However, Try R is not open source so public content authoring is not available; educators are limited to the existing lessons available on the Try R website.

2 DataCamp

2.1 DataCamp course features

The purpose of this article is to introduce a new interactive teaching technology, DataCamp.

2.1.1 DataCamp compared to similar technologies

DataCamp provides the same reproducible capabilities as *swirl* but is a web-based application. Educators can create their own courses that coincide with specific lessons and students can access the course without installing any software. Courses are available through the DataCamp website so students can access the lesson through their web-browser on a laptop, tablet or mobile device, making the courses accessible to anyone with an internet connection. These programming courses can be used in the classroom in a number of ways: as traditional homework assignments, as an activity during lab periods, as pre-class assignments for a flipped classroom, and even as a test or assessment tool.

2.1.2 Wide variety of course topics

The DataCamp website offers pre-existing free community courses as well as more specialized *Premium* classes that require a fee. The collection of *community* courses available on DataCamp include a wide range of topics from *Introduction to R* to *Quantitative Risk Management in R* so novice through advanced programming students benefit.

2.1.3 Integration with Learning Management Systems

One feature that sets DataCamp apart from competing technologies is the ability to track student progress. Educators can set deadlines for assignments and view which sections the students had trouble with in a detailed report. DataCamp courses can also be linked to the school's learning management system (LMS) so student scores are automatically updated to the current grade-book, allowing for a smooth integration of student work.

2.1.4 Modularity

DataCamp courses are broken up into chapters and each chapter is composed of modular exercises. Modularity is a critical component in designing a learning tool because it allows independent pieces to be easily added, modified or removed from the entire functionality [Hare and Kaplan, 2017]. DataCamp course exercises can be added, modified, or deleted as new topics arise and instructors can tailor these exercises to meet the needs of a particular class. The only disadvantage of the modularity feature is that the sessions are not cumulative; every exercise begins with a clear work space so an object defined in a previous exercise needs to be defined again.

2.1.5 Gamification

Each chapter exercise is given a specified number of redeemable points which the students can see when they begin the exercise, introducing a gamification feature to the learning experience. When in a classroom group, students can view their progress (points and chapters completed) as well as then progress of their peers. This can help students perform better because they want to be on par with the rest of the class; furthermore, when students can see the number of available points for an exercise it can motivate them to meet their personal achievement goals. [Chang and Wei, 2016]. If the student completes an exercise without any assistance, they will receive the full points. If the student is having trouble with a problem, they can view the exercise 'hint' option and 30 percent of the available points are deducted; all points are deducted if they ask for the solution.

2.2 Pre-existing R courses

In order to utilize DataCamp's *classroom* benefits, first create an academic group on the website `https://www.datacamp.com/groups/education`. After your university information has been verified and the classroom account has been approved, access is facilitated by either creating a group through the DataCamp website or linking directly to your LMS. You can add students to the course by their university email address on the *Members* tab.

3 Creating a DataCamp Course

3.1 Getting Started

To build a course on DataCamp you will first need to create a DataCamp account by visiting `www.datacamp.com/teach` as well as a Git Hub account `https://github.com`. Every DataCamp course is linked through Git Hub version control, allowing statistics educators to easily collaborate and reproduce code. To add collaborators to the course, go to the course repository on Git Hub and under `Settings` you can enter the user names of collaborators. Collaborators who are not the primary author need to uncheck 'My Courses' in order to view and contribute to the repository. The simplest way to begin building the course is to use the template course files from the `Create DataCamp Course` dialog. This will automatically create a Git Hub repository that is linked to DataCamp; any changes that are made through DataCamp will update the Git Hub repository and vice versa.

The newly created Git Hub repositories will contain three files:

1) `README.md`

2) `course.yml`

3) `chapter1.Rmd`

`Readme.md` contains helpful information of how to get started, citing resources to learn more about the process. `course.yml` is a file that contains general

information about the course you're creating: title, university, description, difficulty level, ect. `chapter1.Rmd` is a template chapter file with examples of 'normal' iterative coding exercises and multiple choice questions. To add more chapters to an R course, create new files in the repository and name them `chapter1.Rmd`, `chapter2.Rmd`, `chapter3.Rmd`, ect.

3.2 Course Editing

Educators are able to edit their courses in a number of ways but the `Teach Editor` located in the teach dashboard is the most straight forward editor. The `Teach Editor` is specifically designed to make the creation process of a DataCamp course as painless as possible, allowing writers to edit, preview, save changes and synchronize to a Git Hub repository. Editing can also be done through Git Hub's online editor or through git locally but these methods do not allow for automatic content viewing.

