Introductory Data Science: A Blueprint to Navigate Curricular, Pedagogical, and Computational Challenges

Elijah Meyer
Department of Statistics, Duke University
and
Mine Çetinkaya-Rundel
Department of Statistics, Duke University

April 4, 2023

Abstract

The text of your abstract. 200 or fewer words.

Keywords: Data Science, Curriculum, Pedagogy

1 The Course

Reader should want to continue reading by thinking "this is something that I would like to teach"

In the following sections, we describe our introductory data science course, offered to predominantly freshman level students at Duke University, titled Introduction to Data Science and Statistical Thinking (STA199). Often, these students are (interested in / major + minor in / description of typical student demographic). Students enrolling in STA199 commonly have little to no statistics, data science, or coding experience. In a typical semester, this course seats roughly 179 students, which is considered to be large by all measures. Both the lack of experience and large class size are identified as two common hurdles by faculty when trying to create and instruct an introductory data science course (Schwab-McCoy et al. 2021, Kokkelenberg et al. 2008). However, by the end of this course, students are able to use data both R and GitHub to create data visualizations, investigate patterns, and model outcomes in a reproducible format.

This course is built on four large-scale learning objectives: learn to explore, visualize and analyze data in a reproducible and shareable manner; gain experience in data wrangling and munging, exploratory data analysis, predictive modeling, and data visualization; work on problems and case studies inspired by and based on real-world questions and data; learn to effectively communicate results through written assignments and project presentation. These objectives are accomplished through interactive lectures and labs that present content, problems, and case studies inspired by and based on real-world questions and data.

When teaching, instructors are committed to the Kaplan Way learning model (Figure 1) "that combines a scientific, evidence-based design philosophy with a straightforward

educational approach to learning" (Schweser 2023, pg. 3).

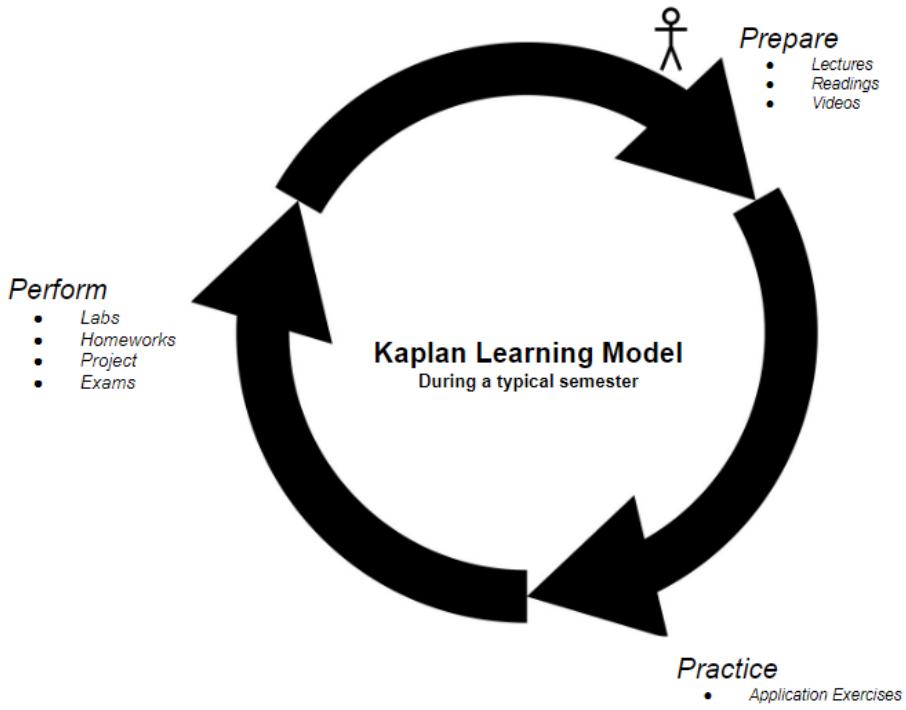

Fig 1: Kaplan Learning Model for STA 199

This model is composed of three phases: prepare, practice, and perform. All three phases are designed to guide the instructor in facilitating an overall quality learning experience for the student. Each of these phases are performed during a typical week within a semester, accomplished through preparation materials, lectures, in-class application exercises, labs, and assessments.

