EMORY
ROLLINS SCHOOL OF PUBLIC HEALTH

DEPARTMENT: Biostatistics and Bioinformatics

COURSE NUMBER: INFO550 **SECTION NUMBER:**

CREDIT HOURS: 2 **SEMESTER:** Fall 2020

COURSE TITLE: Data Science Toolkit

CLASS HOURS AND LOCATION:
A single, two-hour lecture per week

INSTRUCTOR NAME: David Benkeser

INSTRUCTOR CONTACT INFORMATION

EMAIL: benkeser@emory.edu

PHONE: (404)712-9975

SCHOOL ADDRESS OR MAILBOX LOCATION: 1518-002-3AA

OFFICE HOURS

Teaching Assistant(s): TBD

COURSE DESCRIPTION

This course is an elective for Masters and PhD students interested in learning some fundamental tools used in modern data science. Together, the tools covered in the course will provide the ability to develop fully reproducible pipelines for data analysis, from data processing and cleaning to analysis to result tables and summaries. By the end of the course students will have learned the tools necessary to: develop reproducible workflows collaboratively (using version control based on Git/GitHub), execute these workflows on a local computer (using command line operations, RMarkdown, and GNU Makefiles), execute the workflows in a containerized environment allowing end-to-end reproducibility (using Docker), and execute the workflow in a cloud environment (using Amazon Web Services EC2 and S3 services). Along the way, we will cover a few other tools for data science including best coding practices, basic python, software unit testing, and continuous integration services.

PRE-REQUISITES

Many topics covered will involve the R programming language and so **familiarity with R is needed** (e.g., BIOS 544/545 or similar level of competency). Necessary skills include: reading data into R, basic data cleaning in R (e.g., subsetting data, finding

missing values, merging data), operating on data.frames (e.g., changing column names, row names, summarizing rows/columns of data using simple statistics), basic graphics (e.g., plot or ggplot2).

Given the similarities between python and R, students with a background in python programming should also be equipped to succeed in the class, but will possibly require more effort to get up to speed with R.

COURSE LEARNING OBJECTIVES

- Understand why automation is a key element of reproducible data science.
- Develop reproducible workflows for data cleaning, analysis and report generation using the suite of tools learned in the class.
- Understand the importance of version control and best practices for collaborative coding projects.
- Use Docker to develop containerized workflows.
- Set up and work in Jupyter notebooks.
- Utilize cloud computing services for computation and storage.

CONCENTRATION COMPETENCIES:

- Develop public health information systems to support public health efforts
- Identify software for the interface of data entry and statistical analysis
- Assess individual data elements and display results effectively and appropriately
- Apply standard statistical methods in the analysis of public health information

EVALUATION

Students will be evaluated based on components developing into a single project over the course of the semester. Each student should pick an analysis or workflow that they want to develop into a reproducible pipeline. The assignments will be graded week-by-week and will develop into a fully reproducible pipeline that can be executed in a containerized environment both locally and on the cloud.

Ideally, students enrolled in the course will have an existing analysis or data cleaning/visualization problem that they can use for the course (e.g., a thesis or dissertation project), so that the developments throughout the semester are specifically relevant to each student.

Project requirements

Each project must involve the following three components (examples follow each separate component):

- Data manipulation – e.g., reading data from a file and cleaning the data; downloading data from the internet in an automated way; splitting data into training and test sets.

- Data analysis – e.g., making a Table 1 with descriptive summaries of key elements of the data; fitting regression models that answer scientific questions of interest; training machine learning algorithm and summarizing performance.
- Report writing – a report must be generated in the end that includes at least one table and at least one figure, along with writing.

FAQ on project selection:

Q: What if I don't have a data set/workflow to use?
A: We will work with you to identify one. Possible examples could be based on a publicly available data set from an online data repository (e.g., UCI ML library or Kaggle). The workflow could involve producing a report that summarizes descriptive statistics and produces basic visualizations of the data, or one that involves fitting a ML algorithm to the data and summarizing its predictive performance.

Q: What if I have data, but I can't make it public?
A: You can create a mock data set with similar format to your real data, but with made up values. You will produce a pipeline for analyzing the mock data and, since everything we do is reproducible, you can swap in your real data at a later date and still take advantage of the pipeline.

Q: Does the analysis need to involve complicated statistical methods?
A: No. Basic analyses are fine. The goal is not to evaluate the contents of the analysis itself, but rather whether the analysis, whatever it may be, can be executed in a reproducible way.

If you have questions about the suitability of a project, please contact the professor.

GRADING

Each week, an assignment will be given to develop a new aspect of the analysis pipeline. Each student will be responsible for turning in a weekly update of their pipeline. Submissions will occur in the form of a tagged GitHub repository online.

Each week, every student will be assigned to peer review another student's submission and provide comments/feedback on its usability. Remember that the overarching goal is the develop workflows that are reproducible and easy for another researcher (or you, at a future date) to use. As such, you will be evaluating your peers on the usability of their submission each week. A standardized rubric for peer review will be provided each week. An example is below.