3.3 Components of a DataCamp Course

3.3.1 Exercise Layout

Figure1 is an example of an interactive coding exercise for an introductory statistics course. Students will read the lesson information and instructions, answer the question, and submit their code; it should take approximately 5 minutes to complete this exercise.

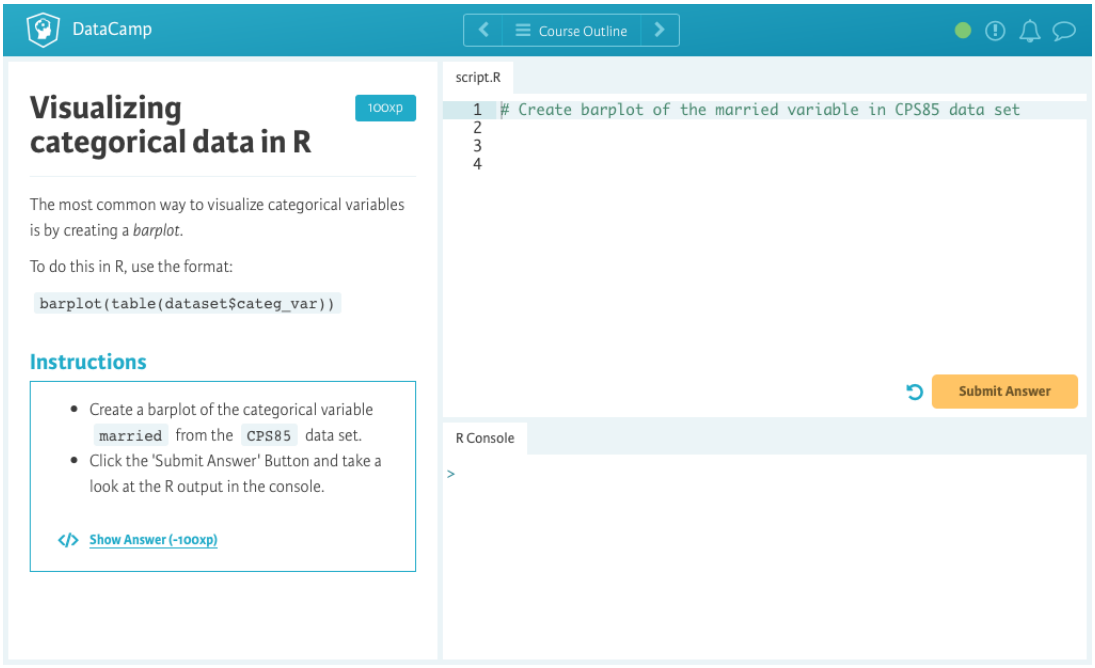
Figure 1: Teach Editor Interface

````
--- type:NormalExercise lang:r xp:100 skills:1 key:9a4fea2da
## Visualizing quantitative data in R

We can visualize quantitative data with a *histogram*.

To create a histogram in R, use the syntax:
`hist(dataset$quant_var)`

*** =instructions
- Create a histogram of the `wage` variable from the
`CPS85` data set.
*** =hint
Follow the format from the lesson: `hist(dataset$variable)`
with specified data set and variable.
*** =pre_exercise_code
```{r}
library(mosaicData)
```
*** =sample_code
```{r}
# Create histogram of the wage variable
```
*** =solution
```{r}
# Create histogram of the wage variable
hist(CPS85$wage)
```
*** =sct
```{r}
test_function("hist", args = "x", incorrect_msg =
"Make sure you specify the `CPS85` data set
and `wage` variable.")
```
````



Whenever a new exercise is added, DataCamp automatically produces the top line (header) in the editor. The header specifies the type of exercise `type`, language `lang`, available points `xp`, skills learned `skills` and a `key`. The key acts as a unique identifier for that exercise, allowing you to modify the content without losing any student data. Currently, DataCamp offers eight different skills which are defined by numbers 1-8. The skills and their corresponding numbers can be found in the DataCamp documentation `https://www.datacamp.com/teach/documentation#tab_gamification`. In the above exercise code (refer to teach editor code), the `type` is Normal, the language is R, the available points is 100 and the skills learned is 1 (which corresponds to R Programming). DataCamp assigns 100 points to normal coding exercises but this value can be modified depending on difficulty level. After students

have completed the exercise, they can check their personal profile to see how many points they earned.