During the prepare phase, students are completing readings, watching videos, and listening to in-class lecture that ultimately helps students' develop a new foundation of data science concepts being learned. The goal of preparing students is to put them in a position where they can build upon what they are learning, and create new knowledge through the practice phase. The practice phase is designed to be an opportunity for students to reinforce the new information gained, as well as uncover new concepts in data science. This is

achieved through the use of interactive application exercises (AEs) administered in class where students either work alone, in groups, or with the instructor in live coding sessions. Finally, students enter the perform phase, showcasing their progress made in the previous two phases. This is typically done through assessments such as homework, labs, exams, and projects. This learning model is a continuous cycle throughout the semester as new topics are introduced.

Topics taught in STA199 fall under two major units: Unit 1 - Exploring data; Unit 2 - Making rigorous conclusions. In Unit 1, students become first introduced to R, R-studio, and Github. During this unit, students start to create data visualizations and learn how to both import and manipulate data to be better suited for analysis. In Unit 2, students extend their investigations with data to include modeling. Specifically, students fit a variety of models (simple linear regression, multiple linear regression, logistic regression), and learn the fundamentals of hypothesis testing. For a more complete description of the topics taught and data sets used in creating these lessons, please see **A fresh look at introductory data science (cite)**.

In this paper, we describe the preparation and implementation process of STA199 in its entirety. This includes details of the teaching team used to instruct, technology chosen to use when creating and implementing STA199, and the pedagogical choices that go into a typical week of teaching. We detail a comprehensive description of a flexible framework on how to create, set up, and implement an introduction to data science course similar to STA199. When describing this framework, we articulate first hand experiences and suggestions surrounding some of the choices made to create and instruct STA199. Throughout this paper, we aim to support newer instructors who are developing or teaching an introductory data science course; instructors with courses that are increasing in size; and instructors

who want to implement more technical tools into their curricula and classroom.

2 Teaching Team

To account for the large class size, we structure and organize a teaching team to best facilitate STA199. Our teaching team as a group consists of one instructor and multiple teaching assistants (TAs). The responsibilities of any TA is to both support the instructor in charge of the class, and support the students in the classroom. These TAs range from undergraduate to PH.D. level students, and vary in teaching experience. **(Writing on TA selection process). Once selected... (writing on TA training).**

Once training is complete, TAs are assigned roles that indicate their responsibilities during the semester. These roles include *course organizer*, *head TA*, *lab leader*, and *lab helper*. Often, these roles are given based on the academic level of student, with more academically experienced TAs taking on the roles of course organizer, head TA, and lab leader, where as students with less experience (i.e. undergraduate students) take on the role of lab helper.

Lab sections are held once a week, and are facilitated in person by both a lab leader and lab helper. The responsibilities of a lab helper are supporting both the students and lab leader as they see fit. Examples of this may include setting up the classroom before class, or conducting small group conversations when students have questions about the material. The lab leader is responsible for facilitating the lab. This involves working through a pre-made lab to ensure they can help students apply concepts discussed in lecture to what's being assigned. In addition, both must hold two hours of office hours each week and have grading responsibilities assigned throughout the semester.

Head TA responsibilities can generally be categorized into the following categories: Ad-

ministrative and Pedagogical. Administrative responsibilities include the organization and distribution of TA responsibilities throughout the semester. It is imperative that the head TA and instructor clearly communicate expectations with each other to establish exactly how rules and responsibilities that are given to the TAs are assigned. Other administrative duties include reminding other TAs about bi-weekly payroll deadlines and ensure TAs are working their alloted hours per week (and not more). Within these alloted hours, a head TA distributes grading assignments and deadlines to both lab helpers and leaders per week. The head TA makes sure that all TAs complete grading within a week and spot check the grading accuracy and quality of all written feedback given. We have had positive experiences in having a head TA delegate administrative tasks to other members of the teaching team. Pedagogically, head TAs are responsible for creating or reviewing answer keys and grading rubrics for homework and lab assessments as the instructor sees fit. Each head TA is also assigned to instruct one lab section during the semester. Before becoming a Head TA, there is additional training that specializes head TAs in their administrative responsibilities.