Example assignment:
Write an Rmarkdown script that performs your analysis and generates an html report detailing the results. Include a README that describes (1) the process needed to compile the report; (2) any packages that need to be installed to compile the report; (3)

the expected output of the report. Push your repository to your GitHub account and tag the final release “hw2.”

Example grading rubric:
You have been assigned Student X for peer reviewer (GitHub user id: student_x). Clone the tagged release hw2 from the master branch of the repository student_x/info550 from GitHub. Please score your peer on the rubric below

1. Were you able to successfully clone the repository? Yes = 1pt, No = 0pt
   If No, describe what errors you encountered.

2. Rate the README on its readability. Do you clearly understand the steps needed to compile the report? Clear = 3pt; Some aspects unclear = 2pt; Many aspects unclear = 1pt; No README/not at all clear = 0pt.
   If some aspects rated as unclear, what were they?

3. Did the report successfully compile? Yes, with no effort on my part = 3pt; Yes, with some effort on my part (e.g., changing file paths, anything that took < 2 min.) = 2pt; No = 0pt.
   If any effort was required, describe what you had to do. If not able to compile, describe the errors encountered.

4. Did the report contain at least one figure and one table? Yes = 3pt; No = 0pt.

The TA/professor will review each peer review and submission and assign a grade accordingly. Each student will receive a grade our of 10 points for their own pipeline (based on peer /TA review) and will receive an additional 10 points for completing their peer review on time. Thus, each assignment will be scored as a percentage of 20 total points. Each weekly assignment will be worth 10% of the final grade.

Grade scale*:

- A = 93 -- 100%
- A- = 90 -- 93%
- B+ = 87 – 90%
- B = 83 – 87%
- B- = 80 – 83%
- C = 65 – 80%
- F = <65%

*final grades are not rounded and the lower limit of each letter grade is inclusive, so e.g., 93.0 is an A, while 92.9 is an A-.

COURSE STRUCTURE

The course will be organized into weekly lectures consisting of a combination of electronic slides and demonstrations. Students are expected to ask and answer questions in class. Students are encouraged to bring a laptop to class to follow along with demonstrations.

Example assignments are shown below along with the associated core competency.

Assignment	Competency
Properly install software for reproducible research: R, R Studio, homebrew (Mac), Git, text editor (e.g., Notepad++ or Sublime Text), LaTex, GNU make	Identify software for the interface of data entry and statistical analysis
Build a GitHub repository, tag a release of your code, and write a proper README file	Develop public health information systems to support public health efforts
Develop an R Markdown document performing a reproducible statistical analysis and summarizing results visually	Apply standard statistical methods in the analysis of public health information Assess individual data elements and display results effectively and appropriately
Build a Docker container for running a fully reproducible analysis pipeline	Develop public health information systems to support public health efforts
Execute basic python commands in a Jupyter notebook.	Identify software for the interface of data entry and statistical analysis
Set up an EC2 instance and S3 bucket on AWS suitable for a reproducible data analysis in the cloud.	Develop public health information systems to support public health efforts

COURSE POLICIES

Students are expected to attend lectures and ask questions during class. For computational assignments, students are encouraged, but not required, to bring a laptop to class to follow along with code demonstrations.

As the instructor of this course I endeavor to provide an inclusive learning environment. However, if you experience barriers to learning in this course, do not hesitate to discuss them with me and the Office for Equity and Inclusion, 404-727-9877.

RSPH POLICIES

Accessibility and Accommodations

Accessibility Services works with students who have disabilities to provide reasonable accommodations. In order to receive consideration for reasonable accommodations, you must contact the Office of Accessibility Services (OAS). It is the responsibility of the

student to register with OAS. Please note that accommodations are not retroactive and that disability accommodations are not provided until an accommodation letter has been processed.

Students who registered with OAS and have a letter outlining their academic accommodations are strongly encouraged to coordinate a meeting time with me to discuss a protocol to implement the accommodations as needed throughout the semester. This meeting should occur as early in the semester as possible.

Contact Accessibility Services for more information at (404) 727-9877 or accessibility@emory.edu. Additional information is available at the OAS website at http://equityandinclusion.emory.edu/access/students/index.html

Honor Code

You are bound by Emory University's Student Honor and Conduct Code. RSPH requires that all material submitted by a student fulfilling his or her academic course of study must be the original work of the student. Violations of academic honor include any action by a student indicating dishonesty or a lack of integrity in academic ethics. *Academic dishonesty refers to cheating, plagiarizing, assisting other students without authorization, lying, tampering, or stealing in performing any academic work, and will not be tolerated under any circumstances.*

The RSPH Honor Code states: "Plagiarism is the act of presenting as one's own work the expression, words, or ideas of another person whether published or unpublished (including the work of another student). A writer's work should be regarded as his/her own property." (http://www.sph.emory.edu/cms/current_students/enrollment_services/honor_code.html)

COURSE CALENDAR AND OUTLINE

Topics and dates are subject to change as the semester progresses.

Date	Topics
	Command line
	Basic shell scripting
	R Markdown
	Best coding practices
	Version control using Git/GitHub
	GNU Makefiles
	Containerization
	Intro to python
	Jupyter notebooks
	Cloud computing and storage
	Continuous integration services