In the *Assignment* portion of the exercise (not labeled but below the header), instructors can add additional information about the assignment that will help students understand and complete the instructions. The *instructions* section contains the actual task the student is asked to execute; instructors can also include a *hint* for extra guidance (optional). The *pre-exercise code* sets up the work space for the student; this section will contain code to import data sets, install packages and any other operation that you want to be available to students. The *sample code* section contains code or notes that students view in the R script panel. Students also use the R script panel to write the code that solves the tasks in the instructions. The correct answer to an exercise is coded in the *solution* section and when the student clicks the `Submit Code` button, their answer passes through the SCT code where their code is checked for errors.

3.3.2 Submission Correctness Test

The submission correctness testing code (or SCT) automatically assesses if the correct code was submitted. If a student submits an incorrect answer, they will receive immediate feedback on what they are doing wrong and the system will guide them to the right answer. Instructors can use the default feedback that DataCamp automatically generates or write their own custom

Figure 2: Examples of DataCamps Default Feedback

```
Student types:
histogram(dataset$var_name)

Default message:
Have you called hist()?

Student types:
hist(var_name)

Default message:
Check your call of hist(). Did you correctly specify
the argument x? Evaluating the expression you specified
caused an error.

Student types:
hist(dataset$misspelled_var)

Default message:
Check your call of hist(). Did you correctly specify the
argument x?
It is a NULL, while it should be a numeric vector.
```

messages. For example, figure blank contains the default feedback for a few possible incorrect submissions of the `hist()` function.

The default feedback is meaningful and specific; most of the time custom messages are not necessary unless there is a common problem that student are struggling with in which more unique feedback is necessary. DataCamp provides a helpful R package called *testwhat* which acts as a guide to writing and using SCTs (this can be accessed on DataCamp's Git Hub repository:

`https://github.com/datacamp/testwhat/wiki`. *testwhat* contains functions that can test different types of problems including loops, object definitions, printed output, functions, ect. The submission correctness testing code compares the ideal answer (from the solution) to the students answer. Different arguments can be added to the test functions to alter the testing process and automatic feedback. (**do I cite datacamp website here?**)

3.4 DataCamp Exercise Types

DataCamp courses may contain 3 different types of exercises: Normal coding, video, and multiple choice exercises.

3.4.1 Normal Exercise

A `NormalExercise` is an exercise that prompts a user to submit code. The components of this include: lesson section, instructions, pre-exercise code, sample code, hint, solution and a submission correctness testing code (SCT). (add reference to teach editor code) However, the interface for students (or the `Preview` mode) only includes the lesson, instructions, R script panel and a R console panel (Figure 2).

3.4.2 Video Exercise

Video exercises act as in instructional video addition to a DataCamp course; instructors can add short videos of them explaining concepts and with relevant slides that appear in the background. The DataCamp website `https://www.datacamp.com/teach/documentation#tab_code_exercises` provides more detail on the content of the various exercises.

3.4.3 Multiple Choice Exercise

The `MultipleChoiceExercise`, as you would expect, contains a question and a list of various answers. Instructors write the questions in the 'lesson' portion of the exercise and offer solutions in the 'Instructions' section. Similar to the 'normal' coding exercises, the lesson, instructions, pre-exercise code and SCTs have to be specified but sample and solution code do not. DataCamp offers two types of multiple choice questions: simple and regular. The simple multiple choice exercises are used for questions that do not require the R console or any R output. Regular multiple choice exercises allow instructors to pre-program objects or plots that the student needs to access to answer the question. Both types of multiple choice exercises are straight forward to write can act as quick checks for understanding throughout the chapter.

Figure 3: This code does this

````
--- type:PlainMultipleChoiceExercise lang:r xp:50 skills:1
    key:9707072219

## Quick check 2

What can R be used for?

*** =instructions
- Graphing data
- Analyzing data
- Calculations
- All of the above

*** =hint
R software can be used for all of these things.

*** =pre_exercise_code
```{r}

```

*** =sct
```{r}

msg_bad <- "That is not correct"
msg_success <- "Exactly! R can do all of these things."
test_mc(correct = 4,
    feedback_msgs = c(msg_bad, msg_bad, msg_bad, msg_success))
```
````

3.5 Advise for smoother course development

DataCamp is fairly new so the only existing information on creating a course is on the DataCamp website. While writing my first introductory R programming class on DataCamp, I learned a few tricks that I wish I knew before I started.