The course organizer is expected to work across each section of STA199, instead of working with a single instructor. Their responsibilities include creating rubrics for and working through homework and lab assignments. Additionally the course organizer, along with the instructor, answers real time questions virtually during labs from lab leaders. Questions often range from content related to technical questions about GitHub and R. Finally, the course organizer is responsible for handling all requested assignment extension requests from students. This includes filing away student exemptions, providing extensions for extreme circumstances, and enforcing the late work policy outlined in the syllabus when necessary.

Although this is our proposed team structure, we want to emphasize that there is great

flexibility in coming up with how a teaching team is built and operates for an introductory to data science course. Among any team, we encourage a system designed to alleviate the grading responsibilities of a large class from a single individual, dispersed into many among the team. When grading, it is suggested that each individual is properly trained on how to grade assessments, and expectations on how to provide feedback are clear. This has often been a point of contention from students in the past. For each assignment, we recommend having one person grade one problem across the entirety of the class roster. This helps adhere to more consistent grading, and encourages more grading questions to be asked when they arise earlier in the grading process.

Through our experiences, it has been imperative that everyone within the team is communicating with each other. A team with many different roles poses risk for the instructor to be unaware of how or what decisions are being enacted at the grading and lab levels of the course. Thus, it is recommended that the instructor trains everyone on the teaching team to use a communication system that allows every member to communicate any questions they may have, or decisions they make to the entire team. In the past, we have used the software *Slack*, with appropriately named channels such as *grading-questions* where TAs can post examples and questions about grading and the instructor can clearly state their expectations in a response. Further, it should be noted that the head TA should not be treated as a "bridge of communication" for the instructor to the rest of the teaching team. It is critical that that the instructor is in consistent contact with all members of the teaching team in making sure all lab leaders and helpers understand the course content, upcoming assignments, and know what's expected of them in their assigned role. We recommend holding a weekly meeting with all members of the teaching team to ensure this. When members are unable to come, it is an expectation that they watch a recorded video

of the meeting and reach out if they have any questions about what was discussed.

3 Technology

We can use R, R-studio, and GitHub to create interactive lessons, and assign pre-created assessments to individual or groups of students. In the following sections, we will detail the computing infrastructure needed to do so. This includes details and examples with R, R-studio, and GitHub, from the instructor's perspective, to set up lectures, AEs, labs and homework. First, we justify our decision to use GitHub in STA199 before detailing necessary steps and student information needed so creation can take place.

3.1 Why GitHub

We choose to use GitHub for STA199 because it easily and efficiently allows us to create and administer assessments and interactive lessons to a large class size in individual repositories. Further, within these repositories, instructors can provide template code and other resources necessary to best facilitate an interactive lesson where students can code together with the instructor during lecture. Additionally, teaching through GitHub provides the ability to re-use what you currently create as a template for subsequent semesters. Thus, this system both rewards and encourages time investment into the creation of assessments and lesson plans. Moreover, exposing students to version control software early in their academic career helps them develop good habits as researchers and better understand the importance of reproducibility in science. These are just some of the many documented benefits of using GitHub in an introductory data science course. To being, we need each student to create their own GitHub account.

3.2 Setting up a GitHub account

To participate in interactive lessons and assessments, students must set up a GitHub account. This is done of the first day of class, and often, students are given time during class to sign up. Following tips from "Happy Git with R" (cite), we suggest students do the following when creating their name:

– Incorporate your actual name

– Reuse your username from other contexts if you can

– Pick a username you will be comfortable revealing to your future boss

– Be as unique as possible in as few characters as possible. Shorter is better than longer

– Avoid words with special meaning in programming (i.e., NA)

Once students create their account, we suggest getting this information from them in a survey. This normally can be done through your learning management system. It is critical to reiterate to students that spelling and capitalization matter when they provide their GitHub username. We suggest asking the question as follows:

> What is your GitHub username?
>
> Answer this question by ONLY typing your GitHub name and nothing else.
>
> Make sure you triple-check the spelling so I don't add random strangers to our course organization on GitHub.

Once this information is complete, it was must be exported and stored as a `.csv` file. We recommend having the following structure of data collection.

Table 1: Example Data Collection

last_name	first_name	github_username

If you, as the instructor, do not have a GitHub account, you will need to create one as well. This student information will be used to enroll students in your created GitHub organization for your course.

3.3 GitHub organization

GitHub organizations are shared accounts where instructors and students can collaborate across many projects at once. Using your account, you can create a new GitHub organization by clicking on your profile icon in the upper right hand corner, go to *Settings*, *Access*, *New Organization*. It's suggested to name this organization the name of your class and the current semester you teaching in (i.e., sta199-s23). Once your organization is created we can use packages within R and R-studio to efficiently invite students and create assessments and interactive lessons for them that they can access on their own laptop device.

4 Why R & R-Studio

R is a statistical programming language for computing and modeling while R-studio is an integrated development environment for R (cite). We choose these resources because they are freely available to download and use, as well as in high demand in different data science positions post graduation (cite). In addition, this programming language has a comprehensive library of packages that allows students to create data visualizations and seamlessly

analyze data. For our course, we choose to accomplish learning objectives through predominantly through the `tidyverse` package in R. This package helps students import, tidy, transform, visualize, model, and communicate results. Further, this software is compatible with online versions that students can access without having to go through a local installment if they do not wish or are unable. There exists online versions that can be set up for students, including pre-populating their software with the appropriate packages needed for the semester.

4.1 Setting Up R & R-Studio

In an introductory course, it is recommended to minimize student frustration and distraction through the use of pre-packaged computational infrastructure (Çetinkaya-Rundel and Rundel, 2018). Per this recommendation, STA199 has students use R and R-studio through a *Duke Container*. Duke containers provides instructors at Duke university the opportunity to facilitate the use of different software, such as R and R-studio, through an online container instead of needing students to locally download both programs. Additionally, instructors have the ability to manage and install packages students will need for the semester, helping provide a neat and well organized starting experience with a new statistical language. (insert discussion on how this is done).

(Thoughts: Transition to R-studio cloud discussion? What alternatives can instructors use to imitate Duke containers if this is something they do not have access to at their university.)

4.2 Using R & R-Studio to manage GitHub organization

To invite students into your GitHub organization, we will use a myriad of functions from the `gh_class` package. (insert discuss on what the gh-class package is).

To start, we suggest creating a separate repository that will contain all code used to invite students to your organization and to create AEs and assessments for them. Within a document in this new repository, we must first define our GitHub organization as a character string to be used in later function. We suggest calling this `org`.

```
org <- "sta-199-s23"
```

The following code invites students to your GitHub organization using your defined org and the GitHub usernames collected from your students in a previous survey. For the example below, we chose to name the student survey "roster.csv".

```
# invite members ------------------------------------------------------------


roster <- read_csv("roster.csv")


org_invite(
  org = org,
  user = roster$gihub_name
)
```

Once the invites are sent, students will need to accept the invitation to the organization.

Streamlining teaching through R, R-studio, and GitHub greatly alleviates common challenges that arise when working with a large class size, such as activity and assignment

distribution. This includes the creation and distribution of in-class AEs, lectures, labs, and assessments. In the following sections, we discuss pedagogy, strategies on how to implement in the classroom, and detail the creation of such items using R, R-studio, and GitHub during a typical week in the semester (Figure 2).