3.5.1 Git Hub repositories

When creating your first DataCamp course, it is helpful to view code from pre-existing community courses which can be found on Git Hub. DataCamp's github repository contains links to the code for several available courses; one of the most helpful when starting to build a course is `https://github.com/datacamp/courses-intro-to-r` which contains the source files for the *Introduction to R* course. You can use these files as a guide to build your own course or duplicate the repository and tailor the content to fit specific needs.

3.5.2 Integrating R packages

All DataCamp courses will have a course badge and most will use data sets and R packages. Information on how to add data sets and images can be found in DataCamps 'Teach' documentation `https://www.datacamp.com/teach/documentation` under the 'Upload Assets' tab. However, there is

currently no information on DataCamp about integrating R packages. To use R packages in a course, you will need to create a new file in the Git Hub course repository named `requirements.r`. The file should contain: `devtools::install_version`, package name and version for every package you use in the DataCamp course. For example, following code will add the `ggplot2` and `mosaic` packages to the repository.

```
devtools::install_version("ggplot2", "2.2.0")
devtools::install_version("mosaic", "0.14.4")
```

In order for DataCamp to recognize this new file, you will need to add `from: r-base-prod:27` to the `course.yml` file.

3.5.3 TeX typesetting

LaTeX is comparable with DataCamp course builders can use LaTeX math symbols to display Greek letters, formulas, ect. For example:

```
$H_0$: $p = 0.07$
$H_a$: $p \neq 0.07$
```

translates to:

H_0: $p = 0.07$

H_a: $p \neq 0.07$

4 Conclusion

References

Try R, 2014. URL http://tryr.codeschool.com/.

American Statistical Association. *Guidelines for Assessment and Instruction in Statistics Education (GAISE) College Report*. 2016. ISBN 9780979174711. doi: 10.3928/01484834-20140325-01.

Ben Baumer, Mine Cetinkaya-Rundel, Andrew Bray, Linda Loi, and Nicholas J. Horton. R Markdown: Integrating A Reproducible Analysis Tool into Introductory Statistics. *Technol. Innov. Stat. Educ.*, 8 (1):20, 2014. ISSN 1933-4214. doi: 10.5811/westjem.2011.5.6700. URL http://arxiv.org/abs/1402.1894.

Nicholas A Carchedi. TURNING THE R CONSOLE INTO AN INTERACTIVE. 2014.

Beth Chance, Dani Ben-Zvi, Joan Garfield, and Elsa Medina. The Role of Technology in Improving Student Learning of Statistics. *Technol. Innov. Stat. Educ.*, 1(1):1–26, 2007. ISSN 1933-4214. URL file:///C:/Users/jacquel7/Downloads/eScholarshipUCitem8sd2t4rr.pdf.

Jen Wei Chang and Hung Yu Wei. Exploring engaging gamification mechan-

ics in massive online open courses. *Educ. Technol. Soc.*, 19(2):177–203, 2016. ISSN 14364522.

J Hardin, R Hoerl, Nicholas J Horton, Deborah Nolan, Ben Baumer, O Hall-Holt, P Murrell, R Peng, P Roback, Duncan Temple Lang, and M D Ward. Data Science in Statistics Curricula: Preparing Students to "Think with Data". *Am. Stat.*, 69(4):343–353, 2015. ISSN 0003-1305. doi: 10.1080/00031305.2015.1077729.

Eric Hare and Andee Kaplan. Designing Modular Software: A Case Study in Introductory Statistics. *J. Comput. Graph. Stat.*, pages 0–0, 2017. ISSN 1061-8600. doi: 10.1080/10618600.2016.1276839. URL `https://www.tandfonline.com/doi/full/10.1080/10618600.2016.1276839`.

Nicholas J. Horton, Benjamin S. Baumer, and Hadley Wickham. Teaching precursors to data science in introductory and second courses in statistics. *arXiv1401.3269 [cs, stat]*, 2014. URL `http://arxiv.org/abs/1401.3269{\%}5Cnhttp://www.arxiv.org/pdf/1401.3269.pdf`.

Sean Kross, Nick Carchedi, Bill Bauer, Gina Grdina, Filip Schouwenaars, and Wush Wu. swirl: Learn R, in R. 2017. URL `https://cran.r-project.org/package=swirl`.

Xiaofei Wang. Data Visualization on Day One: Bringing Big Ideas into Intro Stats Early and Often. *Technol. Innov. Stat. Educ.*, 10(1):1–19, 2017. ISSN 1936-900X. doi: 10.5811/westjem.2011.5.6700.