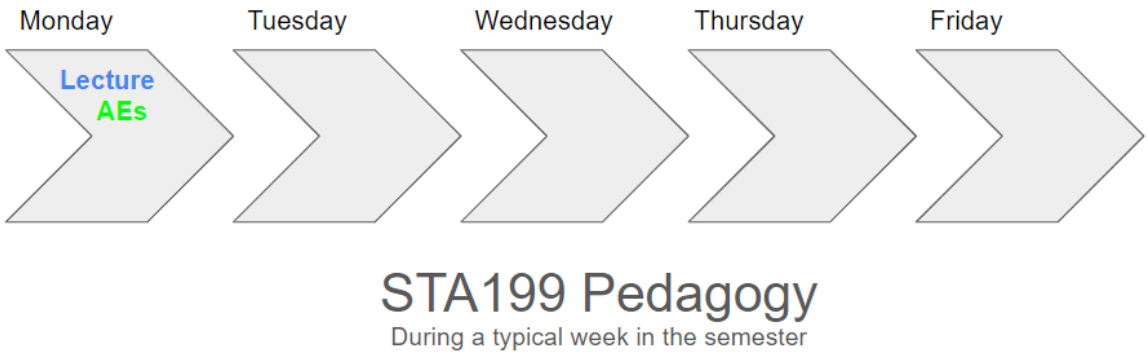

Fig 2: Model of pedagogical choices used in STA199 for a typical week

5 Pedagogy

In STA199, we have chosen a combination of teaching methods, interactive activities, and learning assessments to help prepare introductory data science students the tools they need to be successful outside of university or in future coursework. Our pedagogy includes a combination of lectures, facilitating in-class AEs, running a lab, and assigning assessments to provide students an opportunity to perform what they've learned.

5.1 Lecture & Application Exercises

During a typical week in the semester, class is held twice a week. Class is a combination of lecture and interactive activities called application exercises (AEs). The amount of time dedicated to lecture and AEs can and should very by instructor. Typically, we allot the

first 15 minutes and the last 10 minutes of class for lecture. During this time, students are introduced to new content that will be re-addressed in the AE. To help maximize engagement of such a large class size, we allot more class time to individual and team-based AEs where students are asked to code on their own, together, or along with the instructor.

Application exercises are structured interactive lessons that allow students to learn through doing in class. To participate in AEs, students are expected to bring their laptops to class. This expectation is made clear prior to the first day of teaching. The purpose of these exercises are to give students the opportunity to practice apply the statistical concepts and code introduced in the prepare videos, readings, and anything introduced during lecture. When first starting to create an AE, we suggest streamlining the process through a GitHub template repository cloned into R-studio as a project. We choose to use *Quarto* within R-studio to create AEs. This choice is intentional, to provide students the opportunity to practice writing code and answering questions in a reproducible format supported by such programming.

When creating an AE, we suggest creating a mix of coding and concept questions (e.g., fill in the code blanks, short response questions) that encourage students to follow along with instructor demonstrations and also answer questions on their own. This format has largely been accepted and appreciate by students: "I really enjoy the AE's and that you take the time to walk us through the code and answer questions; I like how the ae gives us a chance to practice the skills on our own after class as well. Additionally, we suggest that questions are scaffold, meaning that if students fall behind or type incorrect code when trying to initially follow along, that they have the ability to continuing engaging for the remaining AE. This is especially important at the beginning of the semester ot minimize students'

frustration around a new coding language. This may include having answers to questions in a separate Quarto document that students have access to, or within the same document as the AE. Moreover, we suggest clearly labeling where students are expected to type out answers in text or code throughout the exercise to further streamline their involvement and continue to minimize frustration.

An example on an AE used to help teach (insert concept) can be found here:

We highly recommended designing lessons with built in time to have students work on their own. The amount of time can largely depend on the question being asked, and the teaching style of the instructor. We have found that multiple 3-5 minute blocks for students to answer questions without the instructor's guidance works well. This has been expecially effective if the code is then created together as a class, built off student responses, to provide immediate feedback before moving on to the next set of material or questions.

In order to distribute this template AE to students within the class GitHub organization, we will use the function `org_create_assignment` from the `gh_class` package. Example code can be seen below.

```
assignment <- "AE-5" #name of AE repository you want to distribute


student_name <- org_members(org = org, include_admins = FALSE) #name of students to c


org_create_assignment(
  org = org,
  repo = paste0(assignment, "-", student_name),
  user = student_name,
```

```
  source_repo = paste0(org, "/", assignment)
)
```

Hands on learning through AEs are the backbone of learning in STA199. We have found that providing students with the opportunity to code in real-time during class keeps them more engaged and eager to continue learning about the material versus simply lecturing for a 75-minute period. This strategy has received positive feedback from students with varying degrees of coding and statistics experience.

As highlighted in previous sections this system, streamlined through GitHub, encourages time investment into the creation of AEs. Investing time into AE creation provides a strong environment for current students to learn, and sets a foundation for AEs to be re-tooled instead of recreated in subsequent semesters. Future renditions of this course can be modified from previous GitHub organizations efficiently, saving instructors valuable time and energy during the semester.

AEs are designed to be a low stakes that students can earn full credit on by completing in class. Typically, we make AEs worth completion points that total to be 5% of a student's end of semester grade. For students that do not show up to class, we make AEs due three days from when the AE was assigned. At the end of the semester, if student's have completed 80% of AEs, they earn a 100% for their grade. Students turn in AEs by pushing their answers within the AE document up to their repository. There have been mixed responses from both students and instructors on assigning a grade to AEs. For one, there is not an efficient way to implement a hard deadline that students must adhere to, without removing push access from their AE repository all-together. Further, if an instructor does not finish an AE in class, students who did not attend can be easily confused on what's

expected of them to complete. We suggest tailoring the decision of grading of AEs to your individual course as you see fit.

5.2 Labs

Labs for STA199 meet once a week for 75 minutes, and allow for students to apply concepts found in prepare material, lecture, and AEs to various data analysis scenarios. Lab section sizes are kept to around 30 students so each have more of an opportunity to converse with each other and the lab leader when working through the assessment.

Much like AEs, the instructor or head TA will create and clone a lab repository to all students that has the assessment and any other intangibles (like a data set) in it by changing the `assignment` vector to contain the name of original lab repository. Roughly one-third of the way into the semester, students are assigned to groups to complete a class project. We suggest strategically assigning groups based on a collection of the following information

-
-
-
- …

Additionally, from this point forward in the semester, we suggest having all labs be completed as group work. Introducing group work during lab can help students learn from different perspectives, practice their communication skills, and improve their problem solving skills in the context of statistics and data science. When groups are formed, they are tasked to come up with a team name. This team name will be the name used to create their team repositories for the remaining labs. To create team repositories, we use the following collected information from students: lab section; team name; last name; first name. Please

visit (insert link) to see an example of the format explained. Note, these are fake student names and have no meaning or association with Duke University.

We can use these data and change the code found in (figure-1??) to create repositories for each team.

```
teams <- read_csv("teams.csv")


org_create_assignment(
  org = org, # name of your organization
  repo = paste0(assignment, "-", teams$team), #name of the repo you are cloning and t
  user = teams$Github_Username, #adding individual students to team repo
  team = teams$team, #team (or group) they are added to
  source_repo = paste0(org, "/", assignment) #naming repo
)
```

The new expectation is that once groups are formed, one lab assessment will be turned in for each group instead of each individual.

After groups are formed, we give a set of recommendations for groups to help ensure a successful and positive group dynamic. This includes:

- Establish a clear line of communication with all members using Slack or another form of communication
- Share ideas. Let your voice be heard.
- Teach each other.
- Do not approach group work as a bunch of individual assignments.

To further ensure positive group dynamic, we initiate three peer review surveys. These peer review surveys provide …… Questions range from only having instructor visibility, to being shared with all team members to promote productive conversations. An example question is as follows:

(insert example question)

Experience talking points for labs:

- Merge Conflicts (hesitancy to cause one)
- Lots of work for large class sizes (groups)
- Pros and cons of group formation
- These are run by TAs (pros + cons)
- Peer review question types

5.3 Assessments

Discussion

Kokkelenberg, E. C., Dillon, M. & Christy, S. M. (2008), 'The effects of class size on student grades at a public university', *Economics of Education Review* **27**(2), 221–233.
URL: *https://www.sciencedirect.com/science/article/pii/S0272775707000271*

Schwab-McCoy, A., Baker, C. M. & Gasper, R. E. (2021), 'Data science in 2020: Computing, curricula, and challenges for the next 10 years', *Journal of Statistics and Data Science Education* **29**(sup1), S40–S50.
URL: *https://doi.org/10.1080/10691898.2020.1851159*

Schweser, K. (2023), 'Our philosophy'.

URL: *https://www.schweser.com/about-kaplan/philosophy*